Proceedings of the
1st Workshop on Artificial Intelligence and Formal Verification, Logics, Automata and Synthesis (OVERLAY),
Rende, Italy, November 19–20, 2019

Optimization and Multistage Systems. The Thawing Case

E. Pippia[1,3], A. Bozzato[2,3], E. Tiberi[3], R. Furlanetto[3], and A. Policriti[1]

[1]Dipartimento di Scienze Matematiche, Informatiche e Fisiche - University of Udine, Via delle Scienze 206, 33100 UD, Italy
[2]Dipartimento di Scienze Agroalimentari, Ambientali e Animali - University of Udine, Via delle Scienze 206, 33100 UD, Italy
[3]The Research Hub by Electrolux Professional S.p.A., Viale Treviso 15, 33170 PN, Italy

Abstract

Multistage systems are systems characterized by n stages, where each stage has a precise parametric evolution law. We propose a procedure to reduce the optimal parameter synthesis on a multistage system to the optimal control design on an automaton with a DAG structure that we will call multistage automaton. We describe the procedure using a thawing system as a case study. We address the optimization problem as a reachability problem and we use backtrack strategy to obtain a feasible solution in the linear case.

Keywords: Hybrid Automata, Cyber-physical systems, Optimal Parameter Synthesis, reachability, backtraking.

1 Introduction

Hybrid systems are dynamical systems combining both discrete and continuous dynamics. This type of systems arise in a large number of application areas and their analysis is a well-established tool in both academic and industry researches.

The system we want to analyse in this paper is a multistage cooking program and (qualitatively/ quantitatively) governing this kind of processes is fundamental for most professional food appliances. Professional chefs, in fact, use different heating methods to bring more flavour to the food or to control the final results. For example, they may start with a short grill for searing a steak and then continue the process at low temperature for a long time. Professional appliances allow to program a cooking cycle by setting up one or more modes, instead of setting different heating stages manually.

Lifting up our view from the single application, we want to focus on the problem of parametrically designing what we can call *multistage systems*, that are systems characterized by n stages, where each stage has a precise—albeit parametric—evolution law. We give multistage systems as a single hybrid automaton [2], whose states are grouped in *layers* and states are characterized by a continuous (parametric) evolution law. Each layer qualitatively represents a stage of the entire process and the discrete transition represents the switching instant from one stage to the next one. The underlying structure of the automaton will be that of a *Directed Acyclic Graphs* (DAG) and global properties of the process will be optimized with respect to full paths in such a DAG.

A multistage system is, ultimately, a parametrized hybrid system. In the literature we can find two main classes of problems analyzed on these systems: *parameter synthesis* and *optimal control design.* Parameter synthesis is simply the problem of choosing a set of parameters in order to ensure a given property. In [9, 8, 7] (among many others) we can find possible approaches to solve this problem. Optimal control design is the problem of choosing the best *trajectory* that satisfies a given property, where with trajectory we mean a possible (full) evolution of the system. Also here we have many examples of possible approaches, such as [12, 4, 10]. The problem we are considering in this paper is a combination of the two: we want to select a setting of parameters suitable to obtain the best trajectory that satisfies a given property.

In the literature this problem goes under the name of *optimal parameter synthesis* (or *safe reachability problem*) and is tackled specifically in [3, 5]. In [5] the authors perform optimal parameter synthesis but they are working with *transient* systems and not with multistage systems as in our case. In a transient system you just have the set of states (finite or infinite) and a transition relation expressed with a formula, that in [5] is a linear arithmetic formula. They use model-checking to obtain the set of all admissible solutions and they illustrate different methods to explore the solution-space in order to find the best solution. [3], instead, are addressing a scheduling problem for a reducing peak power demand of the heating system of multiple zones for a commercial building [13]. In this problem a solution is an *un*ordered list of parameters with the associated duration. The authors are not interested in the specific stage for which we select a given parameter, for this reason they can reduce the optimal parameter synthesis to a *linear* optimization problem.

In this paper we propose a procedure to solve optimal parameter systhesis problem working directly on the hybrid automaton. We work on the structure of the automaton in order to reduce the optimal parameter synthesis on a multistage system to the optimal control design on an automaton with an augmented number of locations—that we will call *multistage automaton.* Doing this we address the optimization problem as a reachability problem and we use a backtrack strategy [1] to obtain a feasible solution in the linear case.

2 Definitions for Hybrid Automata

The notion of hybrid automata was introduced in [2] as a model and specification language for hybrid systems. The following definitions are standard notations of hybrid automata (see [2, 6] for more details).

Definition 2.1. *An* Hybrid Automaton $\mathcal{H} = \langle \mathbf{Z}, \mathbf{Z}', \mathcal{V}, \mathcal{E}, Inc, Dyn, Act, Reset \rangle$ *consists of the following components:* $\mathbf{Z} = (Z_1, \dots, Z_k) \in \mathbb{R}^k$ *and* $\mathbf{Z}' = (Z'_1, \dots, Z'_k) \in \mathbb{R}^k$ *are two vectors of variables;* $\langle \mathcal{V}, \mathcal{E} \rangle$ *is a finite directed graph: the vertices in* $\mathcal{V}$ *are called* locations *and the edges in* $\mathcal{E}$ *are called* transitions*.*

Each vertex $v \in \mathcal{V}$ *is labelled by two formulas:* $Inv(v)[\mathbf{Z}]$ *that is an* invariant*, i.e. a constraint that the variables* $\mathbf{Z}$ *must respect during their evolution;* $Dyn(v)[\mathbf{Z}, \mathbf{Z}', t]$ *that is the* dynamic law*, in particular for each* $v \in \mathcal{V}$ *there is a continuous function* $f_v : \mathbb{R}^k \times \mathbb{R}_{\geq 0} \to \mathbb{R}^k$ *such as* $Dyn(v)[\mathbf{Z}, \mathbf{Z}', t]$ *is* $\mathbf{Z}' = f_v(\mathbf{Z}, t)$ *where the variable* t *represents the time of the system and range over* $\mathbb{R}_{\geq 0}$*.*

Each edge $e \in \mathcal{E}$ *is labelled by other two formulas:* $Act(e)[\mathbf{Z}']$ *is the* activation*, i.e. a constraint on the activation of the edge* e*;* $Reset(e)[\mathbf{Z}, \mathbf{Z}']$ *is the* reset *of the variables while we are using the edge* e*.*

Definition 2.2 (State)**.** *A* state σ *is a pair* $(v, \mathbf{x})$ *considering a location* $v \in \mathcal{V}$ *and a valuation* $\mathbf{x} \in \mathbb{R}^k$ *for the variables* $\mathbf{Z}$ *such as* $Inv(v)[\mathbf{x}]$ *holds.*

Definition 2.3 (Transition relations)**.** *Let* $\mathcal{H}$ *be a hybrid automaton. We have two type of transition relations between two states* $\sigma = (v, \mathbf{x})$ *and* $\delta = (w, \mathbf{y})$*:*

continuous transition relation $\sigma \to_t \delta \Leftrightarrow v = w$ *and there exists* $\mathbf{f} : \mathbb{R}_{\geq 0} \to \mathbb{R}^k$ *continuous function such that* $\mathbf{x} = f(0)$*,* $\mathbf{y} = f(t)$ *and for each* $t' \in [0, t]$ *holds* $Inv(v)[\mathbf{f}(t')]$ *and* $Dyn(v)[\mathbf{x}, \mathbf{f}(t'), t']$

discrete transition relation $\sigma \to_e \delta \Leftrightarrow e \in \mathcal{E}$ *such that* $e = (v, w)$ *and holds* $Inv(v)[\mathbf{x}]$*,* $Inv(w)[\mathbf{y}]$*,* $Act(e)[\mathbf{x}]$ *and* $Reset(e)[\mathbf{x}, \mathbf{y}]$

Definition 2.4 (Trajectory)**.** *Let $\mathcal{H}$ be a hybrid automaton. A* trajectory *is a sequence $(\sigma_j)_{j\in J}$ of states such that: $\forall j \in J \setminus \{0\}$ we have $\sigma_{j-1} \to_e \sigma_j$ with $e \in \mathcal{E}$ or $\sigma_{j-1} \to_t \sigma_j$ with $t > 0$ and if $|J| > 2$ we can not have two continuous transition relations between three consecutive states.*

Definition 2.5 (Reachability)**.** *Let σ and δ be two states of a hybrid automaton $\mathcal{H}$. The state δ is* reachable *from the state σ if there is a trajectory of $\mathcal{H}$ that starts in σ and ends in δ. Given an automaton $\mathcal{H}$, a set of starting states Σ_S and a set of ending states Σ_E, the* reachability problem *is the problem to decide whether there exists a state $\sigma \in \Sigma_S$ from which a state in Σ_E is reachable.*

3 Thawing Optimization Problem

The target is to solve the optimal parameter synthesis for a multistage cooking program. The procedure we describe is very general and can be applied to every multistage system. For example, we can create a continuous "cutting-stock" problem thinking of a painting product-line of a metal tube factory, where we want to maximize the overall length of tubes produced, while keeping the paint consumption for each primary color under a supply limit threshold. As another example, we can apply our methodology to find an ordered solution for the peak power demand problem described in [13].

Here we want to focus on a special multistage heating process, the *thawing* process, that is the process of safely heat up a frozen foodstuff to just above 0°C. In professional applications, the thawing process can be executed inside a specific appliance able to keep the temperature of its cavity within the range $4°C$ - $16°C$. We can set n different *stages*, namely n different temperatures of the cavity in the form of a vector $T_{set} \in [4, 16]^n$ with the temperatures expressed in Celsius unit. In addition we require how long we want to keep these temperatures, so we fix another vector $time_{set}$ with $n - 1$ duration expressed in seconds. We define as *thawing cycle* the system obtained by specifying these parameters. A thawing cycle ends when the minimum temperature of the foodstuff reaches $0°C$.

In this process food safety becomes particularly important because the bacteria that cause food poisoning are found on the surfaces of many foods and they start growing due to the continuous arise of the temperature of these areas. There are different evolution models that take into account the bacterial growth depending on the temperature and the duration, mostly variants of the logistic evolution model or piecewise exponential growth curves obtained with empirical methods [14]. We are going to consider the easiest and the strictest constraint that we can impose. The same construction can be extended to more complex models. In summary, we want keep the maximum temperature $T_{\max}$ of the product below a T_{safe} temperature of $7°C$ following the codex alimentarius for fish fillet [11].

The aim is to select a thawing cycle in order to minimize the total duration of the process keeping bacterial growth under control by looking at the maximum temperature of the foodstuff.

The optimization problem described is quite general. We have a multistage system with $m(m-1)$ variables and we want to minimize the total time having an end condition—in the thawing case is a minimum threshold that we have to reach ($T_{\min} = 0°C$)—and satisfying a constraint that in this case is a maximum threshold that we do not want to overpass ($T_{\max} \leq 7°C$).

Example 3.1 (Linear multistage system)**.** *We introduce the following simplified (linear) version of the thawing cycle. Suppose we have only* 3 *stages and suppose we deal with two variables x_1, x_2 only, evolving linearly. x_1 is the minimum temperature and x_2 is the maximum temperature. Suppose $T_{set} \in \{4, 8, 12, 16\}^3$, so we have to select one of these temperature for each of the 3 stages. If we are in the stage i we select the temperature $T_{set}(i)$ and we have:*

$$x_1(t) = x_1(0) + \frac{T_{set}(i)}{8000}t; \qquad x_2(t) = x_2(0) + \frac{(T_{set}(i) - 2)}{4000} \cdot t \tag{1}$$

The starting values $x_1(0)$ and $x_2(0)$ for the first stage are $-18°C$ (typically the freezer temperature). The starting value in the second and third stage are the last value of the previous stage. In this example choosing T_{set} and $time_{set}$ we obtain a trajectory for x_1 and a trajectory for x_2 that are piecewise linear functions.

4 Hybrid Automaton model

A multistage system with n stages is inherently an hybrid model and the natural way to model it is with an hybrid automaton with $n+1$ locations: n locations for the n stages of the system, plus an extra location for the end condition. Our strategy will consist in increasing (step by step) the number of locations of a hybrid automaton till we reach what we call a *multistage automaton* appropriately modeling our scenario.

Definition 4.1 (Multistage automaton)**.** *A multistage automaton* $(\mathcal{H}, \mathcal{K}, g)$ *consists of the following components:*

$\mathcal{K}$ *a finite set of real parameters;*

$\mathcal{H} = \langle \mathbf{Z}, \mathbf{Z}', \mathcal{V}, \mathcal{E}, Inv, Dyn, Act, Reset \rangle$ *is an hybrid automaton;*

$\langle \mathcal{V}, \mathcal{E} \rangle$ *is a Direct Acyclic Graph (DAG) and* $\mathcal{V} = S \cup V \cup E$ *a partition of* $\mathcal{V}$ *in three not empty sets, such that* $s \in S$ *has only outgoing edges,* $e \in E$ *has only incoming edges, and* $v \in V$ *has at least one outgoing edge and at least one incoming edge;*

$g : V \to \mathcal{K}$ *is a labeling function;*

$\mathbf{Z} = (Z_1, \ldots, Z_n) \in \mathbb{R}^n$, $\mathbf{Z}' = (Z'_1, \ldots, Z'_n) \in \mathbb{R}^n$ *and there exists a function* $f : \mathbb{R}^n \times \mathbb{R}_{\geq 0} \times \mathcal{K} \to \mathbb{R}^n$ *such that* $Dyn(v)$*, for* $v \in V$*, has the following form:* $Dyn(v)[\mathbf{Z}, \mathbf{Z}', t] \equiv (\mathbf{Z}' = f(\mathbf{Z}, t, g(v)))$.

We work with the linear multistage system in the Example 3.1 to illustrate the associated multistage automaton. Set $\mathcal{K} = \{k_1, k_2, k_3, k_4\} = \{4, 8, 12, 16\}$. If we fix the parameter vector T_{set}, we can define a simple linear hybrid automaton $\mathcal{H}_L = \langle \mathbf{Z}, \mathbf{Z}', \mathcal{V}, \mathcal{E}, Inv, Dyn_f, Act, Reset \rangle$ as in Figure 1.

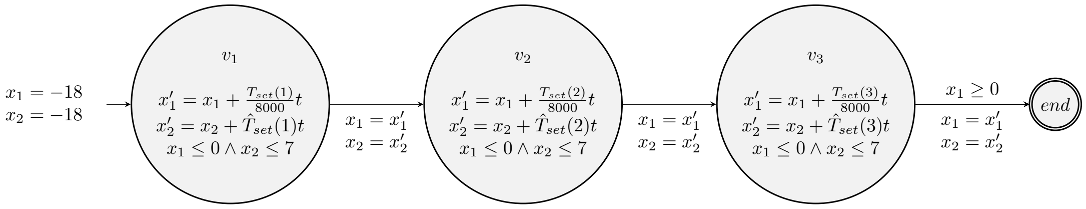
$x_1 = -18$
$x_2 = -18$
v_1
$x'_1 = x_1 + \frac{T_{set}(1)}{8000}t$
$x'_2 = x_2 + \hat{T}_{set}(1)t$
$x_1 \leq 0 \wedge x_2 \leq 7$
$x_1 = x'_1$
$x_2 = x'_2$
v_2
$x'_1 = x_1 + \frac{T_{set}(2)}{8000}t$
$x'_2 = x_2 + \hat{T}_{set}(2)t$
$x_1 \leq 0 \wedge x_2 \leq 7$
$x_1 = x'_1$
$x_2 = x'_2$
v_3
$x'_1 = x_1 + \frac{T_{set}(3)}{8000}t$
$x'_2 = x_2 + \hat{T}_{set}(3)t$
$x_1 \leq 0 \wedge x_2 \leq 7$
$x_1 \geq 0$
$x_1 = x'_1$
$x_2 = x'_2$
end

Figure 1: Linear Hybrid Automata $\mathcal{H}_L$. We indicate with $\hat{T}_{set}(i)$ the value $\frac{(T_{set}(i)-2)}{4000}$. Above each arrow (except the initialization) we indicate the activation condition (if it is no $True$), below we indicate the reset condition.

First of all we are going to explicit the role of the variables $T_{set}(i)$. These constants change the evolution of our variables but we have only a finite number of values to take into account, since the dimension of $\mathcal{K}$ is finite.

We split each stage of $\mathcal{H}_L$ in 4 different locations, one for each possible evolution. We add also a dummy starting node to initialize the variables. What we obtain is a graph with 3 internal layers as in Figure 2. We denote by $v_{i,j}$ a location on layer $i = 1, 2, 3$ with constant $k_j \in \mathcal{K}$. Every location at layer i is connected to all the locations at layer $i+1$. The layered hybrid automaton $\mathcal{L}$ in Figure 2 has:

$\mathbf{Z} = (x_1, x_2)$, $\mathbf{Z}' = (x'_1, x'_2)$; $(\mathcal{V}, \mathcal{E})$ as in Figure 2;

$Reset(e)[\mathbf{Z}, \mathbf{Z}'] \equiv (x_1 = x'_1 \wedge x_2 = x'_2)$ for $e \in \mathcal{E}$;

$Inv(v_{i,j})[\mathbf{Z}] \equiv (x_1 \leq 0 \wedge x_2 \leq 7)$ for $i = 1, 2, 3$ $j, k = 1, \ldots, 4$; $Inv(v)[\mathbf{Z}] \equiv True$ for $v = s, e$;

$Dyn(v_{i,j})[\mathbf{Z}, \mathbf{Z}', t] \equiv (x'_1 = x_1 + \frac{k_j}{8000}t \wedge x'_2 = x_2 + \frac{(k_j - 2)}{4000}t$ for $i = 1, 2, 3$ $j = 1, \ldots, 4$;
$Dyn(s)[\mathbf{Z}, \mathbf{Z}', t] \equiv (x'_1 = -18 \wedge x'_2 = -18)$ and $Dyn(e)[\mathbf{Z}, \mathbf{Z}', t] \equiv (x'_1 = x_1 \wedge x'_2 = x_2)$;

$Act((v_{3,j}, e))[\mathbf{Z}'] \equiv (x'_1 \geq 0)$ for $j = 1, \ldots, 4$ and the activation is $True$ for all the other edges.

For each internal location $v_{i,j} \in \mathcal{V}$ we can define the function $g(v_{i,j}) = k_j$ with $k_j \in \mathcal{K}$. It is evident by construction that the couple $(\mathcal{L}, \mathcal{K}, g)$ is a multistage automaton.

If we consider the multistage automaton $\mathcal{L}$ in Figure 2 and we solve a reachability problem from the location s to the location e we find just a feasible solution for the optimal parameter synthesis but not the optimal one. We need to make a step ahead adding the information of the total duration. In $\mathcal{L}$ we know that the final duration of our system can vary from 2.5 hours to 10 hours so we can split the location *end* in 9 locations, one per each hour (or more if we want to refine the interval). We obtain the multistage automaton $\bar{\mathcal{L}}$ as in Figure 3. We call e_3 the end location associated with $t = 3$ hours and in general e_h for the end location associated with $t = h$ hours.

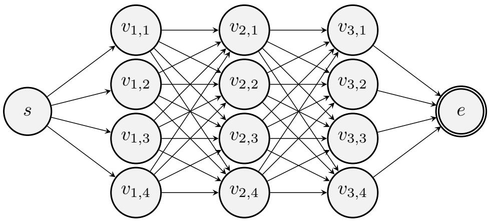

Figure 2: Layer Hybrid Automaton $\mathcal{L}$ with 3 layers

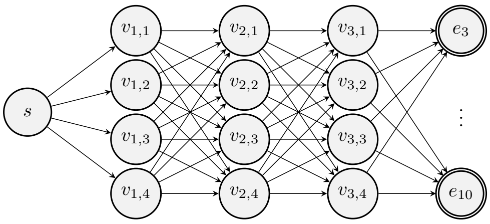

Figure 3: Layer Hybrid Automata $\bar{\mathcal{L}}$ with 3 layers

What we have to modify is the activation constraint adding the time bound:

$$Act((v_{3,j}, e_3))[\mathbf{Z}'] \equiv (x'_1 \geq 0 \wedge t \leq 3 \cdot 3600);$$

$$Act((v_{3,j}, e_{10}))[\mathbf{Z}'] \equiv (x'_1 \geq 0 \wedge 9 \cdot 3600 < t) \ ;$$

$$Act((v_{3,j}, e_h))[\mathbf{Z}'] \equiv (x'_1 \geq 0 \wedge (h-1) \cdot 3600 < t \leq h \cdot 3600)$$

If the reachability problem from the location s with $x_1 = -18°C$ and $x_2 = -18°C$ to the location e_3 have a solution, we have the existence of a feasible solution for the optimal parameter synthesis with a total duration ≤ 3 hours, vice versa if the reachability problem is empty we know that if a solution exists the optimal value is greater than 3 hours. In general solving the reachability problem on the automaton $\bar{\mathcal{L}}$ give us the existence of a feasible solution for the optimal parameter synthesis problem with a lower/upper bound to the optimal solution.

To conclude we need to be able to solve the reachability problem. In general, if the evolution are linear, we can solve the reachability problem from the set of starting states Σ_S to the set of ending states Σ_E using the backward approach introduced in [2, 1]. The procedure is effective and we can, for example, prove that the automaton $\bar{\mathcal{L}}$ does not have feasible trajectories with a duration less than 6 hours, hence all the reachability problems from s to $e_3, \ldots, e_6$ have a negative solution.

5 Conclusions and future works

We showed how we can reformulate a optimal parameter synthesis problem for a class of hybrid systems that we called multistage systems. For general multistage systems we obtain a multistage automaton, a hybrid automata with a layered DAG as underlying discrete structure. The key point was to reformulate the optimization problem as a reachability problem, use the backtrack technique introduced by Alur [2, 1] to prove the existence of a feasible solution, and obtain a lower/upper-bound to an optimal solution.

True thawing is, in fact, non-linear. The graph construction presented here is applicable to the non-linear case but additional ideas are needed to solve the reachability problem in that case. A further interesting direction consists, therefore, in studying the temperature-evolution in order to approximate its law—to start—with a piece-wise linear function.

References

[1] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, and et al. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138:3–35, 1995.

[2] R. Alur, N. Courcoubetis, C, T. A. Henzinger, and P.-H. Ho. Hybrid automata: an algorithmic approach to the specification and verification of hybrid systems. *Hybrid Systems*, 736:209–229, 1993.

[3] R. Alur, A. Trivedi, and D. Wojtczak. Optimal scheduling for constant-rate multi-mode systems. *Proceedings of the 15th ACM International Conference on Hybrid Systems: Computation and Control*, pages 75–84, 2012.

[4] A. Bemporad and N. Giorgetti. A logic-based hybrid solver for optimal control of hybrid systems. *Conference on Decision and Control*, 2003.

[5] B. Bittner, M. Bozzano, A. Cimatti, M. Gario, and A. Griggio. Towards pareto-optimal parameter synthesis for monotonie cost functions. *2014 Formal Methods in Computer-Aided Design (FMCAD)*, pages 23–30, 2014.

[6] A. Casagrande. *Hybrid System: A First-Order Approach to Verification and Approximation Techniques*. PhD thesis, Università degli Studi di Udine, 2006. pp. 23-34.

[7] A. Cimatti, A. Griggio, S. Mover, and S. Tonetta. Parameter synthesis with ic3. *2013 Formal Methods in Computer-Aided Design, FMCAD 2013*, pages 165–168, 2013.

[8] T. Dreossi. Sapo: Reachability computation and parameter synthesis of polynomial dynamical systems. *Hybrid Systems: Computation and Control (HSCC 2017)*, 2017.

[9] A. Étienne and S. Romain. *The Inverse Method.* Wiley, 2013.

[10] K. Hiraishi. Solving optimiation problems on hybrid systems by graph exploration. *8th International Workshop on Discrete Event Systems*, 2006.

[11] Joint FAO/WHO. *Standard for Quick Frozen Blocks of Fish Fillet, Minced Fish Flesh and Mixtures of Fillets and Minced Fish Flesh.* Codex Alimentarius, CODEX STAN 165-1989.

[12] K. G. Larsen and J. I. Rasmussen. Optimal reachability for multi-priced timed automata. *Theoretical Computer Science*, 390:197–213, 2008.

[13] T. Nghiem, M. Behl, R. Mangharam, and G. Pappas. Green scheduling of control systems for peak demand reduction. In *IEEE Conference on Decision and Control and European Control Conference*, pages 5131–5136, 2011.

[14] M. Peleg and M. Corradini. Microbial growth curves: What the models tell us and what they cannot. *Critical reviews in food science and nutrition*, 51:917–45, 2011.