Neurocomputing 149 (2015) 1620–1630

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Optimal switching between controlled subsystems with free mode sequence

Ali Heydari [a,*], S.N. Balakrishnan [b]

[a] Mechanical Engineering Department, South Dakota School of Mines and Technology, USA
[b] Mechanical and Aerospace Engineering Department, Missouri University of Science and Technology, USA

ARTICLE INFO

Article history:
Received 18 July 2013
Received in revised form
1 July 2014
Accepted 13 August 2014
Communicated by H. Zhang
Available online 28 August 2014

Keywords:
Optimal switching
Approximate dynamic programming
Switching system
Adaptive critics

ABSTRACT

The problem of optimal switching and control of nonlinear switching systems with *controlled* subsystems is investigated in this study where the mode sequence and the switching times between the modes are unspecified. An approximate dynamic programming based method is developed which provides a feedback solution for unspecified initial conditions and different final times. The convergence of the proposed algorithm is proved. Versatility of the method and its performance are illustrated through different numerical examples.

1. Introduction

A switching system is characterized by a group of subsystems with different dynamics of which one is active at each time instant. Hence, in order to control such systems one needs a switching schedule along with a control input to be applied. Examples of switching systems can be found in dynamical systems in different fields, from aerospace to chemical engineering [1–5]. There are a few developments in this area [6–15], however, still there are many open issues even for the case of linear subsystems with a quadratic cost functions [7,16]. It should be noted that the optimal switching problems are much more complicated compared to conventional optimal control problems due to the intercoupling between the effect of the applied continuous input and the discrete switching/scheduling. In other words, the presence of the *discrete events* in the system makes it very hard to find an optimal solution. Formulating an optimal control problem as a function optimization problem, the relation between *conventional optimal control problems* and *optimal switching problems* is similar to the relation between a smooth function optimization and a mixed integer programming, e.g., see [15]. The restriction that some of the variables can only assume integer values, in mixed integer programming, leads to different challenges including the non-convexity of the problem. Such challenges exist in optimal switching problems, as well.

Development in the field can be mainly classified into two categories. In the first category, the sequence of active subsystems, called mode sequence, is selected a priori [6–11], and the problem, i.e., finding the switching instants between the modes, is solved using nonlinear programming methods. In these papers, the gradient of the cost with respect to the switching instants/points is calculated. Afterward, the switching instants/points are adjusted to find the *local* optimum. An iterative solution to a nonlinear optimization problem is suggested in [10] and using the combination of this control approach with ideas from model predictive control, the authors developed the so-called crawling window optimal control scheme for the optimal switching problem. The second category is based on discretizing the problem space in order to deal with a *finite* number of options. Authors of [12] utilized a direct search to evaluate the cost function for different randomly selected switching time sequences among the finite number of options to select the best sequence. In [13], state and input spaces are discretized for calculation of the value function for optimal switching through dynamic programming. In [14], a genetic algorithm is used to find the optimal switching times among the choices.

All the cited methods work only with *a specific initial condition*; each time the initial condition is changed, a new set of computations needs to be performed to find the new optimal switching instants. In order to extend the validity of the results to different initial conditions within a pre-selected set, in [9] a solution is found as the local optimum in the sense that it minimizes the worst possible cost for all trajectories starting in the selected initial states set.

* Corresponding author. Tel.:+1 605 394 2200.
E-mail addresses: ali.heydari@sdsmt.edu (A. Heydari), bala@mst.edu (S.N. Balakrishnan).

http://dx.doi.org/10.1016/j.neucom.2014.08.030
0925-2312/

In the past two decades, approximate dynamic programming (ADP) has been shown to have a lot of promise in providing comprehensive solutions to conventional optimal control problems in a feedback form [17–28]. ADP is usually carried out using two neural network (NN) syntheses called adaptive critics (ACs) [18–20]. In the heuristic dynamic programming (HDP) class with ACs, one network, called the *critic* network, maps the input states to output the cost and another network, called the *actor* network, outputs the control with states of the system as its inputs [20,21]. Motivated by the potentials of ADP in conventional problems, different ADP-based methods were developed for solving optimal switching problems [29–33]. The authors of [29,30] developed a switching method in which the number of functions subject to approximation grows exponentially with the number of iterations and an idea is suggested by the authors for eliminating some of these functions. The authors of this study also investigated the ADP-based approaches to optimal switching [31–33]. The developments in [31,32] are for optimal switching problems with *fixed mode sequence*. In such systems, the mode sequence and the number of switching are given and the problem is finding the optimal switching time. The problem in the current study, however, is much more complicated due to the fact that the mode sequence, the number of switching, and the switching times are all unknown and subject to be calculated such that a cost function is optimized. As compared with [33], the main difference is assuming a *controlled* subsystem in the current study as opposed to *autonomous* subsystems investigated in [33]. Investigation of controlled subsystems makes the problem more complicated due to the intercoupling that exists between the effect of *switching* between the modes and applying different *control inputs* once a mode is active. Another challenge is both finding the switching schedule and the continuous input in the problem subject to this study.

Considering this background, the contribution of this study is presenting an ADP-based solution to switching problems with controlled subsystems and free mode sequence. The idea is as simple as learning the optimal cost-to-go and the optimal control for different active modes. It is shown that having these functions the optimal mode can be found in a feedback form, i.e., as a function of the instantaneous state of the system and the remaining time. An algorithm is developed which fits in the category of HDP for learning the desired functions. The proof of convergence is also provided. This method has several advantages over existing developments in the field: (1) it provides *global* optimal switching (subject to the assumed neural network structure) unlike the nonlinear programming based methods [6–11] which could provide only local optimal solutions. (2) The order of active subsystems and the number of switching are free, as opposed to simpler problems of having a fixed mode sequence [6–14,31,32]. (3) The neurocontroller determines an optimal solution for *unspecified initial conditions*, without needing to retrain the networks. (4) Once trained, the neurocontroller gives solution to *any other final time as well*, as long as the new final time is not greater than the final time for which the network is trained. (5) The switching is scheduled in a *feedback* form, hence, it has the inherent robustness of feedback solutions in disturbance rejection, unlike open loop developments [6–9,11–15].

The closest development in the literature to the current study is Ref. [30]. The differences which highlight the advantages and disadvantages of each method, are fourfold. (a) Ref. [30] presents an algorithm through which neural networks are utilized in approximating *smooth* functions, which potentially leads to more accurate approximations as compared with the algorithm presented in the current study which is based on approximating possibly non-smooth functions. (b) The number of functions needed to be learned at each training iteration and stored for online control grows exponentially with the number of iterations, in Ref. [30]. But, in the current study only one critic and as many actors as the number of subsystems are required to be trained. (c) The development in Ref. [30] admits a hard terminal constraint on the state, while the method presented here admits soft terminal constraints. (d) The algorithm proposed in Ref. [30] trains the NNs for a single selected initial state vector, but, the current study leads to a very versatile neurocontroller in terms of being able to control different initial conditions and different final times, without requiring the weights to be re-tuned.

The rest of this paper is organized as follows: the problem formulation and the solution idea are presented in Section 2. Approximations of the optimal cost-to-go and the optimal control with neural networks are explained in Section 3. Numerical analyses are given in Section 4 and conclusions from this study are given in Section 5.

2. Problem formulation and solution idea

A switching system with nonlinear input-affine subsystems can be represented by the set of M subsystems or modes modeled by:

$$\dot{x}(t) = \bar{f}_{v(t)}(x(t)) + \bar{g}_{v(t)}(x(t))u(t), \tag{1}$$

where functions $\bar{f}_v : \mathbb{R}^n \to \mathbb{R}^n$ and $\bar{g}_v : \mathbb{R}^n \to \mathbb{R}^{n\times m}$, $\forall v \in \mathbb{V} \equiv \{1, 2, \ldots, M\}$, represent the dynamics of the subsystems and are assumed to be smooth. Integers n and m denote the dimensions of state vector x and control vector u, respectively. The continuous time is denoted with t and the initial and final times are denoted with t_0 and t_f, respectively. Controlling the switching systems requires a *control input*, $u : [t_0, t_f) \to \mathbb{R}^m$, and a *switching schedule*, $v : [t_0, t_f) \to \mathbb{V}$. The latter determines the active subsystem at time t and the former provides the input to the active subsystem. The optimal solution, however, is a solution that minimizes cost function:

$$J = \psi(x(t_f)) + \int_{t_0}^{t_f} (\bar{Q}(x(t)) + u(t)^T \bar{R} u(t))dt. \tag{2}$$

Convex positive semi-definite functions $\bar{Q} : \mathbb{R}^n \to \mathbb{R}$ and $\psi : \mathbb{R}^n \to \mathbb{R}$ penalize the states and $\bar{R} \in \mathbb{R}^{m\times m}$ is a positive definite matrix penalizing the control effort, in the selected cost function. The problem is determining an input history $u(t)$ and a switching history $v(t)$ such that cost function (2) is minimized, subject to dynamics (1).

The selected approach in this study for solving the problem is ADP [17–20], formulated with discrete-time dynamics. Therefore, the dynamics and the cost function are discretized using a small sampling time Δt,

$$x_{k+1} = f_{v_k}(x_k) + g_{v_k}(x_k)u_k, \quad k \in \mathbb{K}, \ v_k \in \mathbb{V}, \tag{3}$$

$$J = \psi(x_N) + \sum_{k=0}^{N-1} (Q(x_k) + u_k^T R u_k) \tag{4}$$

where $N = (t_f - t_0)/\Delta t$, $x_k = x(k\Delta t + t_0)$, $u_k = u(k\Delta t + t_0)$, and $v_k = v(k\Delta t + t_0)$. Subscript k denotes the discrete time index and $\mathbb{K} \equiv \{0, 1, \ldots, N-1\}$. If Euler integration is used for discretization, one has $f_v(x) \equiv x + \Delta t \bar{f}_v(x)$, $g_v(x) \equiv \Delta t \bar{g}_v(x)$, $Q(x) \equiv \Delta t \bar{Q}(x)$, and $R \equiv \Delta t \bar{R}$.

Defining the *cost-to-go* as the incurred cost from current time k and state x_k to the final time N, denoted by $J_k(x_k)$, one has:

$$J_k(x_k) = \psi(x_N) + \sum_{j=k}^{N-1} (Q(x_j) + u_j^T R u_j). \tag{5}$$

From the form of the cost function, it directly follows:

$$J_N(x_N) = \psi(x_N),$$
$$J_k(x_k) = Q(x_k) + u_k^T R u_k + J_{k+1}(x_{k+1}), \quad \forall k \in \mathbb{K}. \tag{6}$$