Near-Optimal Control of Nonlinear Switched Systems with Non-Cooperative Switching Rules

Jihene Ben Rejeb, Lucian Buşoniu, Irinel-Constantin Morărescu, Jamal Daafouz

Abstract— **This paper presents a predictive, planning algorithm for nonlinear switched systems where there are two switching signals, one controlled and the other uncontrolled, both subject to constraints on the dwell time after a switch. The algorithm solves a minimax problem where the controlled signal is chosen to optimize a discounted sum of rewards, while taking into account the worst possible uncontrolled switches. It is an extension of a classical minimax search method, so we call it optimistic minimax search with dwell time constraints, OMSδ. For any combination of dwell times, OMSδ returns a sequence of switches that is provably near-optimal, and can be applied in receding horizon for closed loop control. For the case when the two dwell times are the same, we provide a convergence rate to the minimax optimum as a function of the computation invested, modulated by a measure of problem complexity. We show how the framework can be used to model switched systems with time delays on the control channel, and provide an illustrative simulation for such a system with nonlinear modes.**

I. Introduction

Switched systems toggle their dynamics among those in a set of linear or nonlinear modes, according to controlled or uncontrolled switching rules [13]. They model real-world systems subject to known or unknown abrupt parameter changes, e.g. in the automotive, aerospace, and energy management industries. Switched systems are therefore heavily studied, with a main research focus placed on stability and stabilization [21], [14], while work has also been done in performance optimization [1], [24]. Here, we focus on performance optimization for a class of switched systems where there are two different switching signals: one controlled and another uncontrolled. Such systems may be used to model important practical situations in e.g. smart grids [20], wireless networks [23], or networked control systems, as we illustrate in this paper. However, they have only recently started to be considered in the literature, e.g. by [2] where they are called dual switched systems.

We aim to optimize the controlled switching signal so that a discounted sum of rewards (negative costs) is maximized, subject to taking into account the worst-case values of the uncontrolled switching signal. Time is discrete, while both the controlled and uncontrolled switches may be subject to dwell time constraints, so that after a switch they must be kept constant for at least an imposed number of steps. The modes can have arbitrary nonlinear dynamics, while the rewards must be bounded. This is a minimax problem, which we solve by extending the approach from [5], called optimistic minimax search (OMS). OMS explores a tree representation of the possible sequences of max and min agent actions (here, mode switches); it is a variant of B* [17] and related to other classical minimax search methods [10], [18], [12]. It returns a near-optimal sequence with respect to the minimax-optimal value.

To extend OMS to the dual switching problem, the dwell-time conditions must be imposed, by ensuring that sequences that switched too recently keep their action constant. This is easy to implement for any combination of max and min dwell times, obtaining a variant that we call OMSδ, but the impact on the analysis turns out to be nontrivial. In particular, while the algorithm produces an a posteriori near-optimality bound as easily as OMS, obtaining an a priori convergence rate is more challenging, because the structure of the tree obtained after eliminating nodes that violate the dwell time condition is very intricate. We provide a convergence rate in the case where the dwell-time limits of the two signals are the same, equal to δ; the complexity of the algorithm in this case is exponential in the depth reached (horizon) divided by δ, compared to the original OMS where it was exponential in just the depth, and thus larger in general.

OMSδ is to our knowledge the first algorithm for optimal control in dual switched systems with nonlinear modes; the earlier work by [2] was for linear modes and focused on stability. Here we focus instead on near-optimality guarantees, since stability is a separate, difficult problem for the discounted costs that we use [19]. Our work also bears a relation to robust control in switched systems [7].

Note that due to its origins in minimax search for games, OMSδ natively handles problems where the max and min switches are applied in turn, so the min signal is considered to be generated by a smart agent that already knows the max action chosen. Nevertheless, we show how to model in this framework problems in which the max and min switches are generated simultaneously, with some conservativeness since the extra knowledge is in fact not available to the min agent in this setting. Finally, we show how to use the min action to model a time delay on the communication channel between the controller and the actuator, see also [6]; and provide illustrative numerical results in such a problem with nonlinear modes.

In the context of artificial intelligence and optimistic planning [16], [11], [9], [22], [15], the closest algorithm is again OMS [5], compared to which the main novelty here is the convergence analysis under dwell-time constraints, leading to a new complexity measure. The planning method

J. Ben Rejeb, I.C. Morărescu, and J. Daafouz are with Université de Lorraine, CRAN, UMR 7039 and CNRS, CRAN, UMR 7039, 2 Avenue de la Forêt de Haye, Vandœuvre-lès-Nancy, France. L. Buşoniu is with the Automation Department, Technical University of Cluj-Napoca, Romania. Contact: jihene.ben-rejeb@univ-lorraine.fr. This work was supported by the Agence Universitaire de la Francophonie (AUF) and the Romanian Institute for Atomic Physics (IFA), under the AUF-RO project NETASSIST; and by the PICS project No 6614 “Artificial-Intelligence-Based Optimization for the Control of Networked and Hybrid Systems”.

for max-only switched systems from our work [3] leads to a similar complexity measure and reduction compared to the no-dwell-time case, but there the analysis is much easier due to the lack of the min agent.

Next, Section II introduces our formal framework, Section III gives the algorithm, Section IV provides its analysis, Section V gives an application to systems with a delayed switching signal, and Section VI concludes.

II. Problem definition

Consider an adversarial switched problem where a controlled, maximizer switching signal affects the system together with an uncontrolled, minimizer switching signal. The max and min actions (mode switches) are respectively denoted u and w, and belong to sets U and W. These sets contain N_u and N_w elements respectively, so that there are a total of $N_u \cdot N_w$ modes. A generic action is denoted $z \in Z := U \cup W$, and can be either a max or min action.[1] In general, max and min mode changes are applied in turn, so that each step h is alternately either a max or a min decision step, which can be differentiated by checking if $z_h \in U$ or $\in W$ (if the two sets are not disjoint, special markers can be added). We will show in Example 1 how simultaneous decisions can be handled. For many switched systems it is important to ensure a minimum amount of time during which the mode remains constant, e.g. to guarantee fundamental stability or performance properties, to obey actuation constraints, etc. Therefore, each switching signal u and w may be required to obey a minimum dwell-time limit $\overline{\delta}_u$ and $\overline{\delta}_w$, respectively. E.g. for the max agent the dwell-time is defined as the number of max decision steps during which the action/mode u remains constant after a change, and the condition requires that all dwell times along the sequence are at least as large as $\overline{\delta}_u$. The situation is similar for the min actions w. Note that taking a limit equal to 1 is equivalent to not imposing a dwell-time condition for that signal.

Denote an infinite sequence of actions by $\mathbf{z}_\infty = (z_0, z_1, z_2, z_3, \dots, z_{2k}, z_{2k+1}, \dots) = (u_0, w_0, u_1, w_1, \dots, u_k, w_k, \dots) \in Z_{\overline{\delta}_u, \overline{\delta}_w} \subset (U \times W)^\infty$ where $Z_{\overline{\delta}_u, \overline{\delta}_w}$ is the set of sequences that satisfy the two dwell-time conditions. Here h counts all decision steps; while step k only increases with *pairs* of max-min decisions. Note that by definition, dwell times only increase once every two steps h (corresponding to one step k). A finite sequence of h actions is denoted $\mathbf{z}_h = (z_0, z_1, \dots, z_{h-1})$, with $\mathbf{z}_0$ the empty sequence by convention. The truncation of $\mathbf{z}_\infty$ to h initial elements is denoted $\mathbf{z}_{\infty|h}$. An example of switching minimax actions is given in Figure 1.

At each step $h \in \mathbb{N}$, the system evolves as follows:

$$x_{h+1} = f(x_h, z_h) \tag{1}$$

where $x_h \in X$ is the state, $z_h \in Z$ is the max or min action, and $f : X \times Z \to X$ are the mode dynamics. A reward (negative cost) $\rho(x_h, z_h)$ is assigned, where $\rho : X \times Z \to \mathbb{R}$.

[1] Notations u and w are used when the max and min actions are regarded separately; otherwise, we use generic notation z.

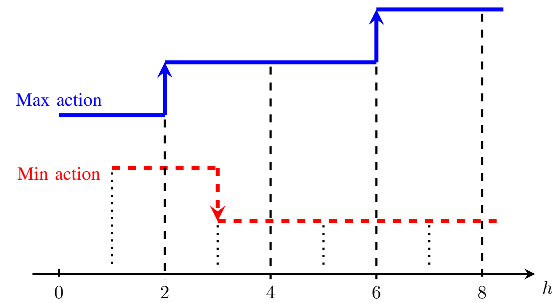

Fig. 1: Illustration of a minimax sequence developed when $\overline{\delta}_u = \overline{\delta}_w = 2$. The applied actions are shown by a blue continuous line and a red dashed line for max and min actions respectively. Note that initially the dwell-time condition is assumed satisfied for both agents.

Then, the overall infinite-horizon value of sequence $\mathbf{z}_\infty$ is:

$$v(\mathbf{z}_\infty) := \sum_{h=0}^{\infty} \gamma^h \rho(x_h, z_h) \tag{2}$$

where $\gamma \in [0, 1]$ is the discount factor. The goal is to find the minimax-optimal value, defined as:

$$v^* := \lim_{k \to \infty} \left[\max_{u_0 \in U(\mathbf{z}_0)} \min_{w_0 \in W(\mathbf{z}_1)} \cdots \cdots \max_{u_k \in U(\mathbf{z}_{2k})} \min_{w_k \in W(\mathbf{z}_{2k+1})} \sum_{h=0}^{2k} \gamma^h \rho(x_h, z_h) \right] \tag{3}$$

when this limit exists. Here, $U(\mathbf{z}_h)$ and $W(\mathbf{z}_h)$ respectively denote the set of all max and min actions at depth h that satisfy the dwell-time constraints given prior actions $\mathbf{z}_h$. E.g., $U(\mathbf{z}_h) = U$ when $\mathbf{z}_h$ already satisfies the max dwell time condition at h, and otherwise $U(\mathbf{z}_h)$ is equal to the last max action along sequence $\mathbf{z}_h$.

Assumption 1: The rewards $\rho(x, z)$ are in $[0, 1]$ for all $x \in X, z \in Z$.

This boundedness assumption means that (2) is in $[0, \frac{1}{1-\gamma}]$ for any sequence. It also helps to define, for any finite sequence $\mathbf{z}_h$, lower and upper bounds on the values of all sequences $\mathbf{z}_\infty$ starting with $\mathbf{z}_h$, which are essential in developing our algorithm later:

$$l(\mathbf{z}_h) := \sum_{j=0}^{h-1} \gamma^j \rho(x_j, z_j), \quad b(\mathbf{z}_h) := l(\mathbf{z}_h) + \frac{\gamma^h}{1-\gamma} \tag{4}$$

with the convention that an empty sum is 0. Thus, $v(\mathbf{z}_\infty) \in [l(\mathbf{z}_h), b(\mathbf{z}_h)]$. Let $\delta(h) = \frac{\gamma^h}{1-\gamma}$ denote the *gap* between the two bounds, an uncertainty on the values of sequences $\mathbf{z}_\infty$ starting with $\mathbf{z}_h$.

Next, we show how to represent problems in which max and min mode changes are applied simultaneously.

Example 1: Simultaneous min-max switching. Define the dynamics $y_{k+1} = g(y_k, u_k, w_k)$ and the rewards $r_{k+1} = r(y_k, u_k, w_k)$, with $y_k \in Y$, for a problem where max and min decisions u and w are simultaneous. The infinite-horizon value to optimize is $\sum_{k=0}^{\infty} \beta^k r(y_k, u_k, w_k)$. To represent this problem in the turn-based formalism (1)-(3), we introduce an augmented state vector $x_h = [x_{1,h}^\top, x_{2,h}]^\top \in Y \times \{U \cup \{s\}\}$.

The first element of this vector is always the current state of the system. The second element takes special value $s \notin U$ to signify max decision steps, and at min steps it remembers the latest max decision. Formally, at steps $h = 2k$, $x_h = [y_k^\top, s]^\top$, while at $h = 2k+1$, $x_h = [y_k^\top, u_k]^\top$. Using this augmented state, the simultaneous dynamics g are represented by the following turn-based dynamics f in (1):

$$f(x_h, z_h) = \begin{cases} [x_{1,h}^\top, z_h]^\top = [y_k^\top, u_k]^\top & \text{if } x_{2,h} = s \\ [g^\top(x_{1,h}, x_{2,h}, z_h), s]^\top & \\ \quad = [g^\top(y_k, u_k, w_k), s]^\top & \text{otherwise} \end{cases}$$

where $k = \lfloor h/2 \rfloor$ (floor). Rewards are similarly represented:

$$\rho(x_h, z_h) = \begin{cases} 0, & \text{if } x_{2,h} = s \\ r(x_{1,h}, x_{2,h}, z_h) = r(y_k, u_k, w_k) & \text{otherwise} \end{cases} \tag{5}$$

We have $\sum_{h=0}^{\infty} \gamma^h \rho(x_h, z_h) = \gamma \sum_{k=0}^{\infty} \gamma^{2k} r(y_k, u_k, w_k)$, so to optimize the intended objective function with discount factor β, it suffices to take $\gamma = \sqrt{\beta}$. In closing, recall that this turn-based representation is conservative since it assumes the min agent knows, and can react to, the max action, even though in fact it does not due to the simultaneous actions. □

III. Algorithm

Optimistic minimax search with dwell-time constraints (OMSδ) explores a tree representation of the possible action sequences. It starts with a root node corresponding to the empty sequence, and iteratively expands n nodes taking into account dwell-time conditions. Figure 2 illustrates, with squares representing max decision nodes, and disks min decision nodes. Each node is labeled by two dwell times, for max and min decisions, separated by slashes in the figure. Note that by convention both dwell time conditions are taken satisfied at $h = 0$, so e.g. the root node in the figure has dwell times $\overline{\delta}_u / \overline{\delta}_w$, namely 2/2. A max decision node is expanded by adding children corresponding to max actions, and similarly for min decision nodes. Each arc is labeled by the action taken at the parent node to reach the child. Specifically, at max nodes, if the max dwell-time is at least $\overline{\delta}_u$ then N_u children nodes are created, one for every max action; otherwise, i.e. if the max dwell-time condition is not satisfied, only the child that keeps the action constant is added. Similarly, N_w children nodes are added at a min node if its min dwell-time is at least $\overline{\delta}_w$, and only the constant-action child is added otherwise. For example, the node labeled 1/3 in the figure has max dwell time 1 because the max action taken to reach it, 'b', is different from the previous max action 'a' taken two levels higher (at the root), so a max switch just occurred. These two different actions are highlighted by gray arcs. Note that this particular node is not immediately affected by the non-satisfaction of the max dwell time, since it is a min decision node; indeed, both its children are created since the min dwell time is still 3, and the constraint only has an effect at the *next* depth, where the only allowed max action is 'b'. Figure 2 also illustrates some constrained min node expansions, from depth 5 to 6.

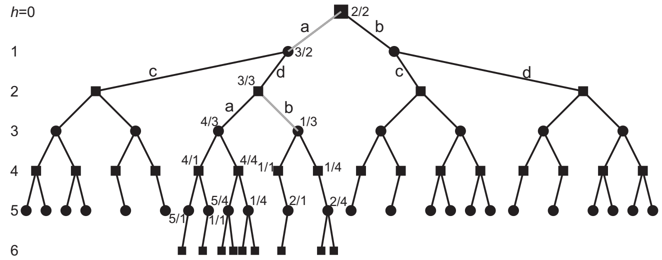

Fig. 2: Illustration of a minimax tree developed by the algorithm. The max agent has modes a and b, while the min agent has modes c and d, so that $N_u = N_w = 2$. The dwell time limits are taken $\overline{\delta}_u = \overline{\delta}_w = 2$.

Each node at some depth h is reached via a unique path through the tree, and so is uniquely associated to the sequence of actions $\mathbf{z}_h$ on this path. We denote by $\delta_u(\mathbf{z}_h)$ and $\delta_w(\mathbf{z}_h)$ the current max and min dwell-times at depth h. We will work interchangeably with sequences and nodes.

Let $\mathcal{T}$ denote the current tree, $\mathcal{L}(\mathcal{T})$ the leaf nodes of this tree, and $\mathcal{C}(\mathbf{z})$ the children of node $\mathbf{z}$, all satisfying the dwell-time constraints. The algorithm computes lower and upper bounds $L(\mathbf{z})$ and $B(\mathbf{z})$ for each node. They are initialized at the leaves using l and b and propagated upwards in the tree:

$$L(\mathbf{z}) = \begin{cases} l(\mathbf{z}), & \text{if } \mathbf{z} \in \mathcal{L}(\mathcal{T}) \\ \max_{\mathbf{z}' \in \mathcal{C}(\mathbf{z})} L(\mathbf{z}'), & \text{if } \mathbf{z} \text{ max node, } \mathbf{z} \notin \mathcal{L}(\mathcal{T}) \\ \min_{\mathbf{z}' \in \mathcal{C}(\mathbf{z})} L(\mathbf{z}'), & \text{if } \mathbf{z} \text{ min node, } \mathbf{z} \notin \mathcal{L}(\mathcal{T}) \end{cases}$$

$$B(\mathbf{z}) = \begin{cases} b(\mathbf{z}), & \text{if } \mathbf{z} \in \mathcal{L}(\mathcal{T}) \\ \max_{\mathbf{z}' \in \mathcal{C}(\mathbf{z})} B(\mathbf{z}'), & \text{if } \mathbf{z} \text{ max node, } \mathbf{z} \notin \mathcal{L}(\mathcal{T}) \\ \min_{\mathbf{z}' \in \mathcal{C}(\mathbf{z})} B(\mathbf{z}'), & \text{if } \mathbf{z} \text{ min node, } \mathbf{z} \notin \mathcal{L}(\mathcal{T}) \end{cases} \tag{6}$$

At each iteration, to choose the next leaf to expand, OMSδ starts from the root and constructs a path by recursively selecting an optimistic child for the agent at the current node, in the same way as OMS in [5]: a child with the largest upper bound at max nodes, and one with the smallest lower bound at min nodes. The main difference between OMSδ and OMS is in the expansion of this leaf, which in OMSδ is constrained to only create the children that obey the dwell time conditions, as explained above. After n node expansions, the algorithm stops and returns, like OMS, the sequence $\hat{\mathbf{z}}$ and bounds of the deepest node expanded. Algorithm 1 summarizes OMSδ, where $(\cdot, \cdot)$ denotes the concatenation of two sequences and $h(\cdot)$ yields the depth of a sequence.

OMSδ will typically be used to find max decisions to apply. The algorithm should then be applied in receding horizon, calling it with the current state at max decision steps where the dwell time condition is satisfied. If it is not, then the max action must be kept constant anyway so it is not useful to run OMSδ. To exemplify, assume that the minimax switching sequence in Figure 1 is obtained by such a closed-loop application of OMSδ. The algorithm is first called at $h = k = 0$, resulting in the first max action (mode), which is applied and the min agent generates its own mode in an unknown way. OMSδ is next called at $h = 2$, corresponding to $k = 1$, at which time it generates a different mode, and

Algorithm 1 OMS with dwell-time constraints (OMSδ)

Input: budget n

initialize: $\mathcal{T} \leftarrow \{\mathbf{z}_0\}$, the root
for iteration $t = 1$ to n **do**
  $\mathbf{z} \leftarrow \mathbf{z}_0$
  while $\mathbf{z} \notin \mathcal{L}(\mathcal{T})$ **do**
    $\mathbf{z} \leftarrow \begin{cases} \arg\max_{\mathbf{z}' \in \mathcal{C}(\mathbf{z})} B(\mathbf{z}'), & \text{if } \mathbf{z} \text{ max node} \\ \arg\min_{\mathbf{z}' \in \mathcal{C}(\mathbf{z})} L(\mathbf{z}'), & \text{if } \mathbf{z} \text{ min node} \end{cases}$
  end while
  $\mathbf{z}(t) \leftarrow \mathbf{z}$
  expand $\mathbf{z}(t)$, by adding its children to $\mathcal{T}$:
  if $\mathbf{z}(t)$ max node **then**
    if $\delta_u(\mathbf{z}(t)) \geq \overline{\delta}_u$, add children $(\mathbf{z}(t), u)\, \forall u \in U$
    else, add the single child that keeps u constant
  else
    if $\delta_w(\mathbf{z}(t)) \geq \overline{\delta}_w$, add children $(\mathbf{z}(t), w)\, \forall w \in W$
    else, add the single child that keeps w constant
  end if
  compute bounds for all $\mathbf{z} \in \mathcal{T}$ with (6)
end for

Output: $\hat{\mathbf{z}} := \arg\max_{\mathbf{z}(t), t=1,\ldots,n} h(\mathbf{z})$, $l(\hat{\mathbf{z}})$, $b(\hat{\mathbf{z}})$

again the min agent responds. Now, since a max switch has occurred, the max action must be kept constant for the next step too, and OMSδ is only called again at $h = 6$, or $k = 3$, and so on. Note that min switches also satisfy their own dwell time, and OMSδ takes advantage of this information.

IV. Analysis

We extend the analysis of OMS in [5] to OMSδ. The first part of our analysis establishes basic properties of the minimax algorithm that still hold under the additional dwell-time constraints. The second part gives our main novel results: a complexity measure of the problem and a corresponding convergence rate of OMSδ. Due to space limits we skip all proofs except that of the main result, but where applicable we point out relations to [5].

Lemma 2: At any iteration t, for any nodes $\mathbf{z}$, $\mathbf{z}' \in \mathcal{C}(\mathbf{z})$ on the optimistic path, $[L(\mathbf{z}), B(\mathbf{z})] \subseteq [L(\mathbf{z}'), B(\mathbf{z}')]$.

This is a direct extension of Lemma 5 in [5]. Define now for any node $\mathbf{z}_h$ of finite depth h the minimax value $v(\mathbf{z}_h)$ among infinite sequences starting with $\mathbf{z}_h$. Formally:

$$v(\mathbf{z}_h) = \sum_{j=0}^{h-1} \gamma^j \rho(x_j, z_j) + \begin{cases} \max\limits_{z_h \in U(\mathbf{z}_h)} \min\limits_{z_{h+1} \in W(\mathbf{z}_{h+1})} \cdots \sum\limits_{j=h}^{\infty} \gamma^j \rho(x_h, z_h), \text{if } \mathbf{z}_h \text{ max} \\ \min\limits_{z_h \in W(\mathbf{z}_h)} \max\limits_{z_{h+1} \in U(\mathbf{z}_{h+1})} \cdots \sum\limits_{j=h}^{\infty} \gamma^j \rho(x_h, z_h), \text{if } \mathbf{z}_h \text{ min} \end{cases} \tag{7}$$

Recall that $U(\mathbf{z}_h)$ and $W(\mathbf{z}_h)$ denote the sets of allowed max or min actions following sequence $\mathbf{z}_h$ that satisfy the dwell-time constraints.

Next we characterize the subset of nodes that the algorithm will expand, which is in general smaller than the full tree. This result is a nontrivial adaptation of Lemma 3 from [5] to the dwell-time case.

Lemma 3: At depth h in the tree, OMSδ only expands nodes in the set:

$$\mathcal{T}_h^* := \big\{\mathbf{z}_h \,\big|\, |v^* - v(\mathbf{z}_p)| \leq \delta(h), \quad \forall \mathbf{z}_p \text{ on path from root to } \mathbf{z}_h\big\} \tag{8}$$

The following result, corresponding to Theorem 6 in [5], gives an *a posteriori* near-optimality bound, which can be directly evaluated once the algorithm has stopped.

Theorem 4: Let h^* be the largest depth of any expanded node. Then, $|v^* - v(\hat{\mathbf{z}})| \leq \delta(h^*)$ and $v^* \in [L(\mathbf{z}_0), B(\mathbf{z}_0)]$.

The results presented so far, in this first part of the analysis, are extensions of those for OMS in [5]. The goal of the second part is to provide an *a priori* near-optimality bound, and this will differ significantly from [5] because the size of the expanded subtree $\mathcal{T}^* = \bigcup_{h \geq 0} \mathcal{T}_h^*$ must be characterized, and this tree has a very complicated structure due to the elimination of sequences that violate the dwell-time conditions. Another essential remark about the results up to now is that they hold in general, for any dwell time conditions. For the same reason of tree complexity, to make the subsequent convergence rate analysis feasible we must impose the following, admittedly conservative, condition.

Assumption 5: The max and min switching signals have equal dwell-times, $\overline{\delta}_u = \overline{\delta}_w =: \delta$.

We believe similar convergence rates apply when this assumption is not satisfied, but we leave this extension for future work. Denote also $q = \max(N_u, N_w)$. Thus, both max and min nodes check the same dwell time limit, and create at most q children nodes. We define next a complexity measure that characterizes the rate of growth of $\mathcal{T}^*$ with the depth.

Definition 6: Let κ be the smallest positive number so that $\exists C > 0, |\mathcal{T}_h^*| \leq C\kappa^{h/\delta}, \forall h \geq 0$, where $|\cdot|$ denotes set cardinality.

The value of κ quantifies the complexity of the search problem: the larger κ is, the more difficult the problem. The following two interesting special cases show that κ always exists in the interval $[1, \delta q]$.

Case 1. All sequences optimal: Consider the problem where all the rewards are identical, equal to 1. Any sequence is optimal in this case, and the algorithm must explore the entire tree uniformly, so $\mathcal{T}_h^*$ contains all the nodes at h. It can be shown that the number of these nodes is upper-bounded by $\delta^3 q(q-1)(\delta q)^{\frac{h}{\delta}}$, so $\kappa = \delta q$. Since $\mathcal{T}_h^*$ is the largest possible in this case, this value is also the largest for κ. □

Case 2: One optimal sequence: Consider a problem where $|\mathcal{T}^*|$ has a single path that satisfies the dwell-time constraints and is minimax optimal. At each max node along this path, one child satisfying the max dwell time has reward 1 and all other children have reward 0. The situation is reversed at min nodes. Figure 3 illustrates a tree with one such optimal path, highlighted by the thick lines (dwell time constraints are ignored for clarity). It can be shown that, with or without dwell time constraints, the number of nodes expanded is at most a constant C at each depth, i.e. $|\mathcal{T}_h^*| \leq C$ and $\kappa = 1$. Since there must always be at least one node in $\mathcal{T}_h^*$, this value of κ is also the smallest possible. □

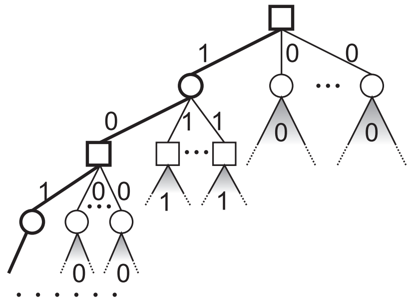

Fig. 3: Illustration of a tree with $\kappa = 1$. Rewards are shown along the transitions. Figure adapted from [5].

We are finally ready to give our main *a priori* result.

Theorem 7: Given budget n, we have:

$$|v^* - v(\hat{\mathbf{z}})| \leq \delta(h^*) \leq \begin{cases} O(n^{-\delta \frac{\log 1/\gamma}{\log \kappa}}) & \text{if } \kappa > 1 \\ O(\gamma^{n/C}) & \text{if } \kappa = 1 \end{cases} \tag{9}$$

where C is the constant from the definition of κ and γ is the discount factor.

Proof: The first inequality is due to Theorem 4, so we prove the second one. Define $h(n)$ to be the smallest depth so that $n \leq \sum_{j=0}^{h(n)} |\mathcal{T}_h^*|$; this means the algorithm has expanded nodes at $h(n)$ (perhaps not yet at $h(n)+1$) so, $h^* \geq h(n)$. Since sequence $\delta(h)$ is decreasing, one has $\delta(h^*) \leq \delta(h(n))$.

Let $\kappa > 1$, then $n \leq \sum_{j=0}^{h(n)} C\kappa^{j/\delta}$, which yields $n \leq C\frac{\kappa^{(h(n)+1)/\delta}-1}{\kappa^{1/\delta}-1}$. After some derivations, $h(n) \geq \delta \frac{\log n}{\log \kappa} - c_1$. Thus, $\delta(h(n)) \leq c_2 n^{-\delta \frac{\log 1/\gamma}{\log \kappa}}$. Here, c_1, c_2 denote unknown positive constants.

If $\kappa = 1$, then $n \leq \sum_{j=0}^{h(n)} C = C(h(n)+1)$, and $h(n) \geq \frac{n}{C} - 1$ so $\delta(h(n)) \leq \frac{\gamma^{\frac{n}{C}-1}}{1-\gamma}$. The theorem is proven. ■

Therefore, when κ is smaller (the problem is simpler), the algorithm converges faster with n, since the negative exponent of n is larger in magnitude. Furthermore, stronger dwell time conditions, represented by larger δ, directly increase this magnitude, so the algorithm is also faster when the dwell time limits are larger, which makes sense since there are fewer solutions to consider. When $\kappa = 1$ (the simplest possible type of problem), convergence is exponential in n.

V. Application to switched systems with delays

We provide a numerical illustration of the OMSδ algorithm for problems with communication delays on the control channel. This is relevant in networked control systems where the controller is connected to the actuator by a communication network. Inspired by [8], we model this time-varying delay as an uncontrolled switch, represented in our framework by a min agent. Specifically, consider the system:

$$\tilde{y}_{k+1} = \tilde{g}(\tilde{y}_k, u_{k-w_k}), \qquad \forall k > 0 \tag{10}$$

where $\tilde{y}_k \in \mathbb{R}^n$ represents the system state at time $k \in \mathbb{Z}^+$, u_{k-w_k} is a controlled, but delayed switching signal, and w_k is the delay at step k, which takes integer values in $\{0, 1, \ldots, m\}, m \geq 0$. A reward function $\tilde{r}(\tilde{y}_k, u_{k-w_k})$ is used that takes values in $[0, 1]$; note that the reward uses the delayed input, which means that it is generated at the system side. The delay is taken to satisfy a min dwell-time condition such that its value should be maintained for $\overline{\delta}_w$ steps after a change. In other words, if $w_{k+1} \neq w_k$ then $w_{k+1} = w_{k+2} = \ldots = w_{k+\overline{\delta}_w}$. Such a condition arises e.g. when the time delays have bounded rates of change, which is often assumed. The switching signal generated u is itself constrained to have a dwell-time of at least $\overline{\delta}_u$ (although note that it cannot be guaranteed that the signal obtained *after* the application of the time delay will still satisfy this condition).

The goal is to optimize the controller decisions u so as to maximize the discounted sum of rewards, while taking into account the worst possible delays w. To this end, we will transform the problem in the minimax form of Example 1, by defining dynamics $g(y, u, w)$ and rewards $r(y, u, w)$ that work with an augmented state vector y. This vector is, at step k:

$$y_k = [y_k^0, y_k^1, y_k^2, \cdots, y_k^m]^\top := [\tilde{y}_k, u_{k-1}, u_{k-2}, \cdots, u_{k-m}]^\top$$

Then, the augmented dynamics that represent (10) are:

$$\begin{aligned} y_{k+1} = g(y_k, u_k, w_k) &= [\tilde{y}_{k+1}, u_k, u_{k-1}, \cdots, u_{k-m+1}]^\top \\ &= [\tilde{y}_{k+1}, u_k, y_k^1, \cdots, y_k^{m-1}]^\top \end{aligned}$$

where the underlying state $\tilde{y}_{k+1}$ is computed as follows:

$$\tilde{y}_{k+1} = \begin{cases} \tilde{g}(\tilde{y}_k, u_k) = \tilde{g}(y_k^0, u_k) & \text{if } w_k = 0 \\ \tilde{g}(\tilde{y}_k, u_{k-w_k}) = \tilde{g}(y_k^0, y_k^{w_k}) & \text{if } w_k > 0 \end{cases}$$

The augmented reward function that represents $\tilde{r}$ is:

$$r(y_k, u_k, w_k) := \begin{cases} \tilde{r}(\tilde{y}_k, u_k) = \tilde{r}(y_k^0, u_k) & \text{if } w_k = 0 \\ \tilde{r}(\tilde{y}_k, u_{k-w_k}) = \tilde{r}(y_k^0, y_k^{w_k}) & \text{if } w_k > 0 \end{cases}$$

By further transforming this problem into the form (1), (2) as in Example 1, we can then apply OMSδ in closed loop, receding horizon to produce a switching signal u_k. Recall that once a switch has occurred, u_k is simply held constant for $\overline{\delta}_u$ steps before calling OMSδ again. To our knowledge, no other existing technique can handle this type of switched minimax control problem with dwell-time constraints.

Our framework is general enough to allow any nonlinear modes in dynamics $\tilde{g}$. Next, we illustrate it in a simulation of an inverted pendulum driven by a DC motor, with two states: angle α and angular velocity $\dot{\alpha}$. The delay w_k is generated uniformly randomly in the set $\{0, 1\}$ (so it is at most one step long), at all steps where it satisfies a minimum dwell-time of $\overline{\delta}_w = 2$; at other steps it is kept constant. The continuous-time dynamics are given e.g. in [4], and are discretized via numerical integration with $T_\text{s} = 0.05\,\text{s}$ to obtain $\tilde{g}$. The goal is to stabilize the mass pointing upwards (corresponding to $\alpha = 0$), and the maximum voltage is $3\,\text{V}$, which from some initial states is insufficient to bring the mass up in one go; instead it must be swung back and forth to accumulate energy before being stabilized. To perform these swing-ups, the control therefore requires large planning horizons, as well as fast actions, so adding time delays makes the problem very challenging. The reward is taken quadratic, $-(5\alpha^2 + 0.1\dot{\alpha}^2 + u^2)$, and normalized to $[0, 1]$ using the state bounds $\alpha \in [-\pi, \pi]\,\text{rad}$, $\dot{\alpha} \in [-15\pi, 15\pi]\,\text{rad/s}$; we also take $\gamma = \sqrt{0.95}$. Noted that the implementation computes tighter bounds than the general formulas (4), by exploiting the fact that rewards are 0 at max steps, see (5).

As before, the modes u represent voltage levels: $-3, 0$, or $3\,\mathrm{V}$. No dwell time is imposed on u. Figure 4 shows typical results for budget $n = 3000$. The pendulum is stabilized, although it requires two swings, whereas without delay it would only require 1. Sometimes the controller 'loses' the pendulum and must re-swing it. This happens because nonzero actions must be applied to keep the pendulum around the unstable equilibrium, and depending on the delays these actions may sometime fail and the pendulum falls. Thus, even with $m = 1$ the problem is already very difficult; indeed we increased m to 2 and in that case the pendulum can only rarely be stabilized for longer periods.

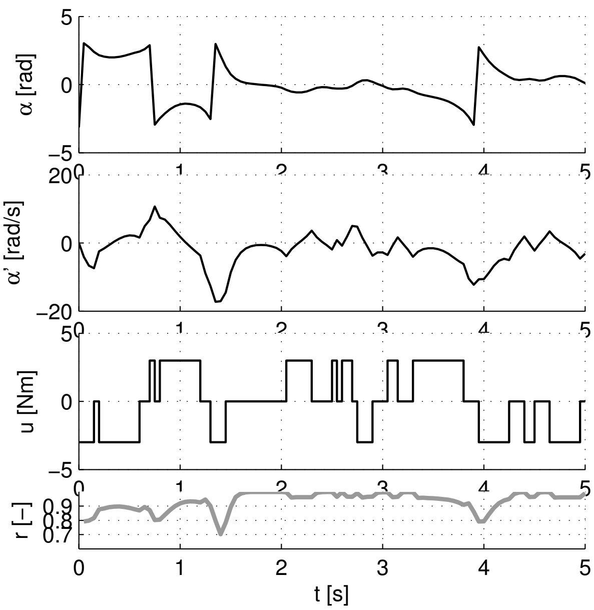

Fig. 4: Inverted pendulum trajectory.

VI. Conclusions

The paper introduced OMSδ, an optimistic minimax search algorithm for a dual switched problem where maximizer and minimizer switching signals must obey dwell-time conditions. We showed that the algorithm converges towards the optimal value, and provided a convergence rate with respect to the computational budget when the two dwell time limits are the same. The framework was used to model switched systems with time delays on the control channel, and illustrated in a simulation of such a system with nonlinear modes. An interesting future direction is to extend the convergence rate analysis by removing the equality condition on the dwell time constraints.

References

[1] U. Ali and M. Egerstedt, "Optimal control of switched dynamical systems under dwell time constraints," in *53rd IEEE Conference on Decision and Control*, Dec 2014, pp. 4673–4678.

[2] P. Bolzern, P. Colaneri, and G. D. Nicolao, "Design of stabilizing strategies for dual switching stochastic-deterministic linear systems," in *Proceedings 19th IFAC World Congress*, Cape Town, South Africa, 24–29 August 2014, pp. 4080–4084.

[3] L. Buşoniu, M.-C. Bragagnolo, J. Daafouz, and C. Morarescu, "Planning methods for the optimal control and performance certification of general nonlinear switched systems," in *Proceedings 54th IEEE Conference on Decision and Control (CDC-15)*, Osaka, Japan, 15–18 December 2015, pp. 3604–3609.

[4] L. Buşoniu, R. Munos, and R. Babuška, "A review of optimistic planning in Markov decision processes," in *Reinforcement Learning and Adaptive Dynamic Programming for Feedback Control*, F. Lewis and D. Liu, Eds. Wiley, 2012.

[5] L. Buşoniu, E. Páll, and R. Munos, "An analysis of optimistic, best-first search for minimax sequential decision making," in *2014 IEEE International Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL-14)*, Orlando, 10–12 December 2014.

[6] A. Cicone, A. D'Innocenzo, N. Guglielmi, and L. Laglia, "A suboptimal solution for optimal control of linear systems with unmeasurable switching delays," in *2015 54th IEEE Conference on Decision and Control (CDC)*, Dec 2015, pp. 2933–2938.

[7] P. Hauroigne, P. Riedinger, and C. Iung, "Switched affine systems using sampled-data controllers: Robust and guaranteed stabilization," *IEEE Transactions on Automatic Control*, vol. 56, no. 12, pp. 2929–2935, Dec 2011.

[8] L. Hetel, J. Daafouz, and C. Iung, "Equivalence between the lyapunov-krasovskii functional approach for discrete delay systems and the stability conditions for switched systems," *Nonlinear Analysis: Hybrid Systems*, vol. 2, no. 3, pp. 697–705, 2008.

[9] J.-F. Hren and R. Munos, "Optimistic planning of deterministic systems," in *Proceedings 8th European Workshop on Reinforcement Learning (EWRL-08)*, Villeneuve d'Ascq, France, 30 June – 3 July 2008, pp. 151–164.

[10] D. E. Knuth and R. W. Moore, "An analysis of alpha-beta pruning," *Artificial Intelligence*, vol. 6, no. 4, pp. 293–326, 1975.

[11] L. Kocsis and C. Szepesvári, "Bandit based Monte-Carlo planning," in *Proceedings 17th European Conference on Machine Learning (ECML-06)*, Berlin, Germany, 18–22 September 2006, pp. 282–293.

[12] R. E. Korf and D. M. Chickering, "Best-first minimax search," *Artificial Intelligence*, vol. 84, no. 1–2, pp. 299–337, 1996.

[13] D. Liberzon, *Switching in Systems and Control.*, ser. Systems and Control: Foundations and Applications. Birkhauser, 2003.

[14] H. Lin and P. J. Antsaklis, "Stability and stabilizability of switched linear systems: A survey of recent results," *IEEE Transactions on Automatic Control*, vol. 54, no. 2, pp. 308–322, 2009.

[15] C. Mansley, A. Weinstein, and M. L. Littman, "Sample-based planning for continuous action Markov decision processes," in *Proceedings 21st International Conference on Automated Planning and Scheduling*, Freiburg, Germany, 11–16 June 2011, pp. 335–338.

[16] R. Munos, "From bandits to Monte Carlo tree search: The optimistic principle applied to optimization and planning," *Foundations and Trends in Machine Learning*, vol. 7, no. 1, pp. 1–130, 2014.

[17] A. J. Palay, "The B* tree search algorithm – new results," *Artificial Intelligence*, vol. 19, pp. 145–163, 1982.

[18] A. Plaat, J. Schaeffer, W. Pijls, and A. de Bruin, "Best-first fixed-depth minimax algorithms," *Artificial Intelligence*, vol. 87, no. 1–2, pp. 255–293, 1996.

[19] R. Postoyan, L. Buşoniu, D. Nešić, and J. Daafouz, "Stability analysis of discrete-time infinite-horizon optimal control with discounted cost," *IEEE Transactions on Automatic Control*, 2016, in press.

[20] W. Saad, Z. Han, H. V. Poor, and T. Basar, "Game-theoretic methods for the smart grid: An overview of microgrid systems, demand-side management, and smart grid communications," *IEEE Signal Processing Magazine*, vol. 29, no. 5, pp. 86–105, 2012.

[21] R. Shorten, F. Wirth, O. Mason, K. Wulff, and C. King, "Stability criteria for switched and hybrid systems," *Automatica*, vol. 49, no. 7, pp. 545–592, 2007.

[22] T. J. Walsh, S. Goschin, and M. L. Littman, "Integrating sample-based planning and model-based reinforcement learning," in *Proceedings 24th AAAI Conference on Artificial Intelligence (AAAI-10)*, Atlanta, US, 11–15 July 2010.

[23] Y. Xing and R. Chandramouli, "Stochastic learning solution for distributed discrete power control game in wireless data networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 4, pp. 932–944, 2008.

[24] F. Zhu and P. J. Antsaklis, "Optimal control of switched hybrid systems: A brief survey," *Discrete Event Dynamic Systems*, vol. 25, no. 3, pp. 345–364, 2015.