Optimal Control Synthesis for ω-regular Objectives with Abstraction-Refinement

Yoke Peng Leong, *Student Member, IEEE,* Pavithra Prabhakar, *Member, IEEE,*

Abstract

This paper presents an abstraction-refinement based framework for optimal controller synthesis of discrete-time systems with respect to ω-regular objectives. It consists of first abstracting the discrete-time "concrete" system into a finite weighted transition system using a finite partition of the state-space. Then, a two-player mean payoff parity game is solved on the product of the abstract system and the Büchi automaton corresponding to the ω-regular objective, to obtain an optimal "abstract" controller that satisfies the ω-regular objective. The abstract controller is guaranteed to be implementable in the concrete discrete-time system, with a sub-optimal cost. The abstraction is refined with finer partitions to reduce the sub-optimality. In contrast to existing formal controller synthesis algorithms based on abstractions, apart from synthesizing a controller, this technique also provides an upper bound on the trajectory cost when implementing the suboptimal controller. In addition, under the assumption on the existence of certain robust controllers, the refinement procedure guarantees to find controllers whose costs are arbitrarily close to the optimal cost. An example is presented to illustrate the feasibility of the approach.

I. Introduction

Formal synthesis techniques that automatically generate controllers from high level specifications have gained momentum in the recent past [1], [2], [3], [4]. The high level specifications, for instance, capture mission level objectives such as surveillance, that require autonomous systems to operate over an unbounded amount of time. Such specifications are captured in logics such as Linear Temporal Logic (LTL) or using automata such as Büchi automata [2], [3], [4], [5], [6]. While algorithmic results based on automata and game theory exist for controller synthesis of finite state systems [2], abstraction based methods have gained prominence in the case of dynamical systems with a potentially infinite state space [3], [7], [8], [9], [10]. Abstraction based formal controller synthesis consists of the following steps: 1) A finite state abstraction of the system is constructed that "under-approximates" the control actions and "over-approximates" the environment actions; 2) An abstract controller for the abstract system is constructed using results from automata and game theory; 3) A concrete controller for the original system is extracted from the abstract controller. This method has been successfully applied in controller synthesis of switched systems [5], [6], and robotic path planning [11], [12].

In addition to qualitative properties such as surveillance, often optimality conditions on the costs are important for performance reasons. For instance, one would require that an autonomous system performing surveillance use

Y. P. Leong is with the Control and Dynamical Systems, California Institute of Technology, Pasadena, CA 91125, USA ypleong@caltech.edu

P. Prabhakar is with the Department of Computer Science, Kansas State University, Manhattan, KS 66506, USA pprabhakar@ksu.edu

minimum fuel in the long run or reduce the time between visits to a particular region. While formal controller synthesis with respect to qualitative objectives over infinite horizon such as those expressed in LTL or quantitative objectives such as mean payoff have independently received attention in the context of dynamical systems, a framework that considers both aspects has received relatively less attention. In this paper, we combine ideas from abstraction based controller synthesis for dynamical systems and finite quantitative games [13], [14] to obtain an abstraction refinement scheme for optimal controller synthesis of ω-regular objectives.

We start by defining pre-orders on systems which respect optimality and existence of controllers. That is, if a system $\mathcal{H}_2$ is higher up in the ordering than a system $\mathcal{H}_1$, then the existence of controller for $\mathcal{H}_2$ with respect to an infinite horizon qualitative property implies the existence of the same for $\mathcal{H}_1$, and the cost of the optimal controller for $\mathcal{H}_1$ is at most that of $\mathcal{H}_2$. In our case, $\mathcal{H}_2$ is a finite state weighted transition system, on which we solve the optimal controller synthesis problem with respect to ω-regular objective by solving a quantitative game, namely, mean payoff parity game. Here, mean payoff refers to the "average" cost and parity games encode the Büchi condition. Our pre-order and abstraction algorithm are derived in our earlier work on *bounded horizon* regular objectives [15], the same definitions and constructions preserve optimal controllers for ω-regular games. However, we need to solve more complex games on the abstract system. In addition, we show convergence to a controller with costs arbitrarily close to the optimal cost through iterative refinement, when a certain kind of robust controller exists.

In this work, we have implemented the abstraction algorithm and the algorithm for solving the mean payoff parity games; we illustrate the optimal controller synthesis algorithm on an example involving a surveillance robot that needs to avoid an obstacle and visit two check points. A bottleneck of this method is the increase in the complexity of the analysis with that in the number of regions in the partition. Currently, we consider uniform partition, however, in the future, we will investigate refinement algorithms that non-uniformly partition the state-space using ideas such as counter-example guided abstraction refinement [16]. Although the method introduced in this paper applies to the general class of discrete-time hybrid systems, the optimizations that compute the weights depend on the cost function and the dynamics.

A. Related Work and Contributions

This paper provides the optimality guarantees on the controllers synthesized for systems that satisfy specifications over infinite length traces. Most previous works that study optimal controller synthesis using formal approaches lack such optimality guarantees [7], [9], [17]. Our approach provides cost guarantees on the trajectory resulting from the suboptimal controller enabling control engineers to make decisions on system design before implementation.

Furthermore, the mixed-integer linear program based techniques in [4], [18] synthesize optimal controllers for a large class of systems and cost functions with LTL specifications. However, unlike our approach which returns a feedback controller, these techniques return an open loop controller that requires another layer of feedback controller to handle disturbances.

In addition, [2], [19] also solve for optimal control under LTL specifications. However, [2] minimizes a specific cost function — the maximum time between satisfying instances of the optimizing proposition, and [19] considers

the weighted average cost. A larger class of cost functions is considered in [20], but it restricts the specifications to a fragment of LTL specifications for computational efficiency. Our method allows for a larger class of cost functions in comparison to previous works [2], [19], and it does not place any restriction on the possible specifications. This generality enables control engineers to synthesize controllers with more flexible requirements and cost considerations.

Lastly, our work is complimentary to [13], [14], [21] that study finite quantitative games. Our approach constructs a two-player weighted graph from the abstraction of a given dynamical system and formulates a two-player mean payoff parity game based on ω-regular properties. The algorithms from [13], [14] solve the mean payoff parity game, and return the optimal strategy that are projected back to the original dynamical system using the approach in our work. A mean payoff parity game may not return a finite strategy in general [14]. However, a finite memory strategy is achievable for an ε-optimal cost through solving an energy parity game [13]. Hence, this paper brings together techniques in finite quantitative games and abstraction based controller synthesis.

The rest of this paper is organized as follows: Section II defines the weighted transition system, the Büchi automaton, and other relevant concepts. The abstraction and refinement of a weighted transition system is provided in Section III. Section IV describes the controller synthesis procedure. Section V formulates the optimal controller synthesis problem for piecewise linear systems, proves the convergence of the controller synthesis procedure, and presents an example. Lastly, Section VI summarizes the paper and states future directions of this work. Proofs are provided in the Appendix.

II. Weighted Transition Systems

This section describes the semantic model for discrete time hybrid systems with cost (i.e., weighted transition systems) and formalizes the optimal control problem. We begin with defining useful mathematical notations that are used throughout this paper.

A. Notation

The sets of real numbers, non-negative real numbers, and natural numbers (including zero) are represented as $\mathbb{R}$, $\mathbb{R}_+$, and $\mathbb{N}$, respectively. The set of integers $\{0, \ldots, k\}$ is written as $[k]$ and a sequence $x_0, \ldots, x_k$, is denoted as $\{x_i\}_{i\in[k]}$. If $z = (z_0, \ldots, z_k) \in \mathbb{R}^{k+1}$ is a vector, $\|z\|_1 = \sum_{t\in[k]} |z_t|$ and $\|z\|_\infty = \max_{t\in[k]} |z_t|$. An ε-ball around a point x is denoted as $\mathcal{B}_\varepsilon(x)$ where $\mathcal{B}_\varepsilon(x) = \{y \in \mathbb{R}^n \mid \|y - x\|_\infty \leq \varepsilon\}$. The function *Grid* splits S into rectangular sets with ϵ width. That is, $\mathit{Grid}(S, \epsilon) =$

$$\left\{ S' \;\middle|\; \exists d_1, \ldots, d_k \in \mathbb{Z}, S' = S \bigcap \prod_{i=1}^{k} (d_i\epsilon, (d_i + 1)\epsilon) \right\}.$$

Given a function, $f : \mathcal{A} \to \mathcal{D}$, for any $A \subseteq \mathcal{A}$, $f(A) = \{f(a) | a \in A\}$. Given an equivalence relation $R \subseteq A \times A$ and an element $a \in A$, $[a]_R = \{b \,|\, (a, b) \in R\}$ denotes the equivalence class of R containing a. Given a finite set $\mathcal{A}$, $|\mathcal{A}|$ denotes the total number of elements in $\mathcal{A}$.

B. Weighted Transition Systems

This section defines the weighted transition systems and related concepts.

Definition 1. *A weighted transition system is defined as* $\mathcal{T} = (\mathcal{S}, \mathcal{S}^{init}, \mathcal{U}, \mathcal{P}, \Delta, \mathcal{L}, \mathcal{W})$*, where:*

- $\mathcal{S}$ *is a set of states;*
- $\mathcal{S}^{init} \subseteq \mathcal{S}$ *is a set of initial states;*
- $\mathcal{U}$ *is a set of control inputs;*
- $\mathcal{P}$ *is a set of propositions;*
- $\Delta \subseteq \mathcal{S} \times \mathcal{U} \times \mathcal{S}$ *is a transition relation;*
- $\mathcal{L} : \mathcal{S} \to \mathcal{P}$ *is a state labeling function, and*
- $\mathcal{W} : \mathcal{S} \times \mathcal{U} \times \mathcal{S} \to \mathbb{R}_+$ *is the transition cost function.*

An equivalent notation for the set of propositions is to define $\mathcal{P}'$ as the set of propositions and let the labeling function map from states to the power set of $\mathcal{P}'$. Here, $\mathcal{P}$ is related to $\mathcal{P}'$ whereby $\mathcal{P} = 2^{\mathcal{P}'}$. Henceforth, a weighted transition system is referred as a transition system. For any $s \in \mathcal{S}$, define the set $\textit{Enabled}(s) = \{u \in \mathcal{U} \mid \exists s' \in \mathcal{S}\ s.t.\ (s, u, s') \in \Delta\}$ to represent all inputs for which there is a transition from the state s. A transition system is *complete* if for all $s \in \mathcal{S}$, $Enabled(s) = \mathcal{U}$. A transition system is *finite* if $\mathcal{S}$ and $\mathcal{U}$ are finite. For the rest of the section, fix the transition system $\mathcal{T} = (\mathcal{S}, \mathcal{S}^{init}, \mathcal{U}, \mathcal{P}, \Delta, \mathcal{L}, \mathcal{W})$.

a) Paths and traces: A *path* of the transition system $\mathcal{T}$ is a finite or infinite sequence of states and inputs, $\zeta = s_0 u_0 s_1 u_1 s_2 \ldots$, where $s_0 \in \mathcal{S}^{init}$, $s_i \in \mathcal{S}$, $u_i \in \mathcal{U}$, and $(s_i, u_i, s_{i+1}) \in \Delta$. A trace of a transition system is the sequence of state labels of a path. The trace of ζ, denoted $\textit{Tr}(\zeta)$, is the sequence $\mathcal{L}(s_0)\mathcal{L}(s_1)\ldots$. The set of all infinite paths of $\mathcal{T}$ is denoted $\textit{Paths}(\mathcal{T})$, and the set of finite paths of $\mathcal{T}$ is denoted as $\textit{Paths}_f(\mathcal{T})$. In the sequel, we will refer to an infinite path as just a path.

b) Properties: A *property* Π over a set of propositions $\mathcal{P}$ is a set of infinite sequences $\pi = p_0 p_1 \ldots$, where each $p_i \in \mathcal{P}$. A property describes the desired behaviors of the system. For algorithmic purposes, we need to restrict ourselves to properties that can be specified using some finite data structure. We consider *ω-regular properties* that are an expressive class of properties involving infinite behaviors which can be represented by Büchi automata.

Definition 2. *A Büchi automaton is defined as* $\mathcal{B} = (\mathcal{Q}, \mathcal{Q}^{init}, \mathcal{P}, \mathcal{E}, \mathcal{F})$*, where:*

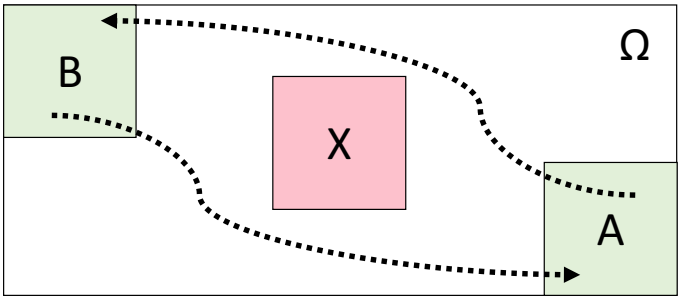
- $\mathcal{Q}$ *is a finite set of states;*
- $\mathcal{Q}^{init} \subseteq \mathcal{Q}$ *is a set of initial states;*
- $\mathcal{P}$ *is a finite set of propositions;*
- $\mathcal{E} \subseteq \mathcal{Q} \times \mathcal{P} \times \mathcal{Q}$ *is a transition relation, and*
- $\mathcal{F} \subseteq \mathcal{Q}$ *is a set of accepting states.*

A sequence π over Π is accepted by the Büchi automaton if there is a "path" in the automaton whose labels correspond to π and the path visits some state from $\mathcal{F}$ infinitely often. A run of a Büchi automaton $\mathcal{B}$ over a sequence $\pi = p_0 p_1 \ldots$ is a sequence $r = q_0 q_1 \ldots$ such that $q_0 \in \mathcal{Q}^{init}$ and $(q_i, p_i, q_{i+1}) \in \mathcal{E}$ for all i. Given a sequence of

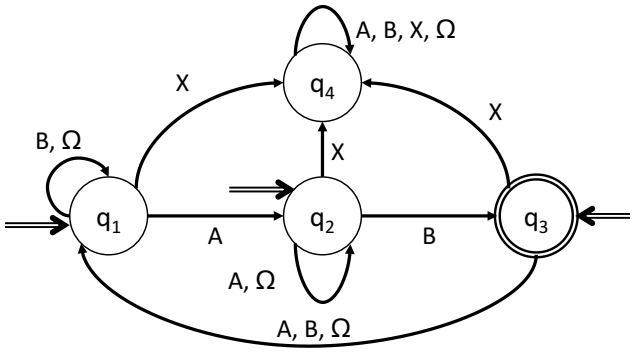
states $r = q_0 q_1 \ldots$, *Inf*(r) denotes the states that occur infinitely often in r, that is, *Inf*$(r) = \{q \,|\, \forall i, \exists j > i, q_j = q\}$, A sequence π is *accepted* by $\mathcal{B}$ if there exists a run r over π such that *Inf*$(r) \cap \mathcal{F} \neq \emptyset$. The *language* of a Büchi automaton $\mathcal{B}$, denoted $\mathcal{L}(\mathcal{B})$, is the set of all sequences π that are accepted by it. A property Π is *ω-regular* if it is the language of some Büchi automaton $\mathcal{B}$, that is, $\Pi = \mathcal{L}(\mathcal{B})$.

Figure 1a shows a navigation scenario where we need to perform surveillance of the regions A and B (visit them infinitely often), while avoiding the region X. Let Ω represent the rest of the region (shown in white). Figure 1b shown a Büchi automaton corresponding to the surveillance objective. Each region is captured using a proposition. We start in state q_1, q_2 or q_3; we move to state q_4 if we encounter X, and never return to $\{q_1, q_2, q_3\}$. We move from q_1 to q_2 on an A and from q_2 to q_3 on a B. So, every time we visit q_3, we have paid at least one visit to both A and B. From state q_3 we return to q_1 to again visit each of A and B.

c) Strategies: A strategy specifies the control inputs to a transition system at different time points. More specifically, a *partial strategy* σ for the transition system $\mathcal{T}$ is a partial function $\sigma : \mathit{Paths}_f(\mathcal{T}) \to \mathcal{U}$ such that for a finite path $\zeta = s_0 u_0 s_1 \ldots u_{i-1} s_i$, $\sigma(\zeta) \in \mathit{Enabled}(s_i)$. A finite or infinite path $\zeta = s_0 u_0 s_1 u_1 s_2 \ldots$ is said to *conform* to a partial strategy σ, if for all i, $\sigma(s_0 u_0 \ldots s_i) = u_i$. A *strategy* σ is a partial strategy such that for all finite paths ζ that conform with σ, $\sigma(\zeta)$ is defined. Let $\mathit{Paths}_\sigma(\mathcal{T}, s_0)$ denote the set of all paths that conform to σ and start at state s_0. Let *Str*$(\mathcal{T})$ denote the set of all strategies for $\mathcal{T}$.



(a)



(b)

Fig. 1. The system is required to visit regions A and B infinitely often and avoid region X. This property is represented as a Büchi automaton in (b). The accepting state is q_3, and the initial states are q_1, q_2 and q_3.

Definition 3. *A strategy σ for the transition system $\mathcal{T}$ is* winning *for a state $s_0 \in \mathcal{S}$ with respect to a property Π over the propositions $\mathcal{P}$, if $Tr(Paths_\sigma(\mathcal{T}, s_0)) \subseteq \Pi$.*

d) Cost of strategies: We consider mean payoff costs as a generalization of average costs for infinite weighted paths. Given a path $\zeta = s_0 u_0 s_1 \dots$ of the transition system $\mathcal{T}$, define:

$$\mathcal{W}(\zeta) = \lim_{N \to \infty} \frac{1}{N} \sum_{j=0}^{N-1} \mathcal{W}(s_j, u_j, s_{j+1}).$$

Consequently, the following proposition holds.

Proposition 1. *Given $\zeta = s_0 u_0 s_1 \dots$ and $\zeta' = s'_0 u'_0\ s'_1 \dots$, if $\mathcal{W}(s_j, u_j, s_{j+1}) \leq \mathcal{W}(s'_j, u'_j, s'_{j+1})$ for all j, then $\mathcal{W}(\zeta) \leq \mathcal{W}(\zeta')$.*

The cost of a strategy σ of the transition system $\mathcal{T}$ with respect to an initial state s_0 is defined as

$$\mathcal{W}(\mathcal{T}, \sigma, s_0) = \sup\{\mathcal{W}(\zeta) \,|\, \zeta \in Paths_\sigma(\mathcal{T}, s_0)\}.$$

Hence, given a property Π over $\mathcal{P}$, the optimal cost of winning $\mathcal{T}$ from an initial state s_0 with respect to a property Π is defined as $\mathcal{W}(\mathcal{T}, s_0, \Pi) =$

$$\inf\{\mathcal{W}(\mathcal{T}, \sigma, s_0) \,|\, \sigma \in Str(\mathcal{T}), Tr(Paths_\sigma(\mathcal{T}, s_0)) \subseteq \Pi\}.$$

The cost is infinity if the minimization is over an empty set. Denote an optimal strategy that achieves the optimal cost as $\sigma(\mathcal{T}, s_0, \Pi)$. Note that the optimal strategy may not be unique or even exist.

e) Optimal control problem: Given the transition system $\mathcal{T}$, an initial state s_0 and a property Π, the optimal control problem is to compute an optimal winning strategy from s_0 with respect to Π, if it exists, and the optimal cost of winning.

III. Abstraction/Refinement

In this section, we extend certain results on the construction of abstractions and refinements from [15] for regular objectives to the case of ω-regular objectives. Broadly, [15] describes a procedure for constructing finite state abstractions from potentially infinite state transition systems using equivalence relations on the state and input spaces. In particular, the state and input spaces are divided into finite number of parts, and they are used as symbolic states and inputs, respectively, in the abstract transition system. The concrete (given) system $\mathcal{T}_1$ and the abstract (finite) system $\mathcal{T}_2$, are related using certain pre-orders called simulation relations that preserve the existence and optimality of the costs of the controllers. In particular, if $\mathcal{T}_2$ simulates $\mathcal{T}_1$, denoted, $\mathcal{T}_1 \preceq \mathcal{T}_2$, then there is a correspondence between controllers for $\mathcal{T}_2$ and $\mathcal{T}_1$ in the sense that for every controller for $\mathcal{T}_2$, there is a controller for $\mathcal{T}_1$ such that the cost of the controller for $\mathcal{T}_2$ is an upper bound on that for the controller for $\mathcal{T}_1$. Hence, the cost of the optimal controller for $\mathcal{T}_2$ provides an upper-bound on the cost of any controller for $\mathcal{T}_1$. Here, we briefly review the construction of abstract systems, and observe that the same results carry over when we consider ω-regular objectives instead of the regular objectives.

A. Abstraction

Let us fix a transition system $\mathcal{T} = (\mathcal{S}, \mathcal{S}^{init}, \mathcal{U}, \mathcal{P}, \Delta, \mathcal{L}, \mathcal{W})$ Given the transition system $\mathcal{T}$, and two equivalence relations $\equiv_S$ and $\equiv_U$ on the state space $\mathcal{S}$ and the input space $\mathcal{U}$, respectively, an abstraction function constructs an abstract transition system $Abs(\mathcal{T}, \equiv_S, \equiv_U)$. An equivalence relation $\equiv_S$ on $\mathcal{S}$ respects $\mathcal{L}$, if for all $(s_1, s_2) \in \equiv_S$, $\mathcal{L}(s_1) = \mathcal{L}(s_2)$. Furthermore, an equivalence relation $\equiv_S$ on $\mathcal{S}$ respects $\mathcal{S}^{init}$, if for all $(s_1, s_2) \in \equiv_S$, $s_2 \in \mathcal{S}^{init}$ where $s_1 \in \mathcal{S}^{init}$.

Definition 4. *Let $\equiv_S \subseteq \mathcal{S} \times \mathcal{S}$ and $\equiv_U \subseteq \mathcal{U} \times \mathcal{U}$ be two equivalence relations of finite index such that $\equiv_S$ respects the labeling function $\mathcal{L}$ and the initial states $\mathcal{S}^{init}$. $Abs(\mathcal{T}, \equiv_S, \equiv_U) = (\mathcal{S}', \mathcal{S}^{init'}, \mathcal{U}', \mathcal{P}, \Delta', \mathcal{L}', \mathcal{W}')$, where:*

- $\mathcal{S}' = \{[s]_{\equiv_S} \mid s \in \mathcal{S}\}$ *is the equivalence classes of* $\equiv_S$*;*
- $\mathcal{S}^{init'} = \{[s]_{\equiv_S} \mid s \in \mathcal{S}^{init}\} \subseteq \mathcal{S}'$*;*
- $\mathcal{U}' = \{[u]_{\equiv_U} \mid u \in \mathcal{U}\}$ *is the equivalence classes of* $\equiv_U$*;*
- $\Delta' = \{(S_1, U, S_2) \mid \exists s \in S_1,\ s' \in S_2,\ u \in U,\ s.t.\ (s, u, s') \in \Delta\}$*;*
- *For* $S \in \mathcal{S}'$, $\mathcal{L}'(S) = \mathcal{L}(s)$ *for any* $s \in S$*, and*
- *For* $(S_1, U, S_2) \in \Delta'$, $\mathcal{W}'(S_1, U, S_2) = \sup\{\mathcal{W}(s_1, u,\ s_2) \mid s_1 \in S_1, s_2 \in S_2, u \in U, (s_1, u, s_2) \in \Delta\}$.

Call $\mathcal{T}$ the concrete system and $Abs(\mathcal{T}, \equiv_S, \equiv_U)$ the abstract system. Then, the abstract system simulates the concrete system.

Proposition 2 ([15, Proposition 4])**.** *If $\mathcal{T}$ is a complete transition system, $\mathcal{T} \preceq Abs(\mathcal{T}, \equiv_S, \equiv_U)$.*

The next theorem repeated from [15] states that simulation relations preserve the existence and optimality of controllers when we consider properties over infinite traces. The proof is available in Appendix A.

Theorem 1. *Given two transition systems $\mathcal{T}_i = (\mathcal{S}_i, \mathcal{S}_i^{init}, \mathcal{U}_i, \mathcal{P}, \Delta_i, \mathcal{L}_i, \mathcal{W}_i)$ for $i = 1, 2$, let Π be a property over a set of propositions $\mathcal{P}$, $\mathcal{T}_1 \preceq_{(\alpha,\beta)} \mathcal{T}_2$ and $(s_0, s_0') \in \alpha$ for $s_0 \in \mathcal{S}_1^{init}$ and $s_0' \in \mathcal{S}_2^{init}$. If there exists a winning strategy σ_2 for $\mathcal{T}_2$ from s_0' with respect to Π, then there exists a winning strategy σ_1 for $\mathcal{T}_1$ from s_0 with respect to Π such that $\mathcal{W}_1(\mathcal{T}_1, \sigma_1, s_0) \leq \mathcal{W}_2(\mathcal{T}_2, \sigma_2, s_0')$. Hence, $\mathcal{W}_1(\mathcal{T}_1, s_0, \Pi) \leq \mathcal{W}_2(\mathcal{T}_2, s_0', \Pi)$.*

B. Refinement

A sequence of abstract systems which are closer to the original system than their predecessors in the sequence can be constructed by choosing finer equivalence relations on the state and input spaces.

Definition 5. *Let $\mathcal{T}_1$ and $\mathcal{T}_3$ be transition systems such that $\mathcal{T}_1 \preceq \mathcal{T}_3$. A transition system $\mathcal{T}_2$ is said to be a refinement of $\mathcal{T}_3$ with respect to $\mathcal{T}_1$, if $\mathcal{T}_1 \preceq \mathcal{T}_2 \preceq \mathcal{T}_3$.*

Proposition 3 ([15, Proposition 5])**.** *Let $\equiv_S, \equiv_S' \subseteq \mathcal{S} \times \mathcal{S}$ and $\equiv_U, \equiv_U' \subseteq \mathcal{U} \times \mathcal{U}$ be equivalence relations of finite index such that $\equiv_S' \subseteq \equiv_S$ and $\equiv_U' \subseteq \equiv_U$. Then, $Abs(\mathcal{T}, \equiv_S', \equiv_U')$ is a refinement of $Abs(\mathcal{T}, \equiv_S, \equiv_U)$ with respect to $\mathcal{T}$.*

IV. Optimal Control for ω-regular Objectives

In this section, given a weighted transition system $\mathcal{T}$, an initial state s_0, and an ω-regular property Π, we seek to synthesize an optimal controller σ for $\mathcal{T}$. The first part of this section discusses the overall abstraction refinement procedure and the main components of the framework. This framework summarized in Algorithm 1 is similar to Algorithm 1 in [15]. Next part discusses one of the components of Algorithm 1 that is specific to ω-regular objectives, namely, optimal strategy synthesis procedure given a finite state transition system, an initial state, and an ω-regular property Π.

A. Abstraction-Refinement Procedure

Algorithm 1 presents the abstraction-refinement procedure that is similar to Algorithm 1 in [15]. It first partitions the state and input spaces of the given system $\mathcal{T}$ into grids with particular cell sizes, ϵ, and constructs an abstract system $\hat{\mathcal{T}}$ using *ConsAbs*. Then, it computes the optimal cost J and strategy for the abstract system by solving a two-player game using *SolveTwoPlayerGame*. A suboptimal strategy for $\mathcal{T}$ is extracted from the strategy of the abstract system $\hat{\mathcal{T}}$ with the cost upper bounded by J using the procedure *ExtractController*. To reduce J, refine the state space partitions using smaller grids size, $\epsilon/2$, and repeat the whole process. The details of each of the functions can be found in [15]. Here, instead of solving a two-player game over finite sequences on a finite graph, Algorithm 1 solves a two-player game over infinite sequences on a finite graph because the ω-regular objectives consist of infinite traces. This step is explained in detail in the next subsection.

This algorithm outputs a sequence of suboptimal strategies, whose costs are non-increasing over the iterations. Note that the optimal cost for $\mathcal{T}$ might not be achievable due to the optimality gap ε. Hence, the algorithm is chosen to terminate after a fixed number of iterations. However, for a special class of systems that has a "lasso-type" robust optimal path, we show that the suboptimal cost converges to the optimal cost as i goes to infinity. See Section V-C for more details.

Algorithm 1 OptCAR (Abstraction Refinement Procedure)

Require: System $\mathcal{T}$, property Π as a finite state automaton, initial state s_0, rational number $\varepsilon_0 > 0$, maximum iterations i_m, and optimality gap ε

Set $\varepsilon := \varepsilon_0$, $i := 0$

while $i < i_m$ **do**

$\hat{\mathcal{T}}, \hat{x}_0 :=$ *ConsAbs*$(\mathcal{T}, s_0, \varepsilon)$

$J, \hat{\sigma} :=$ *SolveTwoPlayerGame*$(\hat{\mathcal{T}}, \hat{x}_0, \Pi, \varepsilon)$

$\sigma :=$ *ExtractController*$(\hat{\sigma}, \hat{\mathcal{T}}, \mathcal{T})$

Output σ and J

$\varepsilon := \frac{\varepsilon}{2}$

$i := i + 1$

end while

A winning strategy $\hat{\sigma}$ might not exist when the partitioning is coarse even if the underlying system $\mathcal{T}$ has an optimal solution. However, once a winning strategy is found for a particular partitioning, a controller that has bounded trajectory cost is obtained. Furthermore, a more refined partitioning will return a controller with trajectory cost that is no worse than the trajectory cost resulting from the controller corresponding to the winning strategy for the current partitioning. Hence, the algorithm can be terminated at a specific iteration based on the application and the computational resources.

B. Synthesizing Optimal Strategy

Here, we explain the procedure to compute the optimal cost and optimal strategy for a finite transition system $\mathcal{T}$ with respect to a ω-regular property Π performed by the subroutine *SolveTwoPlayerGame* in Algorithm 1 that is different from the previous work [15]. Algorithm 2 reduces the problem of optimal strategy synthesis for general ω-regular objectives to that of a specific ω-regular objective which is an instance of a mean payoff parity games. Essentially, it constructs a weighted product transition system from the weighted transition system $\mathcal{T}$ and a Büchi automaton that encodes the ω-regular property Π. This product system has an associated ω-regular property Π_p that is the independent of $\mathcal{T}$ and Π. Solving for the optimal winning strategy of the product system with respect to Π_p is an instance of solving a mean payoff parity game [14]. Once the optimal cost and winning strategy $\sigma_{\mathcal{A}}$ of the product system are computed, the winning strategy for $\mathcal{T}$ is extracted from $\sigma_{\mathcal{A}}$ by projecting $\sigma_{\mathcal{A}}$ onto $\mathcal{T}$. Finally, Algorithm 2 returns both the optimal cost and the winning strategy for $\mathcal{T}$.

More precisely, the function *ProductSystem* forms a weighted product transition system that is a product of a weighted transition system $\mathcal{T}$ and the Büchi automaton $\mathcal{B}$ representing the property Π. From here onwards, fix $\mathcal{T} = (\mathcal{S}, \mathcal{S}^{init}, \mathcal{U}, \mathcal{P}, \Delta, \mathcal{L}, \mathcal{W})$ as the weighted transition system and $\mathcal{B} = (\mathcal{Q}, \mathcal{Q}^{init}, \mathcal{P}, \mathcal{E}, \mathcal{F})$ as the Büchi automaton representing the property Π where $\Pi = \mathcal{L}(\mathcal{B})$.

Definition 6. *A weighted product transition system of $\mathcal{T}$ and $\mathcal{B}$ is defined as $\mathcal{A}_{\mathcal{T},\mathcal{B}} = (\mathcal{Q}_{\mathcal{A}}, \mathcal{Q}^{init}_{\mathcal{A}}, \mathcal{U}_{\mathcal{A}}, \mathcal{P}_{\mathcal{A}}, \Delta_{\mathcal{A}}, \mathcal{L}_{\mathcal{A}}, \mathcal{W}_{\mathcal{A}})$ where*

- $\mathcal{Q}_{\mathcal{A}} = \{(s, q) \in \mathcal{S} \times Q\}$;
- $\mathcal{Q}^{init}_{\mathcal{A}} = \{(s, q) \in \mathcal{S}^{init} \times \mathcal{Q}^{init}\}$;
- $\mathcal{U}_{\mathcal{A}} = \mathcal{U} \times \mathcal{E}$;
- $\mathcal{P}_{\mathcal{A}} = \{0, 1\}$;
- $\Delta_{\mathcal{A}} = \{((s, q), (u, e), (s', q')) \in \mathcal{Q}_{\mathcal{A}} \times \mathcal{U}_{\mathcal{A}} \times \mathcal{Q}_{\mathcal{A}} \mid (s, u, s') \in \Delta, e = (q, p, q') \in \mathcal{E}, \mathcal{L}(s) = p\}$;
- $\mathcal{L}_{\mathcal{A}} : \mathcal{Q}_{\mathcal{A}} \to \{0, 1\}$ *where* $\mathcal{L}_{\mathcal{A}}(s, q) = 0$ *if* $q \in \mathcal{F}$ *and* $\mathcal{L}_{\mathcal{A}}(s, q) = 1$ *if otherwise, and*
- $\mathcal{W}_{\mathcal{A}} : \mathcal{Q}_{\mathcal{A}} \times \mathcal{U}_{\mathcal{A}} \times \mathcal{Q}_{\mathcal{A}} \to \mathbb{R}_+$ *such that* $\mathcal{W}_{\mathcal{A}}((s, q), (u, e), (s', q')) = \mathcal{W}(s, u, s')$.

Fix $\mathcal{A}_{\mathcal{T},\mathcal{B}} = (\mathcal{Q}_{\mathcal{A}}, \mathcal{Q}^{init}_{\mathcal{A}}, \mathcal{U}_{\mathcal{A}}, \mathcal{P}_{\mathcal{A}}, \Delta_{\mathcal{A}}, \mathcal{L}_{\mathcal{A}}, \mathcal{W}_{\mathcal{A}})$. We consider the following winning objective Π_p for $\mathcal{A}_{\mathcal{T},\mathcal{B}}$: Π_p consists of all sequences over $\{0, 1\}$ that contain infinitely many 0s. Note that this objective is the same for any $\mathcal{T}$ and $\mathcal{B}$, and its simple form enables the problem to be posed as a mean payoff parity game.

Before moving on to the mean payoff parity game, we will discuss the relationship between the winning strategy $\sigma_{\mathcal{A}}$ of the product system $\mathcal{A}_{\mathcal{T},\mathcal{B}}$ and the winning strategy σ of the corresponding transition system $\mathcal{T}$. To do

so, we need to define the projection of a strategy. Let $\zeta_{\mathcal{A}} = (s_0, q_0)(u_0, e_0)(s_1, q_1)(u_1, e_1)(s_2, q_2)\dots$ be a path conforming to $\sigma_{\mathcal{A}}$ from initial state (s_0, q_0) and let $\zeta^i_{\mathcal{A}} = (s_0, q_0)(u_0, e_0)\dots(s_i, q_i)$ be a prefix of $\zeta_{\mathcal{A}}$ up to index i. Furthermore, let $\mathit{Proj}_{\mathcal{T}}(\zeta_{\mathcal{A}})$ be the projection of a path $\zeta_{\mathcal{A}}$ in $\mathcal{A}_{\mathcal{T},\mathcal{B}}$ onto $\mathcal{T}$ where $\mathit{Proj}_{\mathcal{T}}(\zeta_{\mathcal{A}}) = s_0u_0s_1u_1\dots$ and $\mathit{Proj}_{\mathcal{T}}(\zeta^i_{\mathcal{A}}) = s_0u_0\dots s_i$. Then, the projection of a strategy $\sigma_{\mathcal{A}}$ of $\mathcal{A}_{\mathcal{T},\mathcal{B}}$ onto $\mathcal{T}$ is defined as $\sigma = \mathit{Proj}^{\sigma}_{\mathcal{T}}(\sigma_{\mathcal{A}})$ whereby if $\sigma_{\mathcal{A}}(\zeta^i_{\mathcal{A}}) = (u_i, e_i)$, then $\sigma(\mathit{Proj}_{\mathcal{T}}(\zeta^i_{\mathcal{A}})) = u_i$ for all $i \geq 0$. This projection forms a "one-to-one" cost preserving correspondence between $\sigma_{\mathcal{A}}$ and σ.

Theorem 2. *A winning strategy $\sigma_{\mathcal{A}}$ of $\mathcal{A}_{\mathcal{T},\mathcal{B}}$ with respect to Π_p from the state (s_0, q_0) exists if and only if a winning strategy σ for $\mathcal{T}$ with respect to $\mathcal{L}(\mathcal{B})$ from s_0 exists. Furthermore, $\sigma = \mathit{Proj}^{\sigma}_{\mathcal{T}}(\sigma_{\mathcal{A}})$ is a winning strategy for $\mathcal{T}$, and $\mathcal{W}(\mathcal{A}_{\mathcal{T},\mathcal{B}}, \sigma_{\mathcal{A}}, (s_0, q_0)) = \mathcal{W}(\mathcal{T}, \sigma, s_0)$.*

The proof is available in Appendix B. Theorem 2 implies that if a winning strategy for $\mathcal{T}$ exists, there will be a winning strategy $\sigma_{\mathcal{A}}$ for $\mathcal{A}_{\mathcal{T},\mathcal{B}}$ that the algorithm can search for. Furthermore, given $\sigma_{\mathcal{A}}$, we can construct a winning strategy σ for $\mathcal{T}$ by projection, and the cost of σ is the same as the cost of $\sigma_{\mathcal{A}}$. Hence, *ExtractStrategy* in Algorithm 2 extracts a winning strategy for $\mathcal{T}$ from $\sigma_{\mathcal{A}}$ by projecting $\sigma_{\mathcal{A}}$ onto $\mathcal{T}$. Note that *ExtractStrategy* is different from *ExtractController* in Algorithm 1 because *ExtractController* extracts a suboptimal controller for the concrete system $\mathcal{T}$ from the optimal strategy of the abstract system $\hat{\mathcal{T}}$.

Now, we return to the mean payoff parity game. A mean payoff parity game is a two-player game on a finite graph where by the strategy for player 1 is winning if $\min(\mathit{Inf}(\mathit{Tr}(\zeta)))$ is even for all paths ζ that conform to the strategy, and optimal if the cost of the strategy is minimized [14]. Solving for an optimal winning strategy for $\mathcal{A}_{\mathcal{T},\mathcal{B}}$ with respect to Π_p is an instance of a mean payoff parity game. Hence, given $\mathcal{A}_{\mathcal{T},\mathcal{B}}$ and Π_p, *MeanPayoffParityGame* solves a mean payoff parity game to obtain the winning strategy $\sigma_{\mathcal{T}}$ that is ε-optimal, and the optimal value C for each states. If a state s is not winning, $C(s) = \infty$. The strategy is ε-optimal because a mean payoff parity game may not have a finite memory optimal strategy in general [14]. However, a finite memory strategy is achievable for an ε-optimal cost through solving an energy parity game [13]. The example that follows uses the algorithm in [13] to solve for an ε-optimal strategy. Due to this optimality gap, we do not expect to obtain the optimal control for the concrete system in general. Instead, this technique returns a suboptimal controller that has a finite memory. Nonetheless, if the optimal controller has finite memory, the optimality gap is zero.

Algorithm 2 *SolveTwoPlayerGame* (Two-Player Games)

Require: Finite state transition system $\mathcal{T}$, property Π specified as $\mathcal{B}$, initial state s_0, and optimality gap ε.

$\mathcal{A}_{\mathcal{T},\mathcal{B}} := \mathit{ProductSystem}(\mathcal{T}, \mathcal{B})$

$C, \sigma_{\mathcal{A}} := \mathit{MeanPayoffParityGame}(\mathcal{A}_{\mathcal{T},\mathcal{B}}, \varepsilon)$

$\sigma := \mathit{ExtractStrategy}(\sigma_{\mathcal{A}})$

if $C(s_0) < \infty$ **then**

  Output the strategy σ and the cost $C(s_0)$

end if

C. Complexity of Algorithm 1 and 2

The computation cost for Algorithm 1 is mostly due to *ConsAbs*. This step involves $|\mathcal{S}|^2|\mathcal{U}|$ parallelizable optimizations where $|\mathcal{S}|$ is the number of state in $\hat{\mathcal{T}}$, and $|\mathcal{U}|$ is the number of inputs in $\hat{\mathcal{T}}$. All optimizations are linear programs if the system is linear, and the state and input partitions are represented by polyhedra.

In Algorithm 2, *ProductSystem* runs in $O(|\mathcal{S}||\mathcal{Q}||\Delta||\mathcal{E}|)$ time where $|\mathcal{S}|$ and $|\Delta|$ are the number of states and transitions in $\mathcal{T}$, and $|\mathcal{Q}|$ and $|\mathcal{E}|$ are the number of states and transitions in $\mathcal{B}$. The mean payoff parity games solves in $O(|\Delta_{\mathcal{A}}||\mathcal{Q}_{\mathcal{A}}|\mathcal{W}^m_{\mathcal{A}}(|\mathcal{Q}_{\mathcal{A}}|+1))$ time [13] where $|\mathcal{Q}_{\mathcal{A}}|$ and $|\Delta_{\mathcal{A}}|$ are the number of states and transitions in $\mathcal{A}$, and $\mathcal{W}^m_{\mathcal{A}}$ is the maximum value of the cost $\mathcal{W}$ over all transitions.

V. Optimal Control of Piecewise Linear Systems

The abstraction-refinement approach is implemented for discrete-time piecewise linear systems, and the algorithm converges to the optimal controller if the optimal controller is robust and the optimal path is a "lasso". We first formally define the optimal control problem for this class of system with ω-regular objectives.

A. Problem Formulation

A discrete-time piecewise linear system is a tuple $(\mathcal{X}, \mathcal{X}^{init}, \mathcal{U}, \mathcal{P}, \{(A_i, B_i, P_i)\}_{i\in[m]}, \mathcal{L}_d, \mathcal{J})$, where the state-space $\mathcal{X} \subseteq \mathbb{R}^n$ and the input-space $\mathcal{U} \subseteq \mathbb{R}^p$ are compact sets, $\mathcal{X}^{init} \subseteq \mathcal{X}$ is the set of initial states, $\mathcal{P}$ is a finite set of propositions, $A_i \in \mathbb{R}^{n\times n}, B_i \in \mathbb{R}^{n\times p}$ and P_i is a polyhedral set, such that $\{P_i\}_{i\in m}$ is a polyhedral partition of $\mathcal{X}$, $\mathcal{L}_d : [m] \to \mathcal{P}$ is a labeling function, and $\mathcal{J} : \mathcal{X} \times \mathcal{U} \to \mathbb{R}_+$ is a continuous cost function. Note that A_i and B_i do not have to be different for different i. Given an initial state $x_0 \in \mathcal{X}^{init}$ and a sequence of control inputs $\mathbf{u} = u_0u_1\ldots$, where $u_t \in \mathcal{U}$, $\phi(x_0, \mathbf{u}) = x_0x_1\ldots$ is the sequence of states visited under the control $\mathbf{u}$, where $x_{t+1} = A_ix_t + B_iu_t$. The cost of the sequence $\phi(x_0, \mathbf{u})$, $\mathcal{J}(\phi(x_0, \mathbf{u}))$, is given by $\lim_{k\to\infty} \frac{1}{k} \sum_{t\in[k-1]} \mathcal{J}(x_{t+1}, u_t)$. Define the partition sequence of $z = x_0x_1\ldots$, denoted *PS*(z), to be the sequence of partitions visited by the states $P_1P_2\ldots$ such that $x_t \in P_t$ for all t.

Problem 1 (Optimal control problem)**.** *Given an n-dimensional discrete-time piecewise linear system $\mathcal{D} = (\mathcal{X}, \mathcal{X}^{init}, \mathcal{U}, \mathcal{P}, \{(A_i, B_i, P_i)\}_{i\in[m]}, \mathcal{L}_d, \mathcal{J})$, a state $x_0^* \in \mathcal{X}^{init}$ and a ω-regular property Π over $\mathcal{P}$, find a sequence of control inputs $\mathbf{u}^*$ for which $\mathcal{L}_d(\phi(x_0^*, \mathbf{u}^*)) \in \Pi$ and $\mathcal{J}(\phi(x_0^*, \mathbf{u}^*))$ is minimized.*

B. Equivalent Optimal Strategy Problem

A discrete-time piecewise linear system $\mathcal{D} = (\mathcal{X}, \mathcal{X}^{init}, \mathcal{U}, \mathcal{P}, \{(A_i, B_i, P_i)\}_{i\in[m]}, \mathcal{L}_d, \mathcal{J})$ can be represented as a weighted transition system, $\mathcal{T}_{\mathcal{D}} = (\mathcal{X}, \mathcal{X}^{init}, \mathcal{U}, \mathcal{P}, \Delta, \mathcal{L}, \mathcal{W})$ where $\Delta = \{(x, u, x') \in \mathcal{X} \times \mathcal{U} \times \mathcal{X} \mid x' = A_ix + B_iu, \mathcal{L}(x) = \mathcal{L}_d(i)$ where $x \in P_i$, and $\mathcal{W}(x, u, x') = \mathcal{J}(x', u)$. Consequently, Problem 1 is equivalent to the following problem:

Problem 2 (Optimal strategy problem)**.** *Given a weighted transition system $\mathcal{T}_{\mathcal{D}} = (\mathcal{X}, \mathcal{X}^{init}, \mathcal{U}, \mathcal{P}, \Delta, \mathcal{L}, \mathcal{W})$, a state $x_0^* \in \mathcal{X}^{init}$ and a ω-regular property Π over $\mathcal{P}$, find an optimal winning strategy $\sigma(\mathcal{T}_{\mathcal{D}}, x_0^*, \Pi)$ for which the optimal cost of winning $\mathcal{T}_{\mathcal{D}}$ with respect to Π, $\mathcal{W}(\mathcal{T}_{\mathcal{D}}, x_0^*, \Pi)$, is achieved.*

Solving Problem 2 in general is difficult because $\mathcal{T}_{\mathcal{D}}$ is a infinite state system; nonetheless, we can synthesize suboptimal strategies using the approach described in Section IV with formal guarantee on cost sub-optimality. Furthermore, we show that if the transition system $\mathcal{T}_{\mathcal{D}}$ has a robust optimal control and the optimal path is a lasso, the suboptimal cost converges to the optimal cost.

C. Analysis of Algorithm 1 for Problem 2

This section analyzes the output of Algorithm 1 for Problem 2 for a special class of systems that has a robust optimal control and a lasso optimal path. The suboptimal cost converges to the optimal cost for this system.

Definition 7. *A path is a **lasso** if the path has the form $s_p(s_l)^\omega$ where for some constants $k > 0$ and $m > 0$, $s_p = s_0u_0s_1u_1 \dots s_{k-1}u_{k-1}$ (referred to as the chain), and $s_l = s_ku_k \dots s_{k+m-1}u_{k+m-1}$ (referred to as the loop).*

Definition 8. *Let $\mathbf{u}$ be the input sequence that results in a lasso trajectory $\phi(x_0, \mathbf{u}) = x_p(x_l)^\omega$ for an initial state x_0 where $x_p = \{x_i\}_{i\in[k-1]}$, $x_l = \{x_{j+k}\}_{j\in[m-1]}$, and PS$(\phi(x_0, \mathbf{u})) = \{P_{i_t}\}_{t\in\mathbb{N}}$. This input sequence $\mathbf{u}$ is said to be* robust *with respect to the initial state x_0 if*

1) *there exists $\varepsilon_t > 0$ such that $\mathcal{B}_{\varepsilon_t}(x_t) \subseteq P_{i_t}$ for all $t \in \mathbb{N}$;*
2) *there exist bounds $M_x > 0$ and $M_u > 0$, and constant $L < 1$ such that for all $\{x'_i\}_{i\in[m]}$ and $\{u'_i\}_{i\in[m-1]}$, with $(x'_i, u'_i, x'_{i+1}) \in \Delta$ for all $i \in [m-1]$, $x'_0 \in \mathcal{B}_{M_x}(x_k)$ and $u'_i \in \mathcal{B}_{M_u}(u_{k+i})$ for all $i \in [m-1]$, we have $\|x'_m - x_k\|_\infty \leq L \|x'_0 - x_k\|_\infty$.*

Robustness ensures that the sequence of labels of the optimal lasso do not change when the initial state is perturbed slightly, and the perturbed trajectory does not diverge every time it makes a loop. Instead, it becomes strictly closer to the optimal lasso. Note that the second property can be achieved trivially if the system is stable.

Henceforth, let $\mathbf{u}^* = u_p^*(u_l^*)^\omega$ be a robust optimal control input sequence with respect to x_0^* where $u_p^* = \{u_i^*\}_{i\in[k-1]}$ and $u_l^* = \{u_{j+k}^*\}_{j\in[m-1]}$. In addition, let $\phi(x_0^*, \mathbf{u}^*) = x_p^*(x_l^*)^\omega$ be the corresponding optimal trajectory for Problem 1 where $x_p^* = \{x_i^*\}_{i\in[k-1]}$ and $x_l^* = \{x_{j+k}^*\}_{j\in[m-1]}$. Lastly, let $\zeta^* = s_p^*(s_l^*)^\omega$ be the corresponding optimal state and input sequence where $s_p^* = \{x_i^*u_i^*\}_{i\in[k-1]}$ and $s_l^* = \{x_{j+k}^*u_{j+k}^*\}_{j\in[m-1]}$. The proof requires a special kind of strategy which has a unique path conforming to it by choosing inputs that result in exactly one successor state.

Definition 9. *A **chain lasso strategy** for a transition system $\mathcal{T}$ and an initial state s_0 is a strategy $\sigma \in$ Str$(\mathcal{T})$ such that there is one path in Paths$_\sigma^m(\mathcal{T}, s_0)$ and the path is a lasso.*

Next, we will show that the strategy given by Algorithm 1 produces a suboptimal cost that converges to the optimal cost of $\mathcal{D}$. Denote the elements in the iteration of Algorithm 1 corresponding to a particular ε as $\hat{\mathcal{T}}_\varepsilon$ for $\hat{\mathcal{T}}$, $\hat{x}_0^\varepsilon$ for $\hat{x}_0$, J_ε for J, $\hat{\sigma}_\varepsilon$ for $\hat{\sigma}$ and σ_ε for $\sigma_{\mathcal{D}}$.

Theorem 3. *If there exists a robust optimal control $\mathbf{u}^*$ with respect to x_0^* for Problem 1 and the optimal path is a lasso, the sequence of sub-optimal costs $\{J_{\varepsilon_0/2^i}\}_{i\in\mathbb{N}_+}$ output by Algorithm 1 converges to the optimal cost*

$J_{opt} = \mathcal{W}(\mathcal{T}_{\mathcal{D}}, x_0, \Pi)$. *Furthermore, for each sub-optimal cost* $J_{\varepsilon_0/2^i}$*, there exists a suboptimal winning strategy* $\sigma_{\varepsilon_0/2^i}$ *with cost of winning* $J_{\varepsilon_0/2^i}$.

Proof. A sketch of the proof is provided. The full proof is available in Appendix C. First, note that given Definition 8, if the initial states and the inputs are slightly perturbed from the optimal inputs, the perturbed trajectory has error that is bounded relative to the optimal trajectory. In addition, given the cost function in Problem 2, the cost of this suboptimal trajectory is bounded due to continuity of the state transitions. Because the error and the cost are bounded, we can construct an abstraction to give a chain lasso winning strategy σ for a given concrete system $\mathcal{T}_{\mathcal{D}}$ such that the cost of σ is bounded by any given cost sub-optimality $\delta > 0$ with respect to the optimal cost.

To obtain this chain lasso strategy, we will construct a sequence of disjoint neighborhoods $\{N_t^x\}_{t\in[k+m-1]}$ and $\{N_t^u\}_{t\in[k+m-1]}$ around the optimal states and inputs such that every transition from N_t^x on N_t^u will end in N_{t+1}^x, and each neighborhood is an element or a collection of elements in $\varepsilon_0/2^i$ grid for $i \in \mathbb{N}$. The grid size can be different for different neighborhoods, and the neighborhoods is disjoint by picking a small enough ε_0. The same approach as [22, Lemma 9] constructs the chain of the chain lasso strategy backward inductively from the beginning of the loop. The loop of the chain lasso strategy is constructed backward inductively as illustrated by Figure 2. The white square is N_k^x. The white circles are neighborhoods that are reachable from N_k^x on $N_t^u \subseteq \mathcal{B}_{M_u}(u_t^*)$ for $t \geq k$. At $k+m$, define a blue circle such that it is contained within the white square and the white circle is contained within the blue circle. Then, define the red circle at $k+m-1$ such that all states within the red circle will reach a state within the subsequent blue circle on N_{k+m-1}^u. Now, construct the collection of squares in $k+m-1$ to form N_{k+m-1}^x whereby the squares are elements in $\varepsilon_0/2^i$ grid for $i \in \mathbb{N}$. This sequence of construction can be performed backward from $k+m$ to $k+1$, and it ensures that every transition from N_t^x on N_t^u will end in N_{t+1}^x. Furthermore, $|\mathcal{W}(\zeta) - \mathcal{W}(\zeta^*)| \leq \delta$ for any path ζ starting in an ε_x ball around x_0^*. Hence, we have a chain lasso strategy σ with bounded cost.

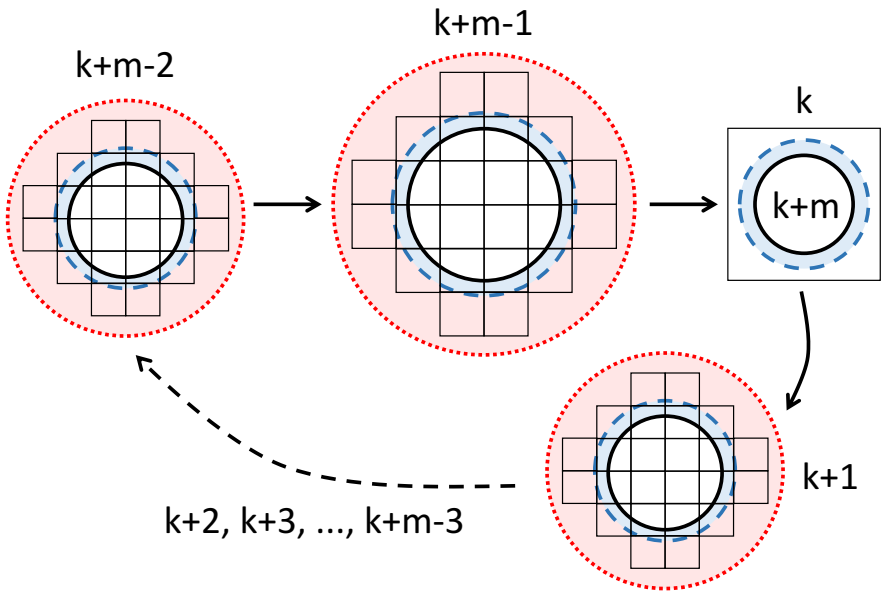


Fig. 2. Construction of neighborhoods for the loop in the optimal path.

Next, refine σ to construct a uniform grid by choosing the size of the uniform grid ε to be the smallest grid of all neighborhoods N_t^u and N_t^x. Then, define a strategy σ_ε (not necessarily a chain lasso anymore) for $\mathcal{T}_\varepsilon$ which follows the neighborhoods N_t^x. All paths in $\mathcal{T}_\varepsilon$ conforming to σ_ε are contained in the neighborhoods N_t^x. Thus, the cost of σ_ε is bounded by that of σ, and consequently, the optimal cost of $\mathcal{T}_\varepsilon$ is at most δ larger than that of $\mathcal{T}_\mathcal{D}$.

Lastly, because the optimal path is a lasso, the optimal strategy has a finite memory, and thus, the optimality gap ε is zero. As a result, $J_\varepsilon = \mathcal{W}(\mathcal{T}_\varepsilon, x_0^\varepsilon, \Pi)$, $J_{opt} = \mathcal{W}(\mathcal{T}_\mathcal{D}, x_0^*, \Pi)$ is the optimal cost, and $|J_\varepsilon - J_{opt}| \leq \delta$ for a given $\delta > 0$. In fact, for each J_ε, there exists a suboptimal winning strategy σ_ε with cost of winning J_ε. In addition, $J_{\varepsilon_0/2^i}$ converges to J_{opt} as $i \to \infty$ because $J_{\varepsilon_0/2^i} \leq J_{\varepsilon_0/2^j}$ for all $i > j$. □

The trace of all the paths given by the strategies in the proof of Theorem 3 is the same. During implementation, Algorithm 1 may return a sequence of suboptimal strategies $\sigma_{\varepsilon_0/2^i}$ that results in paths with different lengths for the chain part and/or the loop part. Nonetheless, the cost of each path results from $\sigma_{\varepsilon_0/2^i}$ is bounded by the cost $J_{\varepsilon_0/2^i}$.

D. Example

Consider a surveillance example that is represented in Figure 1a. A surveillance robot is required to visit region A and B repeatedly while avoiding obstacle X. In addition, the robot has to move using minimal amount of energy. Mathematically, the robot dynamics can be represented with the following linear dynamical system:

$$x_{t+1} = x_t + u_t$$

$$y_{t+1} = y_t + v_t$$

where $[x_t, y_t] \in [-4, 4]^2$, and $[u_t, v_t] \in [-2, 2]^2 \backslash \{(x, y) \mid -1 < x < 1, \ -1 < y < 1\}\}$. The cost function is $\mathcal{J}(\phi(x_0, u)) = \lim_{k\to\infty} 1/k \sum_{t\in[k-1]} \|u_t\|_1$. The goal is to drive the robot from an initial point $x_0 = (-2.5, -2.5)$ to region $A = \{(x, y) \mid -4 \leq x \leq -3, 3 \leq y \leq 4\}$ and $B = \{(x, y) \mid 3 \leq x \leq 4, -4 \leq y \leq -3\}$. The robot is required to visit region A and B infinitely often, and always avoid region $X = \{(x, y) \mid -1 \leq x \leq 1, -1 \leq y \leq 1\}$. The property of this example is represented in Figure 1a (not to scale) and as a Büchi automaton in Figure 1b. The algorithm is implemented on a 8×8 uniform grid on the states. The input, u, is partitioned into 4×4 uniform intervals.

Algorithm 1 and 2 are implemented in Python 2.7. A Python package, NetworkX, is used to represent the graph structures that arise in solving Algorithm 2, and the Parma Polyhedra Library [23] is used to represent the polyhedral sets that arise in the gridding and to solve the linear program problem that arises in the weight computation. The algorithms are implemented on a MacBook Pro with 2.9GHz Intel Core i5 processor, and 16GB RAM.

Figure 3 shows the state trajectory of the robot under the strategy given by Algorithm 1 and 2. The strategy is able to direct the system to move from the initial position at $(-2.5, -2.5)$ to region A and then region B. The cost at the end of time $N = 31$ is 1.33. Since the strategy forms a cycle, the cost when $N \to \infty$ will not be larger than 1.33 which is lower than the cost of the strategy at 1.5 given by the algorithms.

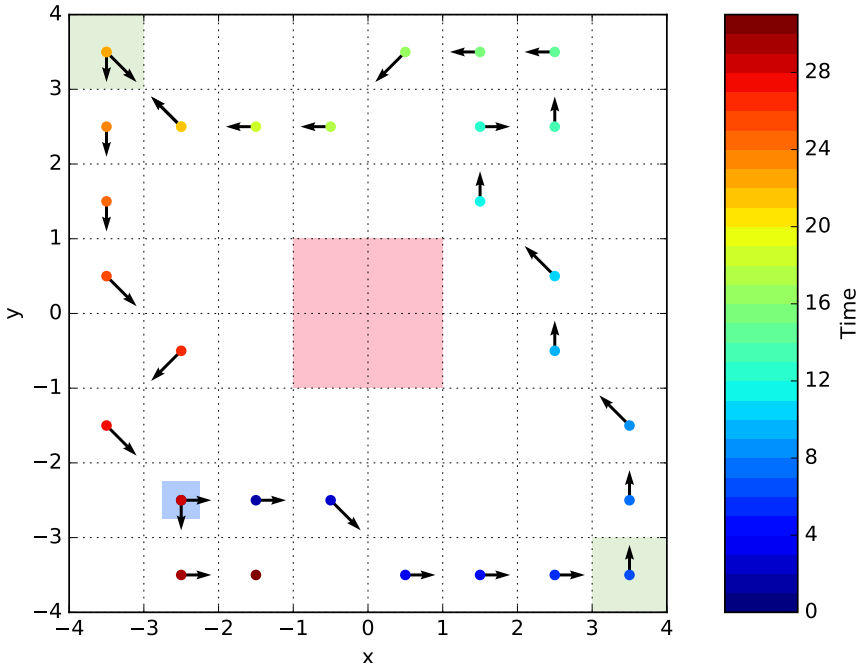


Fig. 3. Simulated results on the linear dynamical system. The blue box indicates the starting position. The arrows represent the control input directions, and the round markers indicates the position from time 0 to time 31 (blue to red).

VI. Conclusion

This paper presents a technique to synthesize optimal controllers for discrete-time piecewise linear systems with ω-regular objectives based on an abstraction-refinement procedure, extending the results in [15]. This method provides a guarantee on the upper bound of the trajectory cost when implementing the suboptimal controller. Furthermore, for systems with a robust optimal controller and a lasso optimal path, the abstraction-refinement procedure converges to the optimal solution. Other future works include extending the results to more complex systems such as nonlinear systems and continuous-time systems, and implementing the technique on more sophisticated examples. To reduce computation time, a more intelligent gridding scheme in the refinement step such as counter-example guided refinement [16] will be developed.

References

[1] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning with deterministic μ-calculus specifications," in *American Control Conference (ACC), 2012*. IEEE, 2012, pp. 735–742.

[2] S. L. Smith, J. Tůumová, C. Belta, and D. Rus, "Optimal path planning for surveillance with temporal-logic constraints," *International Journal of Robotics Research*, vol. 30, no. 14, pp. 1695–1708, 2011.

[3] M. Svoreňová, J. Křetínskỳ, M. Chmelík, K. Chatterjee, I. Černá, and C. Belta, "Temporal logic control for stochastic linear systems using abstraction refinement of probabilistic games," *Nonlinear Analysis: Hybrid Systems*, vol. 23, pp. 230–253, 2017.

[4] E. M. Wolff and R. M. Murray, "Optimal control of nonlinear systems with temporal logic specifications," in *Robotics Research*. Springer, 2016, pp. 21–37.

[5] J. Liu, N. Ozay, U. Topcu, and R. M. Murray, "Synthesis of reactive switching protocols from temporal logic specifications," *IEEE Transactions on Automatic Control*, vol. 58, no. 7, pp. 1771–1785, 2013.

[6] T. Wongpiromsarn, U. Topcu, and R. M. Murray, "Receding horizon temporal logic planning," *IEEE Transactions on Automatic Control*, vol. 57, no. 11, pp. 2817–2830, 2012.

[7] A. Girard, "Controller synthesis for safety and reachability via approximate bisimulation," *Automatica*, vol. 48, no. 5, pp. 947–953, 2012.

[8] M. Mazo and P. Tabuada, "Symbolic approximate time-optimal control," *Systems & Control Letters*, vol. 60, no. 4, pp. 256–263, 2011.

[9] G. Reissig and M. Rungger, "Abstraction-based solution of optimal stopping problems under uncertainty," in *IEEE Int. Conf. on Decision and Control (CDC)*, 2013, pp. 3190–3196.

[10] C. Seatzu, D. Gromov, J. Raisch, D. Corona, and A. Giua, "Optimal control of discrete-time hybrid automata under safety and liveness constraints," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 65, no. 6, pp. 1188 – 1210, 2006.

[11] J. A. DeCastro and H. Kress-Gazit, "Guaranteeing reactive high-level behaviors for robots with complex dynamics," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov 2013, pp. 749–756.

[12] M. Kloetzer and C. Belta, "Temporal logic planning and control of robotic swarms by hierarchical abstractions," *IEEE Transactions on Robotics*, vol. 23, no. 2, pp. 320–330, 2007.

[13] K. Chatterjee and L. Doyen, "Energy parity games," *Theoretical Computer Science*, vol. 458, pp. 49–60, 2012.

[14] K. Chatterjee, T. A. Henzinger, and M. Jurdzinski, "Mean-payoff parity games," in *Logic in Computer Science, 2005. LICS 2005. Proceedings. 20th Annual IEEE Symposium on*. IEEE, 2005, pp. 178–187.

[15] Y. P. Leong and P. Prabhakar, "Optimal control with regular objectives using an abstraction-refinement approach," in *American Controls Conf. (ACC)*, 2016.

[16] E. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith, "Counterexample-guided abstraction refinement," in *International Conference on Computer Aided Verification*. Springer, 2000, pp. 154–169.

[17] E. A. Gol, M. Lazar, and C. Belta, "Temporal logic model predictive control," *Automatica*, vol. 56, pp. 78 – 85, 2015.

[18] S. Karaman, R. G. Sanfelice, and E. Frazzoli, "Optimal control of mixed logical dynamical systems with linear temporal logic specifications," in *IEEE Int. Conf. on Decision and Control (CDC)*, 2008, pp. 2117–2122.

[19] E. M. Wolff, U. Topcu, and R. M. Murray, "Optimal control of non-deterministic systems for a computationally efficient fragment of temporal logic," in *IEEE Int. Conf. on Decision and Control (CDC)*, 2013, pp. 3197–3204.

[20] E. M. Wolff, "Optimal control with weighted average costs and temporal logic specifications," *Proc. of Robotics: Science and Systems VIII*, 2012.

[21] L. Brim, J. Chaloupka, L. Doyen, R. Gentilini, and J.-F. Raskin, "Faster algorithms for mean-payoff games," *Formal methods in system design*, vol. 38, no. 2, pp. 97–118, 2011.

[22] Y. P. Leong and P. Prabhakar, "Abstraction-refinement based optimal control with regular objectives," arXiv:1504.02838.

[23] R. Bagnara, P. M. Hill, and E. Zaffanella, "The Parma Polyhedra Library: Toward a complete set of numerical abstractions for the analysis and verification of hardware and software systems," *Science of Computer Programming*, vol. 72, no. 1–2, pp. 3–21, 2008.

APPENDIX

A. Proof of Theorem 1

We first defines a preorder on the class of transition systems that preserves the optimal cost of winning. In other words, the optimal cost of winning in a system that is higher in the ordering is an upper bound on the optimal cost of winning in a system that is lower in the ordering. For this, we use the definition of alternating simulations with cost from [15]. Then, we show that optimality preservation results extend to infinite trace objectives by proving Theorem 1.

Definition 10. *Given two transition systems $\mathcal{T}_i = (\mathcal{S}_i, \mathcal{S}_i^{init}, \mathcal{U}_i, \mathcal{P}, \Delta_i, \mathcal{L}_i, \mathcal{W}_i)$, for $i = 1, 2$, a simulation from $\mathcal{T}_1$ to $\mathcal{T}_2$ is a pair of relations (α, β), where $\alpha \subseteq \mathcal{S}_1 \times \mathcal{S}_2$ and $\beta \subseteq \mathcal{S}_1 \times \mathcal{U}_1 \times \mathcal{S}_2 \times \mathcal{U}_2$, such that:*

1) $\forall\ (s_1, s_2) \in \alpha,\ \mathcal{L}_1(s_1) = \mathcal{L}_2(s_2)$.
2) $\forall\ s_1 \in \mathcal{S}_1^{init},\ \exists\ s_2 \in \mathcal{S}_2^{init}$ *such that* $(s_1, s_2) \in \alpha$;
3) $\forall\ (s_1, s_2) \in \alpha$ *and* $u_2 \in Enabled(s_2),\ \exists\ u_1 \in Enabled(s_1)$ *such that:*
   a) $(s_1, u_1, s_2, u_2) \in \beta$
   b) $\forall\ (s_1, u_1, s_1') \in \Delta_1,\ \exists\ (s_2, u_2, s_2') \in \Delta_2$ *such that* $(s_1', s_2') \in \alpha$ *and* $\mathcal{W}_1(s_1, u_1, s_1') \leq \mathcal{W}_2(s_2, u_2, s_2')$.

Let $\mathcal{T}_1 \preceq_{(\alpha,\beta)} \mathcal{T}_2$ denote that (α, β) is a simulation from $\mathcal{T}_1$ to $\mathcal{T}_2$. If there exists some (α, β) such that $\mathcal{T}_1 \preceq_{(\alpha,\beta)} \mathcal{T}_2$, then $\mathcal{T}_2$ simulates $\mathcal{T}_1$, and it is denoted as $\mathcal{T}_1 \preceq \mathcal{T}_2$.

Theorem 4 ([15, Theorem 2])**.** *$\preceq$ is a preorder on the class of transition systems over a set of propositions $\mathcal{P}$.*

Previous work [15, Theorem 3] shows that $\preceq$ is an ordering on the transition systems which "preserves" optimal control for transition systems with finite paths. Here, the result is extended to transition systems with infinite paths.

Proof of Theorem 1. Let σ_2 be a strategy for $\mathcal{T}_2$ and s_0'. In addition, define a partial mapping $G : Paths_f(\mathcal{T}_1) \to Paths_f(\mathcal{T}_2)$ such that the domain of G is the set of all paths from s_0 that conform to σ_1, and for any path ζ_1 in the domain of G, $\mathcal{L}_1(\zeta_1) = \mathcal{L}_2(G(\zeta_1))$, and $\mathcal{W}_1(\zeta_1) \leq \mathcal{W}_2(G(\zeta_1))$. This construction ensures that if σ_2 is winning from s_0' with respect to Π, then so is σ_1 from s_0 and $\mathcal{W}_1(\mathcal{T}_1, \sigma_1, s_0) \leq \mathcal{W}_2(\mathcal{T}_2, \sigma_2, s_0')$. We also ensure that if $G(\zeta_1) = \zeta_2$, then $(s_k, s_k') \in \alpha$, where s_k and s_k' are the states of ζ_1 and ζ_2, respectively.

Next, define σ_1 and G by induction. Set $G(s_0) = s_0'$. Suppose σ_1 for paths of length $k-1$ and G for paths of length k, are defined such that the invariant holds. Let $\zeta_1 = s_0 u_0 s_1 \dots s_k$ conform to σ_1. Then, $G(\zeta_1)$ is defined. Let $G(\zeta_1) = s_0' u_0' s_1' \dots s_k'$ and $(s_k, s_k') \in \alpha$. Then, $\sigma_2(G(\zeta_1)) = u_k'$. From the second condition of simulation, there exists u_k such that $(s_k, u_k, s_k', u_k') \in \beta$. Choose $\sigma_1(\zeta_1) = u_k$. For any $\zeta_2 = s_0 u_0 s_1 \dots s_{k+1}$, define $G(\zeta_2) = s_0' u_0' s_1' \dots s_{s+1}'$ such that $(s_{k+1}, s_{k+1}') \in \alpha$ and $\mathcal{W}_1(s_k, u_k, s_{k+1}) \leq \mathcal{W}_2(s_k', u_k', s_{k+1}')$. This construction satisfies the inductive invariant.

B. Proof of Theorem 2

This section proves Theorem 2. First, we show that the projection of the winning strategy $\sigma_{\mathcal{A}}$ of the product system $\mathcal{A}_{\mathcal{T},\mathcal{B}}$ onto $\mathcal{T}$ is a winning strategy for $\mathcal{T}$.

Lemma 1. *Let $\sigma_{\mathcal{A}}$ be a winning strategy of $\mathcal{A}_{\mathcal{T},\mathcal{B}}$ for the state (s_0, q_0) with respect to $\Pi_{\mathcal{T},\mathcal{B}}$. Then, $\sigma = \mathit{Proj}^{\sigma}_{\mathcal{T}}(\sigma_{\mathcal{A}})$ is a winning strategy of $\mathcal{T}$ for the state s_0 with respect to $\mathcal{L}(\mathcal{B})$.*

Proof. By definition of $\mathit{Proj}^{\sigma}_{\mathcal{T}}(\sigma_{\mathcal{A}})$, for any $\zeta \in \mathit{Paths}_{\sigma}(\mathcal{T}, s_0)$, there exists a $\zeta_{\mathcal{A}}$ such that $\zeta = \mathit{Proj}_{\mathcal{T}}(\zeta_{\mathcal{A}})$ and $\zeta_{\mathcal{A}} \in \mathit{Paths}_{\sigma_{\mathcal{A}}}(\mathcal{A}_{\mathcal{T},\mathcal{B}}, (s_0, q_0))$. Let $\zeta \in \mathit{Paths}_{\sigma}(\mathcal{T}, s_0)$, and consider the $\zeta_{\mathcal{A}}$ that satisfies $\zeta = \mathit{Proj}_{\mathcal{T}}(\zeta_{\mathcal{A}})$. Build $r = q_0 q_1 \ldots$ from $\zeta_{\mathcal{A}}$. By definition of $\mathcal{A}_{\mathcal{T},\mathcal{B}}$, $(q_i, \mathcal{L}(s_i), q_{i+1}) \in E$ and $\mathit{Tr}(\zeta) = \mathcal{L}(s_0)\mathcal{L}(s_1)\ldots$. Hence, r is a run of $\mathcal{B}$ over the sequence $\mathit{Tr}(\zeta)$. Because $\sigma_{\mathcal{A}}$ is winning, $\mathit{Inf}(\mathit{Tr}(\zeta_{\mathcal{A}})) \cap \{0\} \neq \emptyset$, and there are infinitely many (s_i, q_i) in $\zeta_{\mathcal{A}}$ such that $q_i \in F$. Consequently, there are infinitely many q_i in r such that $q_i \in F$, and $\mathit{Inf}(r) \cap F \neq \emptyset$. Thus, $\mathit{Tr}(\zeta) \in \mathcal{L}(\mathcal{B})$, and σ is a winning strategy for $\mathcal{T}$ with respect to $\mathcal{B}$. □

Proof of Theorem 2. First, Lemma 1 implies that if a winning strategy of $\mathcal{A}_{\mathcal{T},\mathcal{B}}$ with respect to $\Pi_{\mathcal{T},\mathcal{B}}$ exists, a winning strategy for $\mathcal{T}$ with respect to $\mathcal{L}(\mathcal{B})$ exists.

To prove the converse, we first construct a strategy $\sigma_{\mathcal{A}}$ for $\mathcal{A}_{\mathcal{T},\mathcal{B}}$. Let σ be the winning strategy for $\mathcal{T}$ with respect to $\mathcal{B}$. Consider any path $\zeta \in \mathit{Paths}_{\sigma}(\mathcal{T}, s_0)$ and let $\zeta = s_0 u_0 s_1 u_1 \ldots$. Because σ is winning, there exists a run $r = q_0 q_1 \ldots$ of $\mathcal{B}$ over $\mathit{Tr}(\zeta)$ such that $(q_i, \mathcal{L}(s_i), q_{i+1}) \in E$ and $\mathit{Inf}(\mathit{Tr}(r)) \cap F \neq \emptyset$. Construct a path $\zeta_{\mathcal{A}}$ in $\mathcal{A}_{\mathcal{T},\mathcal{B}}$ from r and ζ such that $\zeta_{\mathcal{A}} = (s_0, q_0)(u_0, e_0)(s_1, q_1)\ldots$ where $e_i = (q_i, \mathcal{L}(s_i), q_{i+1})$. Let $\zeta^i_{\mathcal{A}} = (s_0, q_0)(u_0, e_0)\ldots(s_i, q_i)$ be a prefix of $\zeta_{\mathcal{A}}$, and $\zeta^i = s_0 u_0 \ldots s_i u_i$ be a prefix of ζ. Set $\sigma_{\mathcal{A}}(\zeta^i_{\mathcal{A}}) = (\sigma(\zeta^i), e_i)$.

Next, we show that $\sigma_{\mathcal{A}}$ is winning. For any $\zeta_{\mathcal{A}} \in \mathit{Paths}_{\sigma_{\mathcal{A}}}(\mathcal{A}_{\mathcal{T},\mathcal{B}}, (s_0, q_0))$, there exists a ζ such that $\zeta = \mathit{Proj}_{\mathcal{T}}(\zeta_{\mathcal{A}})$ and $\zeta \in \mathit{Paths}_{\sigma}(\mathcal{T}, s_0)$ by the construction of $\sigma_{\mathcal{A}}$. Hence, $\mathit{Inf}(\mathit{Tr}(r)) \cap F \neq \emptyset$ where $r = q_0 q_1 \ldots$, and there exists some $q \in F$ such that (s, q) where $s \in \mathcal{S}$ occurs infinitely often in $\zeta_{\mathcal{A}}$. So, $\mathit{Inf}(\mathit{Tr}(\zeta_{\mathcal{A}})) \cap \{0\} \neq \emptyset$, and thus, $\sigma_{\mathcal{A}}$ is a winning strategy of $\mathcal{A}_{\mathcal{T},\mathcal{B}}$.

Lastly, we show that the cost is preserved between $\sigma_{\mathcal{A}}$ and σ that is $\mathcal{W}(\mathcal{A}_{\mathcal{T},\mathcal{B}}, \sigma_{\mathcal{A}}, (s_0, q_0)) = \mathcal{W}(\mathcal{T}, \sigma, s_0)$ where $\sigma = \mathit{Proj}^{\sigma}_{\mathcal{T}}(\sigma_{\mathcal{A}})$. For all $\zeta_{\mathcal{A}} \in \mathit{Paths}_{\sigma_{\mathcal{A}}}(\mathcal{A}_{\mathcal{T},\mathcal{B}}, (s_0, q_0))$, there exists a ζ such that $\zeta = \mathit{Proj}_{\mathcal{T}}(\zeta_{\mathcal{A}})$ and $\zeta \in \mathit{Paths}_{\sigma}(\mathcal{T}, s_0)$. Note that the transition cost is preserved such that $\mathcal{W}_{\mathcal{A}}((s_i, q_i), (u_i, e_i), (s_{i+1}, q_{i+1})) = \mathcal{W}(s_i, u_i, s_{i+1})$. Thus, $\mathcal{W}_{\mathcal{A}}(\zeta_{\mathcal{A}}) = \mathcal{W}(\zeta)$, and $\mathcal{W}(\mathcal{A}_{\mathcal{T},\mathcal{B}}, \sigma_{\mathcal{A}}, (s_0, q_0)) \leq \mathcal{W}(\mathcal{T}, \sigma, s_0)$. Similarly, for any $\zeta \in \mathit{Paths}_{\sigma}(\mathcal{T}, s_0)$, there exists a $\zeta_{\mathcal{A}}$ such that $\zeta = \mathit{Proj}_{\mathcal{T}}(\zeta_{\mathcal{A}})$ and $\zeta \in \mathit{Paths}_{\sigma_{\mathcal{A}}}(\mathcal{A}_{\mathcal{T},\mathcal{B}}, (s_0, q_0))$. Thus, $\mathcal{W}(\mathcal{A}_{\mathcal{T},\mathcal{B}}, \sigma_{\mathcal{A}}, (s_0, q_0)) \geq \mathcal{W}(\mathcal{T}, \sigma, s_0)$. As a result, $\mathcal{W}(\mathcal{A}_{\mathcal{T},\mathcal{B}}, \sigma_{\mathcal{A}}, (s_0, q_0)) = \mathcal{W}(\mathcal{T}, \sigma, s_0)$.

C. Proof of Theorem 3

This section proves Theorem 3. First, we show that when initial state and inputs have a bounded deviation from that of the optimal trajectory, the trajectory will have a bounded deviation from the optimal trajectory.

Lemma 2. *There exist bounds $M_x > 0$ and $M_u > 0$ and constants $c_1, c_2 \geq 0$ that depend on $\mathit{PS}(\phi(x^*_0, \mathbf{u}^*))$, such that for all $\varepsilon_x \in [0, M_x]$ and $\varepsilon_u \in [0, M_u]$, if $x_0 \in \mathcal{B}_{\varepsilon_x}(x^*_0)$ and $u_t \in \mathcal{B}_{\varepsilon_u}(u^*_t)$ $\forall t \in \mathbb{N}$, where $\mathbf{u} = \{u_t\}_{t\in\mathbb{N}}$ and $\phi(x_0, \mathbf{u}) = \{x_t\}_{t\in\mathbb{N}}$, then for all $t \in \mathbb{N}$,*

$$\left\|x_{t+1} - x^*_{t+1}\right\|_{\infty} \leq c_1\varepsilon_x + c_2\varepsilon_u,$$

$$\mathit{PS}(\phi(x_0, \mathbf{u})) = \mathit{PS}(\phi(x^*_0, \mathbf{u}^*)).$$

Proof. For $x_p^* x_k^*$ (*i.e.* $t \in [k]$), the lemma is true by [22, Lemma 7] because it is a finite length trajectory. Let c_1', c_2', M_x' and M_u' be the associated bounds for $x_p^* x_k^*$. When $t = k + m$, by robustness property, there exists a constant $L < 1$, and bounds $M_x'' \leq c_1' \varepsilon_x + c_2' \varepsilon_u$, and $M_u'' > 0$ such that $\|x_{k+m} - x_k^*\|_\infty \leq L \|x_k - x_k^*\|_\infty$. Let $R_k = \{x \mid x \in \mathcal{B}_{M_x''}(x_k^*)\}$, and $R_{k+j+1} = \{x \mid x = A_{k+j}x' + B_{k+j}u, x' \in R_{k+j}, u \in \mathcal{B}_{M''u}(u_{k+j}^*)\}$ for all $j \in \mathbb{N}$. The neighborhoods $\{R_{k+j+1}\}_{j\in\mathbb{N}}$ are compact because the transitions are linear (*i.e.* continuous). Furthermore, by robustness property, $R_{k+m} \subset R_k$. Hence, for $j \in [m-1]$ and $i \in \mathbb{N}$, $R_{j+im+k} \subset R_{j+k}$. As a result, x_{j+im+k} is an element in R_{k+j} for all $j \in [m-1]$ and $i \in \mathbb{N}$. Let $d_t = \max\{\|x - x_t^*\|_\infty \mid x \in R_t\}$ be the maximum distance between a state in the neighborhood R_t and the optimal state x_t^* for all $t \geq k$. This maximum is well defined for all $t \geq k$ because $x_t^* \in R_t$. Now, let $\bar{d}$ be the maximum among all d_{k+j} for $j \in [m-1]$. Then, $\|x_t - x_t^*\| \leq \bar{d}$ for all $t \geq k$. Finally, set $M_x = \min\{M_x', M_x''\}$, $M_u = \min\{M_u', M_u''\}$, and c_1 and c_2 such that $c_1\varepsilon_x + c_2\varepsilon_u \geq \max\{\bar{d}, c_1'\varepsilon_x + c_2'\varepsilon_u\}$. □

Lemma 2 ensures that the error from the optimal state at any time is bounded linearly by the error from the initial state and the largest error of control inputs from the optimal ones. The state error decreases to zero when ε_x and ε_u decrease to zero. The constants c_1 and c_2 which depend on t would not be unbounded by the robustness property. The main consequence of a bounded deviation is that the suboptimal cost of this trajectory is also bounded, showed next.

Lemma 3. *Given the cost function in Problem 2, there exist bounds $M_x > 0$ and $M_u > 0$, and constants $c_3, c_4 \geq 0$ such that for all $\varepsilon_x \in [0, M_x]$ and $\varepsilon_u \in [0, M_u]$, if $x_0 \in \mathcal{B}_{\varepsilon_x}(x_0^*)$ and $u_t \in \mathcal{B}_{\varepsilon_u}(u_t^*)$ $\forall t \in \mathbb{N}$,*

$$|\mathcal{W}(\zeta) - \mathcal{W}(\zeta^*)| \leq c_3\varepsilon_x + c_4\varepsilon_u,$$

$$PS(\phi(x_0, \mathbf{u})) = PS(\phi(x_0^*, \mathbf{u}^*)),$$

where $\phi(x_0, \mathbf{u}) = \{x_t\}_{t\in\mathbb{N}}$, and $\zeta = \{x_t u_t\}_{t\in\mathbb{N}}$.

Proof. First, compute

$$\begin{aligned}
&|\mathcal{W}(\zeta) - \mathcal{W}(\zeta^*)| \\
&\quad = \left| \lim_{k\to\infty} \frac{1}{k} \sum_{t=0}^{k-1} \mathcal{J}(x_{t+1}, u_t) - \lim_{k\to\infty} \frac{1}{k} \sum_{t=0}^{k-1} \mathcal{J}(x_{t+1}^*, u_t^*) \right| \\
&\quad = \left| \lim_{k\to\infty} \frac{1}{k} \sum_{t=0}^{k-1} \mathcal{J}(x_{t+1}, u_t) - \mathcal{J}(x_{t+1}^*, u_t^*) \right| \\
&\quad \leq \lim_{k\to\infty} \frac{1}{k} \sum_{t=0}^{k-1} \left| \mathcal{J}(x_{t+1}, u_t) - \mathcal{J}(x_{t+1}^*, u_t^*) \right|.
\end{aligned}$$

The second equality holds because $\mathcal{J}(x_{t+1}, u_t)/k$ forms a convergent series for a lasso path. Since $\mathcal{J}(x, u)$ is a continuous function, there exists a constant $C_t > 0$ such that

$$|\mathcal{J}(x_{t+1}, u_t) - \mathcal{J}(x_{t+1}^*, u_t^*)| \leq C_t \|\bar{x}_t - \bar{x}_t^*\|_\infty.$$

Hence,

$$|\mathcal{W}(\zeta) - \mathcal{W}(\zeta^*)| \leq \lim_{k\to\infty} \frac{1}{k} \sum_{t=0}^{k-1} C \left\|\bar{x}_t - \bar{x}_t^*\right\|_\infty$$

where $C = \sup_{t\in\mathbb{N}_+} C_t$ and $\bar{x}_t = [x_{t+1}, u_t] \in \mathbb{R}^{n+p}$ is a joined vector of x and u. Given that the optimal input is robust,

$$\begin{aligned}|\mathcal{W}(\zeta) - \mathcal{W}(\zeta^*)| &\leq \lim_{k\to\infty} \frac{1}{k} \sum_{t=0}^{k-1} C \max\{c_1\epsilon_x + c_2\epsilon_u, \epsilon_u\} \\ &= C \max\{c_1\epsilon_x + c_2\epsilon_u, \epsilon_u\}.\end{aligned}$$

Then,

$$|\mathcal{W}(\zeta) - \mathcal{W}(\zeta^*)| \leq c_3\epsilon_x + c_4\epsilon_u$$

$$c_3 = \begin{cases} c_1 C, & c_1\epsilon_x + c_2\epsilon_u \geq \epsilon_u \\ 0, & c_1\epsilon_x + c_2\epsilon_u < \epsilon_u \end{cases}$$

$$c_4 = \begin{cases} c_2 C, & c_1\epsilon_x + c_2\epsilon_u \geq \epsilon_u \\ C, & c_1\epsilon_x + c_2\epsilon_u < \epsilon_u \end{cases}$$

□

Lemma 3 implies that there exists a small neighborhood of the optimal trajectory in which the trajectories will go through the same partition sequence and difference in the cost is bounded and decreases to zero if ε_x and ε_u decrease to zero. Given a specific cost sub-optimality, we will now show that there exists a strategy that satisfies this cost error. In other words, we can construct an abstraction to give a chain lasso strategy that satisfies a certain cost error bound.

Lemma 4. *Given any $\delta > 0$, there exists a chain lasso winning strategy σ for some $\hat{\mathcal{T}} = Abs(\mathcal{T}_\mathcal{D}, \equiv_X, \equiv_U)$ such that*

$$|\mathcal{W}(\hat{\mathcal{T}}, \sigma, x_0') - \mathcal{W}(\mathcal{T}_\mathcal{D}, x_0^*, \Pi)| \leq \delta$$

where $x_0' = [x_0^*]_{\equiv_X}$.

Proof. The proof constructs the chain part of the chain lasso strategy followed by the loop part. For each part, we find neighborhoods N_t^x around x_t^* and N_t^u around u_t^* such that N_t^x is contained in the region of the partition containing x_t^*, all transitions from N_t^x and N_t^u lead to N_{t+1}^x and the neighborhoods N_t^x and N_t^u are contained in $B_{M_x}(x_t^*)$ and $B_{M_u}(u_t^*)$. We also bound the maximum cost of any transition from N_t^x to N_{t+1}^x using an input from N_t^u. Lastly, we obtain a chain lasso strategy in $\hat{\mathcal{T}} = Abs(\mathcal{T}_\mathcal{D}, \equiv_X, \equiv_U)$ by choosing N_t^x and N_t^u to be regions of $\equiv_X$ and $\equiv_Y$.

More precisely, let $PS(\phi(x_0^*, \mathbf{u}^*)) = \{P_{i_t}\}_{t\in\mathbb{N}}$. By robustness of the optimal control (Definition 8), we first choose N_k^x to be a grid cell of size $\varepsilon_0/2^j \leq M_x$ for $j \in \mathbb{N}$ that contains an open ball around x_k^* which is contained in P_{i_k}. Next, construct the chain part of the chain lasso strategy inductively, starting from $t = k$ and moving backwards similar to [22, Lemma 9]. Assume $N_{t+1}^x, N_{t+1}^u, \ldots, N_k^x$ are computed. Let $N_t^x \subseteq \mathcal{B}_{M_x}(x_t^*)$ and $N_t^u \subseteq \mathcal{B}_{M_u}(u_t^*)$.

Then, all the transitions from N_t^x on N_t^u will end in N_{t+1}^x. By induction, under this construction, all executions from N_0^x will be in N_t^x after t steps. For all $t \in [k-1]$, choose the N_t^x and N_t^u such that they correspond to an element of a $\varepsilon_0/2^j$ grid for $j \in \mathbb{N}$.

Now, construct the loop part of the chain lasso strategy inductively. Let $R_k^x = N_k^x$ (white square), and for $t = k+1, \ldots, k+m$, let R_t^x be the region where all transitions from R_{t-1}^x on $R_{t-1}^u \subseteq \mathcal{B}_{M_u}(u_{t-1}^*)$ end in (white circles in Figure 2). Define the ε expansion of a set A as $\mathcal{E}_\varepsilon(A) = A \cup \{y \mid \min_{x \in A} \|y - x\|_\infty \leq \varepsilon\}$. The induction begins from $t = k+m$ and moves backwards until $t = k+1$. At $t = k+m$, ε_{k+m} is chosen such that $E_{k+m} = \mathcal{E}_{\varepsilon_{k+m}}(R_{k+m}^x)$ and $E_{k+m} \subset R_k^x$. The latter is possible because of the second robustness property in Definition 8 for the optimal path. Assume that $E_j = \mathcal{E}_{\varepsilon_j}(R_j^x)$ (blue circles in Figure 2) for $j = t+1, \ldots, k+m$, and $F_j = \mathcal{E}_{\varepsilon_j'}(R_j^x)$ (red circles in Figure 2) for $j = t+1, \ldots, k+m-1$ are computed. Construct E_t and F_t by choosing ε_t' such that all trajectories from F_t on $N_t^u \subseteq R_t^u$ will end in E_{t+1}, and ε_t to be smaller than ε_t'. This construction is possible because of the continuity of the transition Δ. At $t = k$, let $N_k^u \subseteq R_k^u$. For all $t = k, \ldots, k+m-1$, N_t^u is chosen to correspond to an element of a $\varepsilon_0/2^j$ grid for $j \in \mathbb{N}$. Given $\{E_{k+j+1}\}_{j \in [m-2]}$ and $\{F_{k+j+1}\}_{j \in [m-2]}$, construct $\{N_{k+j+1}^x\}_{j \in [m-2]}$ where by each N_{k+j+1}^x is a set of elements of a $\varepsilon_0/2^i$ grid for $i \in \mathbb{N}$ such that $E_t \in N_t^x$ and $N_t^x \in F_t$ (group of squares in Figure 2). Note that i may be different for different j. Then, every transition from N_t^x on N_t^u will end in N_{t+1}^x. Hence, the chain of neighborhoods $\{N_j^x\}_{j \in [k+m-1]}$ and $\{N_j^u\}_{j \in [k+m-1]}$ gives us a chain lasso strategy.

In addition, because $N_t^x \subseteq \mathcal{B}_{\varepsilon_x}(x_0^*)$ and $N_t^u \subseteq \mathcal{B}_{\varepsilon_u}(u_t^*)$ where $\varepsilon_x \in [0, M_x]$ and $\varepsilon_x \in [0, M_u]$, the cost of the strategy is within δ of the optimal cost where $c_3\varepsilon_x + c_4\varepsilon_u \leq \delta$ for some constants c_3 and c_4 as given by Lemma 3. Thus, $|\mathcal{W}(\zeta) - \mathcal{W}(\zeta^*)| \leq \delta$ for any path ζ starting in an ε_x ball around x_0^*.

Finally, define $\equiv_X$ and $\equiv_U$ such that the N_j^x and N_j^u are all equivalence classes of $\mathcal{X}$ and $\mathcal{U}$, respectively. Note that we need to ensure that for any $i, j \in [k+m-1]$, N_j^x is the same as N_i^x or the two are disjoint. Similar conditions hold for N_i^u. These conditions can be easily ensured during the construction by picking small enough $\varepsilon_0/2^j$ for $j \in \mathbb{N}$. □

Algorithm 1 contains only uniform grids with grid size $\varepsilon_0/2^i$. But, the partitions corresponding to the neighborhoods of N^x and N^u given by Lemma 4 may not correspond to an uniform grid. Thus, we next construct a uniform grid by refining the chain lasso strategy obtained from Lemma 4.

Lemma 5. *For a given $\delta > 0$, there exists an $\varepsilon = \varepsilon_0/2^i > 0$, such that $|\mathcal{W}(\mathcal{T}_\varepsilon, x_0^\varepsilon, \Pi) - \mathcal{W}(\mathcal{T}_\mathcal{D}, x_0^*, \Pi)| \leq \delta$, where $x_0^\varepsilon = [x_0^*]_{\equiv_X^\varepsilon}$. Furthermore, there exists a winning strategy σ_ε with cost of winning $\mathcal{W}(\mathcal{T}_\varepsilon, x_0^\varepsilon, \Pi)$.*

Proof. From the proof of Lemma 4, we obtain a sequence of neighborhoods N_t^x and N_t^u which corresponds to a chain lasso strategy, say σ starting from N_0^x. Furthermore, every N_t^x corresponds to an element of *Grid*$(\mathcal{X}, \varepsilon_0/2^{i_t})$ or a collection of elements of *Grid*$(\mathcal{X}, \varepsilon_0/2^{i_t})$ for some i_t. Similarly, every N_t^u corresponds to an element of *Grid*$(\mathcal{U}, \varepsilon_0/2^{j_t})$ for some j_t. Let i be the maximum of the i_ts and j_ts. Then, *Grid*$(\mathcal{X}, \varepsilon_0/2^i)$ refines N_t^x, and *Grid*$(\mathcal{U}, \varepsilon_0/2^i)$ refines N_t^u. Define a strategy σ_ε (not necessarily a chain lasso anymore) for $\mathcal{T}_\varepsilon$ which correspond to following the neighborhoods N_t^x. All the paths in $\mathcal{T}_\varepsilon$ conforming to σ_ε are contained in the neighborhoods N_t^x.

As a result, the cost of σ_ε is bounded by that of σ which is at most δ larger than the optimal cost. Therefore, the optimal cost of $\mathcal{T}_\varepsilon$ is at most δ larger than that of $\mathcal{T}_\mathcal{D}$. □

Proof of Theorem 3. Note that $J_{\varepsilon_0/2^i} \leq J_{\varepsilon_0/2^j}$ for all $i > j$, and for any $\delta > 0$, there exists $\varepsilon = \varepsilon_0/2^i$, such that $|\mathcal{W}(\mathcal{T}_\varepsilon, x_0^\varepsilon, \Pi) - \mathcal{W}(\mathcal{T}_\mathcal{D}, x_0^*, \Pi)| \leq \delta$ by Lemma 5. Because the optimal path is a lasso, the optimal strategy has a finite memory, and thus, the optimality gap ε is zero. As a result, $J_\varepsilon = \mathcal{W}(\mathcal{T}_\varepsilon, x_0^\varepsilon, \Pi)$ and $J_{opt} = \mathcal{W}(\mathcal{T}_\mathcal{D}, x_0^*, \Pi)$ is the optimal cost. Hence, $|J_\varepsilon - J_{opt}| \leq \delta$. Therefore, $J_{\varepsilon_0/2^i}$ converges to J_{opt} as i goes to infinity. Furthermore, for each sub-optimal cost $J_{\varepsilon_0/2^i}$, there exists a suboptimal winning strategy $\sigma_{\varepsilon_0/2^i}$ with cost of winning $J_{\varepsilon_0/2^i}$ by Lemma 5.