Optimal Control of Mixed Logical Dynamical Systems with Long-Term Temporal Logic Specifications

Eric M. Wolff and Richard M. Murray

***Abstract*—We present a mathematical programming-based method for control of large a class of nonlinear systems subject to temporal logic task specifications. We consider Mixed Logical Dynamical (MLD) systems, which include linear hybrid automata, constrained linear systems, and piecewise affine systems. We specify tasks using a fragment of linear temporal logic (LTL) that allows both finite- and infinite-horizon properties to be specified, including tasks such as surveillance, periodic walking, repeated assembly, and environmental monitoring. Our method directly encodes an LTL formula as mixed-integer linear constraints on the MLD system, instead of computing a finite abstraction. This approach is efficient; for common tasks the formulation may use significantly fewer binary variables than related approaches. In simulation, we solve non-trivial temporal logic motion planning tasks for high-dimensional continuous systems using our approach.**

I. Introduction

The responsibilities given to robots and other autonomous systems continues to increase faster than our ability to reason about the correctness of their behavior. In safety-critical applications like autonomous driving, air traffic management, and medical robotics, it is desirable to unambiguously specify the desired system behavior and automatically synthesize a controller that provably implements this behavior. Additionally, realistic applications often require hybrid dynamical models with high-dimensional continuous state-spaces and demand efficient (not just feasible) controllers.

Linear temporal logic (LTL) is an expressive task-specification language for specifying a variety of properties such as responding to the environment (if A, then B), repeatedly visiting goals (repeatedly A and repeatedly B), staying safe (never B), and remaining stable (eventually always A). These properties generalize classical point-to-point motion planning.

Standard methods for motion planning with LTL specifications first create a finite abstraction of the original dynamical system. This abstraction can informally be viewed as a labeled graph. Approximate finite abstractions can be computed using either sampling-based methods (rapidly-exploring random trees and probabilistic roadmaps) [7, 15, 20] or reachability-based approaches for special classes of dynamical systems [2, 4, 13, 18, 29]. Given a finite abstraction of a dynamical system and an LTL specification, feasible controllers can be automatically constructed using an automata-based approach [3, 7, 12, 15, 18]. This approach first transforms the LTL formula into an equivalent non-deterministic Büchi automaton whose size is potentially exponential in the length of the formula [3]. A product automaton is created from the system and the Büchi automaton and then a feasible controller is found by searching in the product automaton. While optimal controllers can be computed for the discrete abstraction [**?** 23, 27], these may be suboptimal with respect to the original dynamical system for a given abstraction.

Instead of the automata-based approach, we directly encode an LTL formula as mixed-integer linear constraints on the original dynamical system. We consider Mixed Logical Dynamical (MLD) systems [5], which can model linear hybrid automata and constrained linear systems. MLD systems are formally equivalent to piecewise affine systems [6]. It is well-known that mixed-integer linear programming can be used for reasoning about propositional logic [8, 14], generating state-constrained trajectories [11, 22, 25], and modeling vehicle routing problems [16, 24]. The work most similar to ours is Karaman et. al. [17], who consider controller synthesis for MLD systems subject to finite-horizon LTL specifications. However, finite-horizon properties are too restrictive to model a large class of interesting robotics problems, including persistent surveillance, repeated assembly, periodic walking, and environmental monitoring. Our work specifically addresses these types of periodic tasks with a novel mixed-integer formulation.

Our two main contributions are 1) a novel method for encoding both finite- and infinite-horizon LTL properties as mixed-integer linear constraints on a MLD system, and 2) an improved representation that uses significantly fewer binary and continuous variables for common tasks as compared to previous work [17]. The fragment of LTL that we consider allows one to specify properties such as safety, stability, liveness, guarantee, and response. We demonstrate how this mixed-integer programming formulation can be used with off-the-shelf optimization solvers (e.g. CPLEX [1]) to compute both feasible and optimal controllers for high-dimensional MLD systems with non-trivial specifications.

II. Preliminaries

We now provide preliminaries on the modeling and specification languages, Mixed Logical Dynamical (MLD) systems and linear temporal logic, respectively, used throughout the paper.

An *atomic proposition* is a statement that has a unique truth value (*True* or *False*). Let $\mathcal{T} = \{0, 1, 2, \ldots, T\} \subset \mathbb{N}$ denote a bounded set of discrete time instances and $\mathcal{T}^\infty = \{0, 1, 2, \ldots\}$ denote an unbounded set of discrete time instances.

The authors are with the Department of Control and Dynamical Systems, California Institute of Technology, Pasadena, CA 91125. The corresponding author is ewolff@caltech.edu

A. Mixed Logical Dynamical systems

We consider Mixed Logical Dynamical (MLD) systems, which were introduced in [5]. These discrete-time systems have both continuous and discrete-valued states and allow one to model nonlinearities, logic, and constraints. Following [6], an MLD system is given by

$$\begin{aligned} x(t+1) &= Ax(t) + B_1 u(t) + B_2 \delta(t) + B_3 z(t) \\ y(t) &= Cx(t) + D_1 u(t) + D_2 \delta(t) + D_3 z(t) \\ \text{subject to } & E_2 \delta(t) + E_3 z(t) \leq E_1 u(t) + E_4 x(t) + E_5, \end{aligned} \quad (1)$$

where $t \in \mathcal{T}^{\infty}$, $x \in \mathbb{R}^{n_c} \times \{0,1\}^{n_l}$ are the continuous and binary states, $u \in \mathbb{R}^{m_c} \times \{0,1\}^{m_l}$ are the inputs, $y \in \mathbb{R}^{p_c} \times \{0,1\}^{p_l}$ are the outputs, and $\delta \in \{0,1\}^{r_l}$ and $z \in \mathbb{R}^{r_l}$ are auxiliary binary and continuous variables, respectively. The terms A, B_1, B_2, B_3, C, D_1, D_2, D_3, E_1, E_2, E_3, E_4, and E_5 are system matrices of appropriate dimension. For notational convenience, let $\mathcal{X} := \mathbb{R}^{n_c} \times \{0,1\}^{n_l}$ and $\mathcal{U} := \mathbb{R}^{m_c} \times \{0,1\}^{m_l}$.

Let AP be a finite set of atomic propositions. Let $L_t : \mathcal{X} \to 2^{AP}$ be the time-dependent *labeling function* which maps each state to the set of atomic propositions that are $True$ at time t. Each atomic proposition $\psi \in AP$ is represented by a union of polyhedrons. The finite index set $\mathcal{I}_t^{\psi}$ lists the polyhedrons where ψ holds at time t. The i-th polyhedron is $\{x \in \mathcal{X} \mid H_t^{\psi_i} x \leq K_t^{\psi_i}\}$, where $i \in \mathcal{I}_t^{\psi}$. Thus, the set of states where atomic proposition ψ holds at time t is given by $[[\psi]](t) := \{x \in \mathcal{X} \mid H_t^{\psi_i} x \leq K_t^{\psi_i} \text{ for some } i \in \mathcal{I}_t^{\psi}\}$. This (potentially) time-varying set is the finite union of polyhedrons (finite conjunctions of halfspaces). As non-convex and non-polyhedron regions can be approximated by unions of polyhedrons, this is a minor limitation.

We assume that the MLD system (1) is well-posed (see Definition 1 in [5]). Thus, for an initial condition x_0 and a control input sequence $\mathbf{u} = u_0 u_1 u_2 \ldots$, there is a unique trajectory (or run) $\mathbf{x} = x_0 x_1 x_2 \ldots$ that satisfies the constraints in (1). A *walk* is a finite sequence of states $\mathbf{x} = x_0 x_1 x_2 \ldots x_N$ that satisfy the constraints in (1). A *cycle* is a walk $\mathbf{x} = x_0 x_1 x_2 \ldots x_{N-1}$ where $x_N = x_0$. A trajectory $\mathbf{x}$ induces a corresponding *word* $L(\mathbf{x}) = L_0(x_0) L_1(x_1) L_2(x_2) \ldots$ through the labeling function. A word is similarly defined for a walk or cycle.

B. Linear temporal logic

We use a fragment of linear temporal logic (LTL) to concisely and unambiguously specify desired system behavior. We begin by defining LTL, from which our fragment will inherit syntax and semantics. A comprehensive treatment of LTL is given in [3].

Syntax: LTL includes: (a) a set of atomic propositions, (b) the propositional connectives: $\neg$ (negation) and $\wedge$ (conjunction), and (c) the temporal operators: $\bigcirc$ (next) and $\mathcal{U}$ (until). Other propositional connectives such as $\vee$ (disjunction) and $\implies$ (implication) and other temporal operators such as $\diamond$ (eventually), $\square$ (always), $\square\diamond$ (infinitely often), and $\diamond\square$ (eventually forever) can be derived.

An LTL formula is defined inductively as follows: (1) any atomic proposition is an LTL formula, and (2) given LTL formulas φ_1 and φ_2, $\neg\varphi_1$, $\varphi_1 \wedge \varphi_2$, $\bigcirc\varphi_1$, and $\varphi_1 \, \mathcal{U} \, \varphi_2$ are LTL formulas.

Semantics: An LTL formula is interpreted over an infinite sequence of states (i.e., a trajectory). Given an infinite sequence of states $\mathbf{x} = x_0 x_1 x_2 \ldots$ and a formula φ, the semantics are defined inductively as follows: (i) for atomic proposition p, $x_i \vDash p$ if and only if (iff) $p \in L(x_i)$; (ii) $x_i \vDash \neg\varphi$ iff $x_i \nvDash \varphi$; (iii) $x_i \vDash \varphi \wedge \psi$ iff $x_i \vDash \varphi$ and $x_i \vDash \psi$; (iv) $x_i \vDash \bigcirc\varphi$ iff $x_{i+1} \vDash \varphi$; and (v) $x_i \vDash \varphi \, \mathcal{U} \, \psi$ iff $\exists j \geq i$ s.t. $x_j \vDash \psi$ $\forall k \in [i, j), x_k \vDash \varphi$.

Let φ be an LTL formula. Then, $\square\varphi$ holds at position i iff φ holds at every position in σ starting at position i, $\diamond\varphi$ holds at position i iff φ holds at some position $j \geq i$ in σ, $\square \diamond \varphi$ holds at position i iff φ holds at infinitely many positions $j \geq i$ in σ, and $\diamond\square\varphi$ holds at position i iff there exists some position $j \geq i$ such that φ holds at every position in σ starting at position j.

A *propositional formula* ψ is composed of only atomic propositions and propositional connectives. We denote the set of states where ψ holds by $[[\psi]]$.

An infinite sequence of states $\mathbf{x} = x_0 x_1 x_2 \ldots$ *satisfies* the LTL formula φ, denoted by $\mathbf{x} \vDash \varphi$, if $x_0 \vDash \varphi$. Let $\mathcal{M}^{\mathbf{u}}(x_0)$ denote the unique trajectory of $\mathcal{M}$ from initial state x_0 under the control input sequence $\mathbf{u}$. The system $\mathcal{M}$ *satisfies* the LTL formula φ at state $x_0 \in \mathcal{X}$ if and only if there exists a control input sequence $\mathbf{u}$ such that $\mathcal{M}^{\mathbf{u}}(x_0) \vDash \varphi$.

C. An expressive fragment of LTL

We do not attempt to reason about all possible LTL formulas; instead, we develop a useful library of temporal operators for robotic tasks. This fragment of LTL can specify a wide range motion planning tasks such as safe navigation, surveillance, persistent coverage, response to the environment, and visiting goals. The semantics of the operators are inherited from that of the full LTL as defined in Section II-B. In the following equations, ψ, ϕ, and ψ_j (for a finite number of indices j) are propositional formulas. To simplify the presentation, we loosely split these into three groups of functionality: core Φ_{core}, response Φ_{resp}, and fairness Φ_{fair}.

The core operators, $\Phi_{\text{core}} := \{\varphi_{\text{safe}}, \varphi_{\text{goal}}, \varphi_{\text{per}}, \varphi_{\text{live}}\}$, specify fundamental properties such as safety, guarantee, persistence, and recurrence. These operators are,

$$\varphi_{\text{safe}} := \square\psi, \quad (2)$$

$$\varphi_{\text{goal}} := \diamond\psi, \quad (3)$$

$$\varphi_{\text{per}} := \diamond \square \psi, \quad (4)$$

$$\varphi_{\text{live}} := \square \diamond \psi, \quad (5)$$

where φ_{safe} specifies safety, i.e., a property should invariantly hold, φ_{goal} specifies visiting goals, i.e., a property should eventually hold, φ_{per} specifies persistence, i.e., a property should eventually hold invariantly, as in steady-state behavior or stability, and φ_{live} specifies liveness (recurrence), i.e., a property should hold infinitely often, as in surveillance.

The response operators, $\Phi_{\text{resp}} := \{\varphi_{\text{resp}}^1, \varphi_{\text{resp}}^2, \varphi_{\text{resp}}^3, \varphi_{\text{resp}}^4\}$, specify how the system responds to the environment. These operators are,

$$\varphi^1_{\text{resp}} := \Box(\psi \implies \bigcirc\phi), \tag{6}$$
$$\varphi^2_{\text{resp}} := \Box(\psi \implies \Diamond\phi), \tag{7}$$
$$\varphi^3_{\text{resp}} := \Diamond\Box(\psi \implies \bigcirc\phi), \tag{8}$$
$$\varphi^4_{\text{resp}} := \Diamond\Box(\psi \implies \Diamond\phi), \tag{9}$$

where φ^1_{resp} specifies next-step response to the environment, φ^2_{resp} specifies eventual response to the environment, φ^3_{resp} specifies steady-state next-step response to the environment, and φ^4_{resp} specifies steady-state eventual response to the environment.

Finally, the fairness operators, $\Phi_{\text{fair}} := \{\varphi^1_{\text{fair}}, \varphi^2_{\text{fair}}, \varphi^3_{\text{fair}}\}$, allow one to specify conditional tasks. These operators are,

$$\varphi^1_{\text{fair}} := \Diamond\psi \implies \bigwedge_{j=1}^{m} \Diamond\phi_j, \tag{10}$$
$$\varphi^2_{\text{fair}} := \Diamond\psi \implies \bigwedge_{j=1}^{m} \Box\Diamond\phi_j, \tag{11}$$
$$\varphi^3_{\text{fair}} := \Box\Diamond\psi \implies \bigwedge_{j=1}^{m} \Box\Diamond\phi_j, \tag{12}$$

where φ^1_{fair} specifies conditional goal visitation, and φ^2_{fair} and φ^3_{fair} specify conditional repeated goal visitation.

The fragment of LTL that we consider is built from the temporal operators defined above as follows,

$$\varphi ::= \varphi_{\text{core}} \mid \varphi_{\text{resp}} \mid \varphi_{\text{fair}} \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2, \tag{13}$$

where $\varphi_{\text{core}} \in \Phi_{\text{core}}$, $\varphi_{\text{resp}} \in \Phi_{\text{resp}}$, and $\varphi_{\text{fair}} \in \Phi_{\text{fair}}$.

This LTL fragment was designed to specify many properties relevant to robotic motion planning, especially for surveillance tasks for which no mathematical programming-based approaches exist. However, it does not include properties of the form $\varphi_1\ \mathcal{U}\ \varphi_2$ (until) or nested properties. It is future work to determine all temporal properties that can be expressed within a similar framework.

Remark 1. The following standard identities [3] can be used to simplify the resulting expressions via (i) conjunctions: $\Box\varphi_1 \wedge \Box\varphi_2 = \Box(\varphi_1 \wedge \varphi_2)$, $\Diamond\Box\varphi_1 \wedge \Diamond\Box\varphi_2 = \Diamond\Box(\varphi_1 \wedge \varphi_2)$, (ii) disjunctions: $\Diamond\varphi_1 \vee \Diamond\varphi_2 = \Diamond(\varphi_1 \vee \varphi_2)$, $\Box\Diamond\varphi_1 \vee \Box\Diamond\varphi_2 = \Box\Diamond(\varphi_1 \vee \varphi_2)$, and (iii) negations: $\neg\Box p = \Diamond\neg p$, $\neg\Diamond p = \Box\neg p$, $\neg\Box\Diamond p = \Diamond\Box\neg p$, and $\neg\Diamond\Box p = \Box\Diamond\neg p$.

III. Problem statement

In this section, we formally state both a feasibility and an optimization problem and give an overview of our solution approach. Let $\mathcal{M}$ be a MLD system of the form (1) and φ be an LTL formula of the form (13) defined over AP.

The first problem is determining whether or not $\mathcal{M}$ satisfies the given LTL specification.

Problem 1. Given an MLD system $\mathcal{M}$ of the form (1), with initial condition x_0, and an LTL formula φ of the form (13), determine whether or not there exists a control input sequence $\mathbf{u}$ such that $\mathcal{M}_{\mathbf{u}}(x_0) \models \varphi$.

We now introduce a cost function to distinguish among all trajectories that satisfy Problem 1. Since LTL formulas are defined over infinite state sequences, we define a cost function over infinite state sequences. Let $\bigwedge_{j=1}^{m} \Box\Diamond\phi_j$ be the conjunction of all $\Box\Diamond$ operators in φ, where each ϕ_j is a *repeated task*. A *task cycle* is a trajectory segment that intersects $[\![\phi_j]\!]$ for each $j = 1, \dots, m$ at least once. Similarly to [10], we minimize the average cost per task cycle.

Definition 1. Let $\mathbf{x}$ be a trajectory of $\mathcal{M}$, let $\mathbf{u}$ be the corresponding control input sequence, and let $I_{TC}(t) = 1$ indicate that the system completes a 'task cycle' at time t and $I_{TC}(t) = 0$ otherwise. The *average cost-per-task-cycle* of trajectory $\mathbf{x}$ is

$$J(\mathbf{x}, \mathbf{u}) := \limsup_{n\to\infty} \frac{\sum_{t=0}^{n} c(x_t, u_t)}{\sum_{t=0}^{n} I_{TC}(t)}, \tag{14}$$

where J maps trajectories and control inputs of $\mathcal{M}$ to $\mathbb{R} \cup \infty$.

This cost function is well-defined when (i) $c(x_t, u_t)$ is bounded for all $t \geq 0$, and (ii) there exists a $t' \in \mathbb{N}$ such that $I_{TC}(x_t) = 1$ for infinitely many $t \geq t'$. We assume that (i) is true in the sequel and note that (ii) holds for every trajectory that satisfies an LTL formula φ with at least one $\Box\Diamond\phi_j$ specification. If there are no $\Box\Diamond$ operators in φ, one can add a $\Box\Diamond\, True$ specification so that $I_{TC}(t) = 1$ for all $t \in \mathcal{T}^\infty$. Note that while the cost function $c : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ can be convex, it is typically linear or quadratic in practice.

The second problem is computing a control input sequence that satisfies φ and minimizes J.

Problem 2. Given an MLD system $\mathcal{M}$ of the form (1), with initial condition x_0, and an LTL formula φ of the form (13), compute a control input sequence $\mathbf{u}$ such that $\mathcal{M}_{\mathbf{u}}(x_0) \models \varphi$ and $J(\mathcal{M}_{\mathbf{u}}(x_0), \mathbf{u})$ is minimized.

Remark 2. We only consider open-loop trajectory generation, as this is already a challenging problem due to the nonlinear dynamics and LTL specifications. Disturbances can be dealt with by wrapping a feedback controller around the computed trajectory. Incorporating this information during trajectory generation is the subject of future work.

IV. A sufficient finite trajectory parameterization

In this section, we first note that both Problems 1 and 2 are undecidable via a reduction to the reachability problem for linear hybrid automata. Undeterred, we limit our search to trajectories in a prefix-suffix form that is commonly used in model checking for finite systems. In this structure, the prefix is a finite trajectory and the suffix is a finite trajectory that is repeated infinitely often. This gives us a sufficient condition that is amenable to computation, although we may miss valid trajectories. Finally, we show that the cost function (14) for trajectories in prefix-suffix structure only depends on the repeated suffix, although one can also optimize the prefix (see Remark 3).

A. Undecidability

Unfortunately, even seemingly simple control problems are undecidable for MLD systems. We state known results in Theorem 1 for convenience.

Theorem 1 (see Section 4.4 in [9])**.** *The problem of determining if a discrete-time piecewise affine system can reach the origin from a given initial condition is undecidable.*

Since MLD systems are formally equivalent to piecewise affine systems [6], it follows that reachability is undecidable for MLD systems as well. Since determining if an MLD system satisfies the LTL formula $\varphi = \Diamond\psi$ is undecidable, it follows that Problems 1 and 2 are undecidable as well. Related work achieves decidability by only considering finite-horizon properties [17, 19].

Despite the apparently daunting theoretical barriers to controller synthesis, researchers have successfully used sufficient conditions for related problems, such as creating finite abstractions for piecewise affine systems [4, 13, 29]. In this spirit, we propose a sufficient condition inspired by the theory of LTL model checking for finite systems.

B. A prefix-suffix parameterization

First we define a trajectory in *prefix-suffix* form.

Definition 2. Let $\mathbf{x}_{\text{pre}}$ be a finite walk and $\mathbf{x}_{\text{suf}}$ be a finite cycle of an MLD system. A trajectory $\mathbf{x}$ is in *prefix-suffix* form if it is of the form $\mathbf{x} = \mathbf{x}_{\text{pre}}(\mathbf{x}_{\text{suf}})^{\omega}$, where ω denotes infinite repetition.

In what follows, we require that the (time-varying) labeling function L_t is eventually periodic.

Assumption 1. There exists a finite $t' \in \mathcal{T}^{\infty}$ and a $\Omega \in \mathbb{N}$ such that $L_t = L_{t+\Omega}$ for all $t \geq t' \in \mathcal{T}^{\infty}$. We further assume that Ω is minimal among all possible values.

In the sequel, we will limit our search to trajectories $\mathbf{x} = \mathbf{x}_{\text{pre}}(\mathbf{x}_{\text{suf}})^{\omega}$ in prefix-suffix form. As $\mathbf{x}_{\text{pre}}$ and $\mathbf{x}_{\text{suf}}$ are both finite, this allows the application finite-dimensional optimization techniques. Recall that $\mathbf{x}_{\text{suf}}$ is a cycle, which is enforced by loop closure constraints. The constraint that $\mathbf{x}_{\text{suf}}$ is a cycle allows us to repeat the sequence of states forever. Repeating the same sequence of states is a sufficient condition that the word $L(\mathbf{x}_{\text{suf}})$ (i.e., the sequence of atomic propositions) is also repeated (using Assumption 1). However, only the word $L(\mathbf{x}_{\text{suf}})$ matters for the feasibility of an LTL formula, not the exact sequence of states $\mathbf{x}_{\text{suf}}$. In fact, there may exist other trajectories that produce the same word $L(\mathbf{x}_{\text{suf}})$, but are not periodic. Our approach cannot find such trajectories, although this limitation has not been noticed in our practical experience. This differs from case of finite discrete systems, where a prefix-suffix parameterization is sufficient to find a feasible solution if one exists [3].

We enforce that both $\mathbf{x}_{\text{pre}}$ and $\mathbf{x}_{\text{suf}}$ are valid walks that satisfy the dynamics (1) for some control input sequence. Let $\mathbf{x}_{\text{cat}} := \mathbf{x}_{\text{pre}}\mathbf{x}_{\text{suf}}$ denote the concatenation of $\mathbf{x}_{\text{pre}}$ and $\mathbf{x}_{\text{suf}}$. Assign time indices to $\mathbf{x}_{\text{cat}}$ by $\mathcal{T}_{\text{cat}} := \{0, 1, \ldots, T_s, \ldots, T\}$. Let $\mathcal{T}_{\text{pre}} := \{0, 1, \ldots, T_s-1\}$ and $\mathcal{T}_{\text{suf}} := \{T_s, \ldots, T\}$, where T_s indicates the first time instance on the suffix. The infinite repetition of $\mathbf{x}_{\text{suf}}$ is enforced by the constraint $\mathbf{x}_{\text{cat}}(T_s) = \mathbf{x}_{\text{cat}}(T)$. By Assumption 1, it is sufficient that T_s is greater than t' and that the length of $\mathcal{T}_{\text{suf}}$ is an integer multiple of Ω. When convenient, we identify $\mathbf{x}_{\text{pre}}(0)\cdots\mathbf{x}_{\text{pre}}(T_{\text{pre}})$ with $\mathbf{x}_{\text{cat}}(0)\cdots\mathbf{x}_{\text{cat}}(T_s - 1)$ and $\mathbf{x}_{\text{suf}}(0)\cdots\mathbf{x}_{\text{suf}}(T_{\text{suf}})$ with $\mathbf{x}_{\text{cat}}(T_s)\cdots\mathbf{x}_{\text{cat}}(T)$ in the obvious manner.

We now show that, conditional on a given prefix-suffix parameterization, the cost function only depends on the infinite-horizon behavior of the system.

Lemma 1. *Let $\mathbf{x} = \mathbf{x}_{pre}(\mathbf{x}_{suf})^{\omega}$ be a trajectory in prefix-suffix form. Then, $J(\mathbf{x}, \mathbf{u}) = J((\mathbf{x}_{suf})^{\omega}, \mathbf{u})$.*

Proof: From the assumptions below the cost function (14), all costs are finite and $I_{TC}(t) = 1$ for infinitely many values of t. Let $\mathbf{x}_{\text{pre}} = \mathbf{x}_{\text{pre}}(0)\ldots\mathbf{x}_{\text{pre}}(k-1)$. Thus,

$$\begin{aligned} J(\mathbf{x}, \mathbf{u}) &:= \limsup_{n\to\infty} \frac{\sum_{t=0}^{n} c(x_t, u_t)}{\sum_{t=0}^{n} I_{TC}(t)} \\ &= \limsup_{n\to\infty} \frac{\sum_{t=0}^{k-1} c(x_t, u_t) + \sum_{t=k}^{n} c(x_t, u_t)}{\sum_{t=0}^{k-1} I_{TC}(t) + \sum_{t=k}^{n} I_{TC}(t)} \\ &= \limsup_{n\to\infty} \frac{\sum_{t=k}^{n} c(x_t, u_t)}{\sum_{t=k}^{n} I_{TC}(t)} = J((\mathbf{x}_{\text{suf}})^{\omega}, \mathbf{u}). \end{aligned}$$

■

In the next section, we will write the temporal operators as mixed-integer constraints on $\mathbf{x}_{\text{pre}}$ and $\mathbf{x}_{\text{suf}}$. In Section V-B, we *a priori* select the lengths of the prefix and the suffix. In Section V-C, we automatically select the lengths. In Section V-D, we separate the problem into two sequential optimization problems, which trades completeness (with respect to the given prefix-suffix parameterization) for efficiency.

V. A MIXED-INTEGER LINEAR FORMULATION OF LTL CONSTRAINTS

In this section, we develop mixed-integer programming formulations that solve Problems 1 and 2 with respect to a given prefix-suffix trajectory parameterization, $\mathbf{x}_{\text{cat}} = \mathbf{x}_{\text{pre}}\mathbf{x}_{\text{suf}}$. A trajectory for the MLD system is then implemented as $\mathbf{x} = \mathbf{x}_{\text{pre}}(\mathbf{x}_{\text{suf}})^{\omega}$. Since the system is deterministic, this defines a corresponding control input sequence. The split between $\mathbf{x}_{\text{pre}}$ and $\mathbf{x}_{\text{suf}}$ can either be specified *a priori* or automatically during the optimization. We mix notation in the following and refer to x and $\mathcal{T}$ instead of $\mathbf{x}_{\text{cat}}$ and $\mathcal{T}_{\text{cat}}$ when clear from context. We focus the discussion on Problem 2, which subsumes Problem 1.

Given an LTL formula of the form (13), first rewrite it in disjunctive normal form so that each clause is a conjunction of temporal operators described in Section II-C. Then, the problem can be solved for each clause in parallel. Either the first (feasible) or best (optimal) trajectory is selected. From our experience, tasks are largely composed of conjunctions and thus the number of clauses is typically small. Thus, from now on we assume that a LTL formula is conjunctions of the temporal operators from Section II-C.

The following sections define the necessary mixed-integer linear constraints on the MLD system $\mathcal{M}$. These include dynamical constraints such that $\mathbf{x}_{\text{pre}}$ is a valid walk, $\mathbf{x}_{\text{suf}}$ is a valid cycle, and the initial condition is satisfied, $\mathbf{x}_{\text{pre}}(0) = x_0$. Let $\mathcal{C}_{\mathcal{D}}$ denote the set of all dynamical constraints. The LTL formula φ adds additional logical constraints, the set of which is denoted by $\mathcal{C}_{LTL}$. Once the entire set of dynamical and logical constraints has been formed, the problem can be solved using an off-the-shelf mixed-integer linear (or quadratic) program solver.

Remark 3. As the cost function only depends on $\mathbf{x}_{\text{suf}}$, the prefix $\mathbf{x}_{\text{pre}}$, although feasible, might be unacceptable to a user. Thus, one can optimize a prefix $\mathbf{x}_{\text{pre}}$ with an optimal $\mathbf{x}_{\text{suf}}$ as a terminal boundary condition. We show how to do this later in Section V-D.

A. Relating the dynamics and propositions

We now relate the state of a MLD system to the set of atomic propositions that are $True$ at each time instance. We assume that each propositional formula ψ is described at time t by the union of a finite number of polytopes, indexed by the finite index set $\mathcal{I}_t^{\psi}$. Let $[[\psi]](t) := \{x \in \mathcal{X} \mid H_t^{\psi_i} x \leq K_t^{\psi_i}$ for some $i \in \mathcal{I}_t^{\psi}\}$ represent the set of states that satisfy propositional formula ψ at time t. We assume that these have been constructed as necessary from the system's original atomic propositions.

For each propositional formula ψ, introduce binary variables $z_t^{\psi_i} \in \{0,1\}$ for all $i \in \mathcal{I}_t^{\psi}$ and for all $t \in \mathcal{T}$. Let x_t be the state of the system at time t, M be a sufficiently large constant, and $\mathbf{1} := (1,1,\dots,1)^T$ is a vector of ones of appropriate dimension. Then,

$$H_t^{\psi_i} x_t \leq K_t^{\psi_i} + M(1 - z_t^{\psi_i})\mathbf{1}, \quad \forall i \in \mathcal{I}_t^{\psi} \tag{15}$$

$$\sum_{i \in \mathcal{I}_t^{\psi}} z_t^{\psi_i} = 1 \tag{16}$$

enforces the constraint that $x_t \in [[\psi]](t)$ at time t. Define $P_t^{\psi} := \sum_{i\in\mathcal{I}_t^{\psi}} z_t^{\psi_i} \in \mathbb{N}$. If $P_t^{\psi} = 1$, then $x_t \in [[\psi]](t)$. If $P_t^{\psi} = 0$, then nothing can be inferred.

Note that we only use one binary variable for each polyhedron (i.e., finite conjunction of halfspaces). This compares favorably with the approach in [17], where a binary variable is introduced for each halfspace. Additionally, as we use implication, the additional continuous variables and mixed-integer constraints previously used are not needed. For simple tasks such as $\varphi = \diamond\psi$, our method can use significantly fewer binary variables than previously needed, depending on the number of halfspaces and polytopes needed to describe ψ.

For every temporal operator described in the following sections, the big-M constraints in (15) should be understood to be implicitly applied to the corresponding propositional formulas so that $P_t^{\psi} = 1$ implies that the system satisfies ψ at time t.

B. Simultaneous solution with a priori splitting

In this section, the trajectory parameterization, $\mathbf{x}$, has been *a priori* split into a prefix $\mathbf{x}_{\text{pre}}$ and a suffix $\mathbf{x}_{\text{suf}}$. This assumption will be relaxed in Section V-C. We further assume that $\mathbf{x}_{\text{pre}}$ and $\mathbf{x}_{\text{suf}}$ satisfy Assumption 1.

To be able to repeat the suffix $\mathbf{x}_{\text{suf}}$ infinitely often, $\mathbf{x}_{\text{suf}}$ must be a cycle. Let $\mathbf{x}_{\text{suf}}(0)$ and $\mathbf{x}_{\text{suf}}(T_{\text{suf}})$ be the first and last terms in $\mathbf{x}_{\text{suf}}$ respectively. Then, the constraint $\mathbf{x}_{\text{suf}}(0) = \mathbf{x}_{\text{suf}}(T_{\text{suf}}+1)$ subject to the dynamic constraints of (1) ensures that $\mathbf{x}_{\text{suf}}$ is indeed a cycle.

In the following, the correctness of the constraints applied to $\mathbf{x}_{\text{pre}}$ and $\mathbf{x}_{\text{suf}}$ comes directly from the LTL semantics given in Section II-B and the form of the trajectory $\mathbf{x} = \mathbf{x}_{\text{pre}}(\mathbf{x}_{\text{suf}})^{\omega}$. The most important factors are whether a property can be verified over finite- or infinite-horizons. All infinite-horizon (liveness) properties must be satisfied on the suffix $\mathbf{x}_{\text{suf}}$.

We begin with the fundamental temporal operators Φ_{core}. Safety and persistence require an additional mixed-integer linear constraint for each time step, while guarantee and liveness only require a single additional mixed-integer linear constraint.

Safety, $\varphi_{\text{safe}} = \square\psi$, is satisfied by the constraints

$$P_t^{\psi} = 1 \quad \forall t \in \mathcal{T}_{\text{pre}},$$
$$P_t^{\psi} = 1 \quad \forall t \in \mathcal{T}_{\text{suf}},$$

which ensure that the system is always in a $[[\psi]]$ region. Similarly, persistence, $\varphi_{\text{per}} = \diamond\square\psi$, is enforced by

$$P_t^{\psi} = 1 \quad \forall t \in \mathcal{T}_{\text{suf}},$$

which ensures the system eventually remains in a $[[\psi]]$ region.

Guarantee, $\varphi_{\text{goal}} = \diamond\psi$, is satisfied by the constraints

$$\sum_{t\in\mathcal{T}_{\text{pre}}} P_t^{\psi} + \sum_{t\in\mathcal{T}_{\text{suf}}} P_t^{\psi} \geq 1,$$

which ensures the system eventually visits a $[[\psi]]$ region. Similarly, liveness $\varphi_{\text{live}} = \square\diamond\psi$ is enforced by

$$\sum_{t\in\mathcal{T}_{\text{suf}}} P_t^{\psi} \geq 1,$$

which ensures the system repeatedly visits a $[[\psi]]$ region.

Now consider the response temporal operators Φ_{resp}. For these formulas, the definition of implication is used to convert each inner formula into a disjunction between a property that holds at a state and a property that holds at some point in the future. Loosely speaking, the response formulas require an additional mixed-integer linear constraint for each time step.

For next-step response, $\varphi^1_{\text{resp}} = \square(\psi \implies \bigcirc\phi) = \square(\neg\psi \vee \bigcirc\phi)$, the additional constraints are

$$P_t^{\neg\psi} + P_{t+1}^{\phi} = 1, \quad t = 0,1,2,\dots,T-1,$$
$$P_T^{\neg\psi} + P_{T_s}^{\phi} = 1,$$

Similarly, steady-state next-step response, $\varphi^3_{\text{resp}} = \diamond\square(\psi \implies \bigcirc\phi) = \diamond\square(\neg\psi \vee \bigcirc\phi)$, is satisfied by

$$P_t^{\neg\psi} + P_{t+1}^{\phi} = 1, \quad t = T_s,\dots,T-1,$$
$$P_T^{\neg\psi} + P_{T_s}^{\phi} = 1,$$

Eventual response, $\varphi^2_{\text{resp}} = \Box(\psi \implies \Diamond\phi) = \Box(\neg\psi \vee \Diamond\phi)$, requires the following constraints

$$P_t^{\neg\psi} + \sum_{\tau=t}^{T} P_\tau^{\phi} \geq 1, \quad \forall t \in \mathcal{T}_{\text{pre}},$$
$$P_t^{\neg\psi} + \sum_{t\in\mathcal{T}_{\text{suf}}} P_t^{\phi} \geq 1, \quad \forall t \in \mathcal{T}_{\text{suf}}.$$

Similarly, for steady-state eventual response, $\varphi^4_{\text{resp}} = \Diamond\,\Box\,(\psi \implies \Diamond\phi) = \Diamond\,\Box\,(\neg\psi \vee \Diamond\phi)$, the additional constraints are

$$P_t^{\neg\psi} + \sum_{t\in\mathcal{T}_{\text{suf}}} P_t^{\phi} \geq 1, \quad \forall t \in \mathcal{T}_{\text{suf}}.$$

Now consider the fairness temporal operators Φ_{fair}. In the following, the definition of implication is used to rewrite the inner formula as disjunction between a single safety (persistence) property and a conjunction of guarantee (liveness) properties. These formulas loosely require an additional mixed-integer linear constraint for each conjunction in the response and each time step.

Conditional goal visitation, $\varphi^1_{\text{fair}} = \Diamond\psi \implies \bigwedge_{j=1}^m \Diamond\phi_j = \Box\neg\psi \vee \bigwedge_{j=1}^m \Diamond\phi_j$, is specified by

$$P_t^{\neg\psi} + \sum_{t\in\mathcal{T}} P_t^{\phi_j} \geq 1, \quad \forall j = 1,\ldots,m, \forall t \in \mathcal{T}.$$

Conditional repeated goal visitation, $\varphi^2_{\text{fair}} = \Diamond\psi \implies \bigwedge_{j=1}^m \Box\Diamond\,\phi_j = \Box\neg\psi \vee \bigwedge_{j=1}^m \Box\Diamond\,\phi_j$, is enforced as

$$P_t^{\neg\psi} + \sum_{t\in\mathcal{T}_{\text{suf}}} P_t^{\phi_j} \geq 1, \quad \forall j = 1,\ldots,m, \forall t \in \mathcal{T}.$$

Similarly, $\varphi^3_{\text{fair}} = \Box\Diamond\psi \implies \bigwedge_{j=1}^m \Box\Diamond\phi_j = \Diamond\Box\neg\psi \vee \bigwedge_{j=1}^m \Box\Diamond\,\phi_j$, is represented by

$$P_t^{\neg\psi} + \sum_{t\in\mathcal{T}_{\text{suf}}} P_t^{\phi_j} \geq 1, \quad \forall j = 1,\ldots,m,\ \forall t \in \mathcal{T}_{\text{suf}}.$$

We now gather all of the relevant constraints for Problems 1 and 2. These include the dynamical constraints $C_{\mathcal{D}}$ from the MLD system and the logical constraints C_{LTL} just defined in this section. The cost function (if applicable) is expressed as a function of $\mathbf{x}_{\text{suf}}$ and the corresponding control input sequence. The above constraints can be used to represent any formula of the form (13). Given a set of dynamical and logical constraints, Problems 1 and 2 can be solved using a mixed-integer linear program (MILP) or mixed-integer quadratic program (MIQP) solver. While even a MILP feasibility check is NP-hard, modern solvers using branch and bound methods can routinely solve large problems. In Section VI, we show encouraging results on high-dimensional continuous systems.

C. Simultaneous solution with automatic splitting

In Section V-B, the trajectory was *a priori* split into a prefix and a suffix. In this section, we introduce a binary variable at each time instance to determine when to start the suffix. This requires enforcing the cycle constraint at the appropriate point, evaluating the cost function (if applicable) only over the suffix, and only enforcing certain temporal operators on the suffix.

Let $s_t \in \{0,1\}$ for all $t \in \mathcal{T}$ and $s_{t+1} \geq s_t$ for $t = 0,1,\ldots,T-1$. The binary variable $s_t = 1$ if and only if time instance t is part of the suffix, and $s_t = 0$ if and only if time instance t is part of the prefix. Additional constraints on the switching time or length of the suffix (e.g., to satisfy Assumption 1) can be enforced by $s_{t+1} = s_t$ for desired values of $t \in \mathcal{T}$, which prevents time $t+1$ from being the start of the suffix. Note that there are only T possibilities for the valuations of all s_t, so the increase in complexity is moderate.

The cycle closure constraint is now, $x(t) = x(T)$ if and only if $s_t - s_{t-1} = 1$. This constraint is modeled as

$$x(t) \leq x(T) + M(1 + s_{t-1} - s_t)\mathbf{1},$$
$$x(t) \geq x(T) - M(1 + s_{t-1} - s_t)\mathbf{1}, \forall t \in \mathcal{T},$$

where M is a sufficiently large constant and $\mathbf{1} := (1,1,\ldots,1)^T$ be a vector of ones of appropriate dimension.

We model the fact that a variable is true in the suffix by using a lifting method from [5]. Let $\theta_t^\psi := s_t P_t^\psi$. Note that when time t is on the prefix, $\theta_t^\psi = 0$. When time t is on the suffix, θ_t^ψ is unconstrained. To enforce the relationship $\theta_t^\psi = s_t P_t^\psi$, introduce the mixed-integer linear constraints

$$\theta_t \leq M s_t,$$
$$\theta_t \geq -M s_t,$$
$$\theta_t \leq P_t^\psi + M(1 - s_t)\mathbf{1}, \text{ and}$$
$$\theta_t \geq P_t^\psi - M(1 - s_t)\mathbf{1},$$

where M is a sufficiently large constant and $\mathbf{1} := (1,1,\ldots,1)^T$ be a vector of ones of appropriate dimension. This lifting approach also should be done for the variables in the cost function, so that the cost is only incurred over the suffix.

The constraints in Section V-B that were previously only enforced over the suffix $\mathcal{T}_{\text{suf}}$ must now be modified. This modification occurs in two general ways. For constraints where terms are summed over $t \in \mathcal{T}_{\text{suf}}$, the P_t^ψ variables are replaced with θ_t^ψ for all $t \in \mathcal{T}$. For example, consider liveness, $\varphi_{\text{live}} = \Box\Diamond\,\psi$. The constraint is now $\sum_{t\in\mathcal{T}} \theta_t^\psi \geq 1$, which enforces that $[[\psi]]$ is visited on the suffix, i.e., $s_t = 1$. For constraints where terms must satisfy a constraint for all $t \in \mathcal{T}_{\text{suf}}$, the constraint is relaxed by multiplying it by s_t. For example, consider persistence, $\varphi_{\text{per}} = \Diamond\,\Box\,\psi$. The constraint is now $P_t^\psi \geq s_t$ for all $t \in \mathcal{T}$, which removes the constraint on the prefix, where $s_t = 0$. The reformulation of the constraints in Section V-B is similar and is left to the reader.

D. Sequential solution approach

This section trades completeness for computational efficiency by first computing $\mathbf{x}_{\text{suf}}$ and then computing $\mathbf{x}_{\text{pre}}$. This approach is computationally appealing because it divides the problem into two smaller problems. However, the cycle computed may not be reachable from the initial condition. The computation of $\mathbf{x}_{\text{suf}}$ can be done for a given length as in Section V-B or the length of $\mathbf{x}_{\text{suf}}$ can be selected during the optimization as in Section V-C. The mixed-integer linear constraints that were previously developed for the temporal operators can largely be applied here, although one must relax properties on the prefix that have already been satisfied on the suffix. Finally, these techniques can be used to optimize the prefix after initially using the simultaneous approach to find an optimal reachable suffix.

First, select a suffix parameterization $\mathbf{x}_{\text{suf}}$. Apply only the temporal operator constraints previously introduced that must hold over the suffix. Solve the resulting MILP (or MIQP) for the optimal trajectory. Then, select a prefix parameterization $\mathbf{x}_{\text{pre}}$ that satisfies the initial condition. Apply boundary conditions to $\mathbf{x}_{\text{pre}}$ so that it connects with one of the points on $\mathbf{x}_{\text{suf}}$, i.e., $\mathbf{x}_{\text{pre}}(T_{\text{pre}}+1) = \mathbf{x}_{\text{suf}}(k)$ for some point k along $\mathbf{x}_{\text{suf}}$. This can be formed by using binary variables to model the disjunction. Finally, note that the suffix $\mathbf{x}_{\text{suf}}$ computed in the previous stage satisfies the infinite-horizon properties and might also satisfy some of the finite-time properties. Thus, one only needs to apply constraints to the prefix for the temporal operators that have not already been satisfied along the suffix. This can be automatically checked from the equations defining the temporal constraints in Section V-B.

VI. Examples

We demonstrate our techniques on a variety of motion planning problems. The first example is a hybrid double integrator that was previously considered in [17]. The second example is a chain of integrators parameterized by dimension. Our last example is a quadrotor model that was previously considered in [26]. All computations were done on a laptop with a 2.4 GHz dual-core processor and 4 GB of memory using CPLEX [1] through Yalmip [21].

A. Hybrid integrator

We consider the discrete-time linear hybrid system

$$x_1(t+1) = \begin{cases} x_1(t) + x_3(t) + 0.5u_1(t) & \text{if } x_1(t) \geq 1 \\ x_1(t) + 0.5x_3(t) + 0.5u_1(t) & \text{if } x_1(t) < 1, \end{cases}$$
$$x_2(t+1) = x_2(t) + x_4(t) + 0.5u_2(t),$$
$$x_3(t+1) = x_3(t) + u_1(t),$$
$$x_4(t+1) = x_4(t) + u_2(t),$$

with $|u_1| \leq 1$, $|u_2| \leq 1$, $|x_3| \leq 1$, and $|x_4| \leq 1$. This planar system is a discrete-time double integrator in orthogonal directions and can be rewritten as a MLD system [17]. The variables x_1, x_2 are position and x_3, x_4 are velocity.

We first consider an example motivated by a pickup and delivery task. All properties should be understood to be with respect to polyhedral regions in the $x_1 - x_2$ plane (see Figure 1). Let P be a region where supplies can be picked up and $D1$, $D2$, and $D3$ be regions where supplies must be delivered. The robot must remain safe, i.e., in the white region. Additionally, the robot must alert its boss that it has started the task by visiting region I (the blue region). Formally, the task specification is $\varphi = \square S \wedge \Diamond I \wedge \square\Diamond P \wedge \square\Diamond D1 \wedge \square\Diamond D2 \wedge \square\Diamond D3$. Additionally, we minimize the average cost-per-task-cycle (i.e., visiting P,$D1$,$D3$, and $D4$) of the from

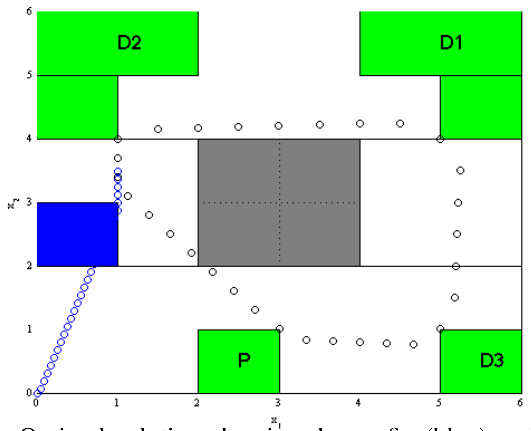

Fig. 1: Optimal solution showing the prefix (blue) and suffix (black).

(14), where $c(x_t, u_t) = |u_1(t)| + |u_2(t)|$ penalizes the control input.

Using the sequential solution approach, we are able to solve for a feasible solution in under 1 second. This solution had a cost of 11.45, which was reduced to 5.7 after 10 seconds. The simultaneous approach with both a user-specified prefix-suffix split and an automatic split found feasible solutions with costs of 6.2 and 6.5 in 8 and 22 seconds, respectively.

In the remainder of this section, we consider the motion planning problem illustrated in Figure 2 for different system models. The problem specification requires the system to avoid all obstacles while repeatedly visiting two regions. Results for the hybrid integrator are given in Table I under "hybrid".

B. Chain of integrators

We now consider a chain of orthogonal integrators in the x and y directions. The k-th derivative of the x and y positions are controlled as $x^{(k)} = u_x$ and $y^{(k)} = u_y$, respectively. The control inputs satisfy the constraints $|u_x| \leq 1$ and $|u_y| \leq 1$. Results are given in Table I under "Chain4", "Chain6", and "Chain8", where "ChainN" indicates that the N-th derivative in both the x and y positions is controlled. The sequential approach performs well even on 16 dimensional continuous systems, exploiting the natural decomposition of the trajectory parameterization.

C. Quadrotor

We model now consider the quadrotor mode used in [26] for point-to-point motion planning, to which we refer the reader for a complete description of the model. The state $x = (p, v, r, w)$ is 10-dimensional, consisting of position $p \in \mathbb{R}^3$, velocity $v \in \mathbb{R}^3$, orientation $r \in \mathbb{R}^2$, and angular velocity $w \in \mathbb{R}^2$. This model is the linearization of a nonlinear model about hover with the yaw constrained to be zero. The control input $u \in \mathbb{R}^3$ is the total, roll, and pitch thrust. Results are given in Table I under "quadrotor". Feasible solutions are returned in a matter of seconds using the sequential approach.

Model	Dim.	Feasible Soln. (sec) Sim.	Feasible Soln. (sec) Seq.	Num. Solved Sim.	Num. Solved Seq.
Hybrid	4	2.72 ±.39	1.07 ±.10	30	30
Chain4	8	14.7 ±2.2	3.42 ±.56	25	30
Chain6	12	32.0 ±2.3	5.81 ±1.2	12	30
Chain8	16	36.3 ±1.8	7.03 ±1.7	5	28
Quadrotor	10	10.9 ±2.4	2.28 ±.48	26	30

TABLE I: Time until a feasible solution was found (mean ± standard error) and number of problems solved in 40 seconds (out of 30). Each trial was a randomly generated 5x5 grid with two regions that must be visited repeatedly. A trajectory of length 80 was used in all cases, and all results were averaged over 30 randomly generated environments. The simultaneous approach used between 342 and 685 binary variables with a mean of 557 ± 14.

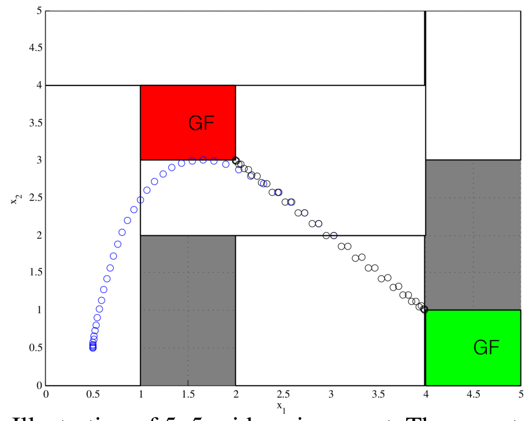

Fig. 2: Illustration of 5x5 grid environment. The repeated goals are the red and green regions. Dark regions are obstacles. A representative system trajectory is shown with the prefix (blue) and suffix (black).

D. *Comparison to Discrete Abstractions*

We compared our approach to a reachability-based algorithm that computes a finite abstraction [28]. The performance of our approach on a series of randomly generated grid environments is shown in Figure 3. These results use the sequential approach, given its superior performance in our preliminary examples. Our results appear promising, especially for situations where the environment is dynamically changing and a finite abstraction must be repeatedly computed.

VII. Conclusion

We presented a novel mixed-integer programming-based method for control of MLD systems with a useful fragment of LTL that allows both finite- and infinite-horizon properties to be specified. Our method is efficient in the number of binary variables used to model the an LTL formula. Additionally, we showed the computational effectiveness of our approach on motion planning examples.

Future work will consider reactive environments by including both continuous and discrete disturbances and potentially using a receding horizon control approach. Additionally, we will expand the space of tasks that can be specified by including additional temporal operators and timing constraints.

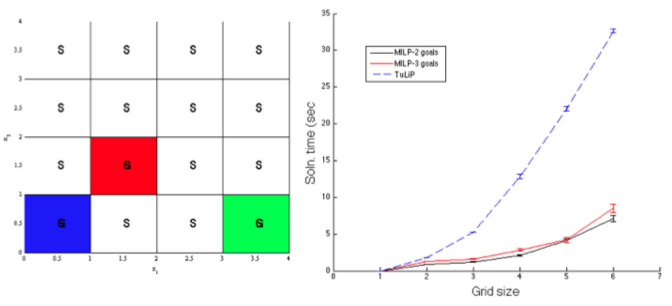

Fig. 3: Left: Typical gridworld with grid size of 4 and 3 goals. Right: Comparison of our approach vs. computing a finite abstraction with TuLiP.

Acknowledgments

This work was supported by a NDSEG Fellowship and the Boeing Corporation.

References

[1] *User's Manual for CPLEX V12.1. IBM, 2009.*

[2] Rajeev Alur, Thomas A. Henzinger, Gerardo Lafferriere, and George J. Pappas. Discrete abstractions of hybrid systems. *Proc. IEEE*, 88(7):971–984, 2000.

[3] C. Baier and J-P. Katoen. *Principles of Model Checking*. MIT Press, 2008.

[4] C. Belta and L. C. G. J. M. Habets. Controlling of a class of nonlinear systems on rectangles. *IEEE Trans. on Automatic Control*, 51:1749–1759, 2006.

[5] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35:407–427, 1999.

[6] A. Bemporad, G. Ferrari-Trecate, and M. Morari. Observability and controllabililty of piecewise affine and hybrid systems. *IEEE Transaction on Automatic Control*, 45:1864–1876, 2000.

[7] A. Bhatia, M. R. Maly, L. E. Kavraki, and M. Y. Vardi. Motion planning with complex goals. *IEEE Robotics and Automation Magazine*, 18:55–64, 2011.

[8] C. E. Blair, R. G. Jeroslow, and J. K. Lowe. Some results and experiments in programming techniques for propositional logic. *Computers and operations research*, 13:633–645, 1986.

[9] V. D. Blondel and J. N. Tsitsiklis. A survey of computational complexity results in systems and control. *Automatica*, 36(9):1249–1274, 2000.

[10] Y. Chen, J. Tumova, and C. Belta. LTL robot motion control based on automata learning of environmental dynamics. In *IEEE Int. Conf. on Robotics and Automation*, 2012.

[11] M. G. Earl and R. D'Andrea. Iterative MILP methods for vehicle-control problems. *IEEE Transactions on Robotics*, 21:1158–1167, 2005.

[12] G. E. Fainekos, A. Girard, H. Kress-Gazit, and G. J. Pappas. Temporal logic motion planning for dynamic robots. *Automatica*, 45:343–352, 2009.

[13] L. Habets, P. J. Collins, and J. H. van Schuppen. Reachability and control synthesis for piecewise-affine hybrid systems on simplices. *IEEE Trans. on Automatic Control*, 51:938–948, 2006.

[14] J. N. Hooker and C. Fedjki. Branch-and-cut solution of inference problems in propositional logic. *Annals of Mathematics and Artificial Intelligence*, 1:123–139, 1990.

[15] S. Karaman and E. Frazzoli. Sampling-based motion planning with deterministic μ-calculus specifications. In *Proc. of IEEE Conf. on Decision and Control*, 2009.

[16] S. Karaman and E. Frazzoli. Linear temporal logic vehicle routing with applications to multi-UAV mission planning. *International Journal of Robust and Nonlinear Control*, 21:1372–1395, 2011.

[17] S. Karaman, R. G. Sanfelice, and E. Frazzoli. Optimal control of mixed logical dynamical systems with linear temporal logic specifications. In *Proc. of IEEE Conf. on Decision and Control*, pages 2117–2122, 2008.

[18] M. Kloetzer and C. Belta. A fully automated framework for control of linear systems from temporal logic specifications. *IEEE Trans. on Automatic Control*, 53(1): 287–297, 2008.

[19] Y. Kwon and G. Agha. LTLC: Linear temporal logic for control. In *HSCC '08: Proc. of the International Workshop on Hybrid Systems*, pages 316–329, 2008.

[20] S. M. LaValle. *Planning algorithms*. Cambridge Univ. Press, 2006.

[21] J. Löfberg. YALMIP : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. Software available at http://control.ee.ethz.ch/~joloef/yalmip.php.

[22] A. Richards and J. P. How. Aircraft trajectory planning with collision avoidance using mixed integer linear programming. In *American Control Conference*, 2002.

[23] S. L. Smith, J. Tumova, C. Belta, and D. Rus. Optimal path planning for surveillance with temporal-logic constraints. *The International Journal of Robotics Research*, 30:1695–1708, 2011.

[24] P. Toth and D. Vigo, editors. *The Vehicle Routing Problem*. Philadelphia, PA: SIAM, 2001.

[25] M. P. Vitus, V. Pradeep, J. Hoffmann, S. L. Waslander, and C. J. Tomlin. Tunnel-MILP: path planning with sequential convex polytopes. In *AIAA Guidance, Navigation, and Control Conference*, 2008.

[26] Dustin J. Webb and Jur van den Berg. Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics. In *IEEE Int. Conf. on Robotics and Automation*, 2013.

[27] E. M. Wolff, U. Topcu, and R. M. Murray. Optimal control with weighted average costs and temporal logic specifications. In *Proc. of Robotics: Science and Systems*, 2012.

[28] T. Wongpiromsarn, U. Topcu, N. Ozay, H. Xu, and R. M. Murray. TuLiP: A software toolbox for receding horizon temporal logic planning. In *Proc. of Int. Conf. on Hybrid Systems: Computation and Control*, 2011. http://tulip-control.sf.net.

[29] T. Wongpiromsarn, U. Topcu, and R. M. Murray. Receding horizon temporal logic planning. *IEEE Trans. on Automatic Control*, 2012.