Model Predictive Control with Signal Temporal Logic Specifications

Vasumathi Raman, Alexandre Donzé, Mehdi Maasoumy, Richard M. Murray, Alberto Sangiovanni-Vincentelli and Sanjit A. Seshia

Abstract—We present a mathematical programming-based method for model predictive control of discrete-time cyber-physical systems subject to signal temporal logic (STL) specifications. We describe the use of STL to specify a wide range of properties of these systems, including safety, response and bounded liveness. For synthesis, we encode STL specifications as mixed integer-linear constraints on the system variables in the optimization problem at each step of model predictive control. We present experimental results for controller synthesis on simplified models of a smart building-level micro-grid and HVAC system.

I. INTRODUCTION

Temporal logics provide a rigorous, precise and unambiguous formalism for specifying and verifying desired behaviors of systems. In particular, algorithms for verification and synthesis of discrete supervisory controllers exist that satisfy the specified properties. These discrete controllers have successfully been employed in the construction of hybrid controllers for cyber-physical systems in various domains, including robotics [6] and aircraft power system design [17]. However, for physical systems that require constraints not just on the order of events, but also on the temporal distance between them, simulation and testing is still the method of choice for validating properties and establishing guarantees; the exact exhaustive verification of such systems is in general undecidable [1].

Signal Temporal Logic (STL) [14] was originally developed in order to specify and monitor the expected behavior of physical systems, including temporal constraints between events. STL allows the specification of properties of dense-time, real-valued signals, and the automatic generation of monitors for testing these properties of individual simulation traces. It has since been applied to the analysis of several types of continuous and hybrid systems, including dynamical systems and analog circuits, where the continuous variables represent quantities like currents and voltages in the circuit. An important advantage of STL is that, in contrast to standard semantics for other temporal logics, the satisfaction of an STL formula with bounded modalities has unambiguous interpretation with respect to a finite signal or sequence of states. Another advantage is that it naturally admits a quantitative semantics, which in addition to the yes/no answer to the satisfaction question provides a real number grading the quality of the satisfaction or violation. Different semantics have been defined for STL [5] and can assess the robustness of the systems to parameter or timing variations.

Model Predictive Control (MPC) is based on iterative, finite horizon, discrete time optimization of a model of the plant. At any given time $t$ the current plant state is observed, and an optimal control strategy computed for some finite time horizon in the future, $[t, t+N]$. An online calculation is used to explore possible future state trajectories originating from the current state, finding an optimal control strategy until time $t+N$. Only the first step of the computed optimal control strategy is implemented; the plant state is then sampled again, and new calculations are performed on a horizon of $N$ starting from the new current state. While the global optimality of this sort of "receding horizon" approach is not ensured, it tends to do well in practice. In addition to reducing computational complexity, it improves the system robustness with respect to exogenous disturbances and modeling uncertainties [15].

In a recent work on optimal control synthesis of aircraft load management systems [10], it was shown in practice how to convert STL-like specifications into time dependent equality and inequality constraints, yielding a MILP. The MILP was then solved in an MPC fashion leading to optimal control policy for the system at hand. However, the transformation of STL-like constraints into equality and inequality constraints was done manually. This process can be very time consuming, cumbersome and is problem-specific. One of our contributions in this paper, is to make the process of converting STL specification to MILP constraints fully automatic.

In this work, we frame the MPC problem as an instance of synthesis from STL specifications. The objective is to specify desired properties of the system using a STL formula, and synthesize control such that the system satisfies that specification, while using a receding horizon approach. Our work extends the standard bounded model checking (BMC) paradigm for finite discrete systems [3] to STL, which accommodates continuous systems. In traditional BMC, discrete state sequences of a fixed length are obtained as counterexamples to a Boolean satisfiability (SAT) problem. The approach has been extended to hybrid systems, either by computing a discrete abstraction of the system as in
or by extending SAT solvers to reason about linear inequalities as in [2], [7]. Similarly, mixed integer linear programming (MILP) encodings inspired by BMC have been used to generate trajectories for continuous systems with linear temporal logic specifications [8], [9], [18]. However, this is the first work to consider a BMC approach for synthesis from signal temporal logic specifications.

Our main contributions are a BMC-inspired encoding for STL specifications as mixed integer linear constraints on a cyber-physical system. We show how this encoding can be used to generate signals that satisfy finite and infinite horizon STL properties in an open-loop fashion, and moreover to generate signals that are maximally robust. We also demonstrate how our MILP formulation of the STL synthesis problem can be used as part of existing controller synthesis frameworks like model predictive control, to compute feasible and optimal controllers for cyber-physical systems that must satisfy timed and temporal properties. We present experimental results for controller synthesis on simplified models of a smart building-level micro-grid and HVAC system. We show how the MPC schemes for controlling the ancillary service power flow and demand in these examples can be framed in terms of synthesis from an STL specification, and present simulation results to illustrate the effectiveness of our proposed synthesis methodology.

II. Preliminaries

A. Discrete Time Continuous Systems

We consider discrete-time continuous systems of the form

$$ x_{t+1} = f(x_t, u_t) \quad (1) $$

where \( t = 0, 1, \ldots, \) are the time indices, \( x \in \mathcal{X} \subseteq \mathbb{R}^{n_c} \times \{0, 1\}^{n_l} \) are the continuous and binary/logical states, \( u \in \mathcal{U} \subseteq \mathbb{R}^{m_c} \times \{0, 1\}^{m_l} \) are the (continuous and logical) control inputs, and \( x_0 \in \mathcal{X} \) is the initial state. A run \( x = x_0 x_1 x_2 \ldots \) is an infinite sequence of its states, where \( x_t \in \mathcal{X} \) is the state of the system at index \( t \), and for each \( t = 0, 1, \ldots \), there exists a control input \( u_t \in \mathcal{U} \) such that \( x_{t+1} = f(x_t, u_t) \). Given an initial state \( x_0 \) and a control input sequence \( u = u_0 u_1 u_2 \ldots \), the resulting run \( x = x(x_0, u) \) of a system modeled by (1) is unique.

Restricting to finite sequences, given a control input sequence \( u^N = u_0 u_1 u_2 \ldots u_N \), we let the resulting horizon-\( N \) run be \( x^N = x(x_0, u^N) = x_0 x_1 x_2 \ldots x_N \). We also introduce the notion of a generic cost function \( J(x, u) \) that maps finite and infinite runs to \( \mathbb{R} \cup \infty \).

B. Model Predictive Control (MPC)

For systems modeled by \ref{eq:system}, we present the optimal control problem, and then the MPC formulation here.

**Problem 1 (Optimal Control):** Given a system of the form (1) and an initial state \( x_0 \), compute

$$ \arg\min_{u} J(x(x_0, u)) $$

**Problem 2 (Model Predictive Control):** Given a system of the form (1) and a horizon \( H \), compute at each time step \( t \)

$$ \arg\min_{u^H} J(x^H(x_t, u^H)) $$

where \( u^H = x_t x_{t+1} x_{t+2} \ldots x_{t+H} \).

Note that solving Problem 2 need not solve Problem 1.

C. Signal Temporal Logic

For this work, we assume that STL formulas are provided in negation normal form, so all negations appear in front of predicates. An STL formula can always be rewritten as a negation normal form formula of size linear in the original size. STL formulas are thus defined recursively as:

$$ \varphi ::= \mu \mid \neg \varphi \mid \varphi \land \psi \mid \varphi \lor \psi \mid \Box_{[a,b]} \varphi \mid \varphi \U_{[a,b]} \psi $$

where \( \mu \) is a predicate which value is determined by the sign of some function of an underlying signal \( x \), i.e., \( \mu \equiv \mu(x) > 0 \). Additionally, we define \( \Box_{[a,b]} \varphi = \top \U_{[a,b]} \varphi \).

The validity of a formula \( \varphi \) with respect to signal \( x \) at time \( t \) is defined inductively as follows:

$$ (x, t) \models \mu \iff \mu(x(t)) = 0 $$

$$ (x, t) \models \neg \mu \iff \neg((x, t) \models \mu) $$

$$ (x, t) \models \varphi \land \psi \iff (x, t) \models \varphi \land (x, t) \models \psi $$

$$ (x, t) \models \varphi \lor \psi \iff (x, t) \models \varphi \lor (x, t) \models \psi $$

$$ (x, t) \models \Box_{[a,b]} \psi \iff \forall t' \in [t + a, t + b], (x, t') \models \psi $$

$$ (x, t) \models \varphi \U_{[a,b]} \psi \iff \exists t' \in [t + a, t + b] \text{ s.t. } (x, t') \models \psi \land \forall t'' \in [t', t''], (x, t'') \models \varphi. $$

A run \( x = x_0 x_1 x_2 \ldots \) satisfies \( \varphi \), denoted by \( x \models \varphi \), if \( (x, 0) \models \varphi \).

Note that since we deal only with discrete-time systems in this work, the STL formulas we consider refer only to intervals over discrete time values. In fact, considering continuous time formulas renders the satisfiability of STL undecidable, so the discrete time restriction is necessary for our approach.

D. Robust Satisfaction of STL formulas

Quantitative or robust semantics define a real-valued function \( \rho^\varphi \) of \( x \) and \( t \) such that \( (x, t) \models \varphi \equiv \rho^\varphi(x, t) > 0 \). This can be done from the above semantics in a straightforward manner, by propagating values of the functions associated with each predicates using min and max operators corresponding to the different operators of STL. For example, the robust satisfaction of \( \mu_1 \equiv x - 3 > 0 \) at time 0 is simply \( \rho^\mu_1(x, 0) = x(0) - 3 \). The robust satisfaction of \( \mu_1 \land \mu_2 \) is the minimum \( \rho^\mu_1 \) and \( \rho^\mu_2 \). Temporal operators can be treated as conjunction and disjunctions along the time axis, e.g., the robust satisfaction of \( \varphi = \Box_{[0,2]} \mu_1 \) is

$$ \rho^\varphi(x, t) = \min_{t \in [0,2]} \rho^{\mu_1}(x, t) + \min_{t \in [0,2]} x(t) - 3. $$

The
complete robust semantics is defined as follows:

\[
\begin{align*}
\rho^\mu(x,t) &= \mu(x(t)), \\
\rho^{-\mu}(x,t) &= -\mu(x(t)), \\
\rho^{\varphi\land\psi}(x,t) &= \min(\rho^\varphi(x,t), \rho^\psi(x,t)), \\
\rho^{\varphi\lor\psi}(x,t) &= \max(\rho^\varphi(x,t), \rho^\psi(x,t)), \\
\rho^{-[a,b]}(x,t) &= \min_{t+a,t+b}(\rho^\psi(x,t)), \\
\rho^\varphi U_{[a,b]}(x,t) &= \max_{t\in[t+a,t+b]}(\min(\rho^\psi(x,t)), \min_{t\in[t',t'']} \rho^\varphi(x,t''))
\end{align*}
\]

III. PROBLEM STATEMENT

We now formally state the STL controller synthesis problem.

**Problem 3 (STL controller synthesis):** Given a system of the form (1), an initial state \(x_0\) and an STL formula \(\varphi\), compute a control input sequence \(u\) such that \(x(x_0,u) \models \varphi\).

In an MPC framework, we can re-cast this as follows:

**Problem 4 (STL MPC):** Given a system of the form (1), an initial state \(x_0\) and an STL formula \(\varphi\), compute at each time step \(t\) a control input sequence \(u^H_t\) such that \(x(x_0,u) \models \varphi\), where \(u\) is constructed such that at each step \(t\), \(u_t = u^H_t(1)\).

Given an objective function \(J\) on the system, we can also formulate a version with optimality:

**Problem 5 (STL MPC OPT):** Given a system of the form (1), an initial state \(x_0\) and an STL formula \(\varphi\), compute at each time step \(t\)

\[
\argmin_{u^H_t} J(x^H_t(x_t,u^H_t),u^H_t) \text{ s.t. } x(x_0,u) \models \varphi.
\]

In Sections IV and V, we present both an open-loop solution to Problem 3, and a solution to Problems 4 and 5 using an MPC formulation. In the absence of an objective function \(J\) on the system, we maximize the robustness of the generated runs with respect to \(\varphi\).

IV. OPEN-LOOP CONTROLLER SYNTHESIS

In order to solve Problem 3, we add STL constraints to an MILP formulation of the open-loop control problem. To do so, first, we represent the system trajectory over the MPC prediction horizon as a finite sequence of states satisfying the model dynamics (1). Then we encode the formula \(\varphi_f\) with a set of Mixed Integer Linear Program (MILP) constraints.

A. Finite Trajectory Parametrization

The notation used in this paper follows that of [4] and [18], but is the first to extend it to support STL specifications.

To allow interpretation of the STL specification over infinite sequences of states, an infinite sequence is represented over this finite horizon by a finite trajectory with a loop. Note that this assumption that the trajectory eventually be periodic renders our approach conservative for general systems of the form (1) with an infinite state space.

In order to specify properties of infinite executions of this finite parametrized trajectory, we will enforce a lasso shape on this finite sequence, requiring a loop that makes some eventual portion of the trajectory periodic. We will then encode the STL formula as mixed integer-linear constraints on this finite trajectory. The constraints from the system model are also included as part of the optimization problem, as with standard MPC. This will enable Problem 3 to be solved using a mixed-integer linear program (MILP) solver, which, although NP-hard, can be solved in practice for large problems using modern solvers with sophisticated search heuristics.

We now present some definitions that are common to most approaches based on bounded model checking.

**Definition 1:** A run \(x\) is a \((N,1)\)-loop if

\[
x = (x_0x_1...x_{l-1})(x_1...x_N)^\omega,
\]

such that \(0 < l \leq N\) and \(x_{l-1} = x_N\), where \((\sigma)^\omega\) denotes infinite repetition of sequence \(\sigma\).

**Definition 2:** Given a run \(x\) and a bound \(N \in \mathbb{N}\), \(x \models_N \varphi\) off \(x\) is a \((N-l)\)-loop for some \(0 < l \leq N\) and \((x,0) \models \varphi\).

We now propose to solve Problem 3 by replacing \(x(x_0,u) \models \varphi\) with \(x(x_0,u) \models_N \varphi\), where \(N\) is the length of the parametrized trajectory. To do so, we all build a set of MILP constraints that is satisifiable if and only if there exists a trajectory of length \(N\) that satisfies \(\varphi\). The satisifiability of these constraints can be checked using a MILP solver, which also yields a solution to the constraints when feasible. This solution gives us the control input \(u\) desired.

There are several components to the encoding of Problem 3 as a set of MILP constraints; these include system constraints, loop constraints and STL constraints.

B. Constraints on system evolution

The first component of the set of constraints is provided by the system model. The system constraints encode valid trajectories of length \(N\) for a system of the form (1) – these constraints hold if and only if the trajectory \(x(x_0,u)\) satisfies (1) for \(t = 0, 1, ..., N\). Note that this is quite general, and accommodates any system for which the resulting constraints and objectives form a mixed integer-linear program. An example is the building-level smart-grid control system model presented in [12]. Other useful examples include mixed-logical dynamical systems such as those presented in [18]. Other cost functions and system dynamics can also be included by using appropriate solvers.

C. Loop constraints for trajectory parametrization

The loop constraints enforce the existence of a loop in the finite system trajectory. We introduce \(N\) binary variables \(l_1, ..., l_N\), which determine where the loop forms. These are constraints such that only one is enabled (set to True) at a time, and if \(l_j = 1\), then \(x_{j-1} = x_N\). The following constraints (which employ the big M method from operations research) enforce these requirements:

- \(\sum_{i=1}^{N} l_i = 1\)
- \(x_N \leq x_{j-1} + M_j(1 - l_j), j = 1, ..., N\)
- \(x_N \geq x_{j-1} + M_j(1 - l_j), j = 1, ..., N\)

where \(M_j\) are sufficiently large positive numbers.
D. STL constraints

Given a formula \( \varphi \), we introduce a variable \( r_{i}^{\varphi} \), whose value is tied to a set of mixed integer-linear constraints required for the satisfaction of \( \varphi \) at position \( t \) in the state sequence. In other words, \( r_{i}^{\varphi} \) has an associated set of MILP constraints such that \( r_{i}^{\varphi} > 0 \) if and only if \( \varphi \) holds at position \( t \). We recursively generate the MILP constraints corresponding to \( r_{i}^{\varphi} \) – the value of this variable determines whether to not a formula \( \varphi \) holds in the initial state. Additionally, we will ensure that the value of \( r_{i}^{\varphi} = \rho^{\varphi}(x, t) \).

1) Predicates: The predicates are represented by constraints on the system state variables. For each predicate \( \mu \in P \), we introduce variables \( r_{i}^{\mu} \) for time indices \( t = 0, 1, \ldots, N \). We enforce that \( r_{i}^{\mu} > 0 \) if and only if \( \mu(x_{t}) > 0 \). This is achieved by setting \( r_{i}^{\mu} = \mu(x_{i}) \).

2) Boolean operations on MILP variables: As described in Section IV-D.1, each predicate \( \mu \) has an associated continuous variable \( r_{i}^{\mu} \) which is \( > 0 \) if \( \mu \) holds at time \( t \), and \( \leq 0 \) otherwise. In fact, by the recursive definition of our MILP constraints on STL formulas, we can assume that each operand \( \varphi \) in a boolean operation has a corresponding variable \( r_{i}^{\varphi} \) which is \( > 0 \) if \( \varphi \) holds at \( t \) and \( \leq 0 \) if not. Here we define boolean operations on these variables; these are the building blocks of our recursive encoding.

Given a formula \( \psi \) containing a boolean operation, we add new continuous variables \( r_{i}^{\psi} \) to represent its truth value at each time step of the parametrized trajectory. These variables are constrained such that \( r_{i}^{\psi} > 0 \) if \( \psi \) holds at time \( t \) and \( r_{i}^{\psi} \leq 0 \) otherwise.

Negation: \( \psi = \lnot \mu \)
\[
r_{i}^{\psi} = -r_{i}^{\mu}
\]

Conjunction: \( \psi = \bigwedge_{i=1}^{m} \varphi_{i} \)
\[
\sum_{i=1}^{m} b_{i}^{\varphi_{i}} = 1 \quad \text{(2)}
\]
\[
r_{i}^{\psi} \leq r_{i}^{\varphi_{i}}, i = 1, \ldots, m \quad \text{(3)}
\]
\[
r_{i}^{\varphi_{i}} - (1 - b_{i}^{\varphi_{i}}) M \leq r_{i}^{\psi} \leq r_{i}^{\varphi_{i}} + (1 - b_{i}^{\varphi_{i}}) \quad \text{(4)}
\]

where we introduce new binary variables \( b_{i}^{\varphi_{i}} \) for \( i = 1, \ldots, m \), and \( M \) is a sufficiently large positive number. Then (2) enforces that there is one and only one \( j \in \{1m\} \) such that \( b_{i}^{\varphi_{j}} = 1 \), (3) enforces that \( r_{i}^{\psi} \) is smaller than all \( r_{i}^{\varphi_{i}} \), and (4) enforces that \( r_{i}^{\psi} = r_{i}^{\varphi_{j}} \) if and only if \( b_{j} = 1 \). Together, these constraints enforce that \( r_{i}^{\varphi} = \min_{i}(r_{i}^{\varphi_{i}}) \).

Disjunction: \( \psi = \bigvee_{i=1}^{m} \varphi_{i} \)
\[
\sum_{i=1}^{m} b_{i}^{\varphi_{i}} = 1 \\
r_{i}^{\psi} \geq r_{i}^{\varphi_{i}}, i = 1, \ldots, m \\
r_{i}^{\varphi_{i}} - (1 - b_{i}^{\varphi_{i}}) M \leq r_{i}^{\psi} \leq r_{i}^{\varphi_{i}} + (1 - b_{i}^{\varphi_{i}})
\]

Using a similar reasoning to the conjunctive case, these constraints enforce \( r_{i}^{\psi} = \max_{i}(r_{i}^{\varphi_{i}}) \).

3) Temporal constraints: We first present encodings for the \( \square \) and \( \Diamond \) operators. We will use these encodings to define the encoding for the \( U_{[a,b]} \) operator.

Always: \( \psi = \lnot [a,b] \varphi \)
Let \( \bar{a}^{N} = \min(t + a, N) \) and \( \bar{b}^{N} = \min(t + b, N) \).

Define:
\[
r_{i}^{\psi} = \lor_{i=0}^{N} r_{i}^{\varphi_{1}} \land \land_{j=1}^{N} \land_{i=j+\bar{a}} \land_{i=j+\bar{b}} r_{i}^{\varphi_{2}}
\]

The logical operation \( \land \) on the variables \( r_{i}^{\varphi} \) here is as defined in Section IV-D.2.

Eventually: \( \psi = \top [a,b] \varphi \)
Define:
\[
r_{i}^{\psi} = \lor_{i=0}^{N} r_{i}^{\varphi_{1}} \land \land_{j=1}^{N} \land_{i=j+\bar{a}} \land_{i=j+\bar{b}} r_{i}^{\varphi_{2}}
\]

Until: \( \psi = \top a_{[a,b]} \varphi_{2} \)
Given the encoding of \( \lnot [a,b] \varphi \) above, we encode
\[
r_{i}^{\psi} = U_{[a,b]} \varphi_{2} = \bigvee_{t' \in [t+a,t+b]} r_{t'}^{\varphi_{1}} \land r_{t}^{\top [a,b] \varphi_{2}}.
\]

TODO: is this ok?

By induction on the structure of STL formulas \( \varphi \), \( r_{i}^{\varphi} > 0 \) if and only if \( \varphi \) holds on the system at time \( t \), and also that \( r_{i}^{\varphi} = \rho^{\varphi}(x, t) \). With this motivation, we add a final constraint to the MILP: \( r_{0}^{\varphi} > 0 \). The union of the STL constraints, system constraints and loop constraints gives the MILP encoding of Problem 3; this enables checking feasibility of this set and finding a solution using an MILP solver. Given an objective function on runs of the system, this approach also enables finding the optimal open-loop trajectory that satisfies the STL specification. If we in addition maximize the value of \( r_{0}^{\varphi} \), we also get a trajectory that maximizes robustness of satisfaction.

Mixed integer-linear programs are NP-hard, but we can still describe the computational costs of our encoding and approach in terms of the number of variables and constraints in the resulting MILP. If \( P \) is the set of predicates used in the formula, then \( O(N \cdot |P|) \) continuous variables are introduced. In addition, binary and continuous variables are introduced during the MILP encoding of the STL formula. The number of binary and continuous variables introduced is \( O(N \cdot |\varphi|) \), where \( |\varphi| \) is the length (i.e. the number of operators) of the formula. Finally, loop constraints introduce \( N \) additional binary variables.

V. MODEL PREDICTIVE CONTROL SYNTHESIS

We will solve Problem 4 by adding STL constraints to an MPC problem formulation. At each step \( t \) of the MPC computation, we will search for a finite trajectory of decreasing horizon length, such that the accumulated trajectory satisfies \( \varphi \).

We assume that we are generating control for a finite system trajectory of length \( T \) using the MPC paradigm. At
time step 0, we will synthesize control $u^T_0$ using the open-loop formulation in Section IV, including the STL constraints on the entire trajectory. We will then execute only the first time step $u^T_0 (1)$. At the next step of the MPC, we will solve for $u^T_1$, while constraining the previous values of $x_0, u_0$ in the MILP. In this manner, we will keep track of the history of states in order to ensure that the formula is satisfied over the length-$T$ trajectory, while solving for $u^T_{t-1}$ at every time step $t$. In this formulation, while the initial problem solves for the entire length of the trajectory, each subsequent MILP is further constrained based on the past, and so the problem at time $t+1$ is always smaller than that at time $t$.

A. Determining $T$

The length of the trajectory is set based on the formula $\phi$. To pick $T$, we compute the sum for each set of nested upper bounds on the temporal operators, and then compute the maximum over this value. TODO: EXAMPLE. This provides a conservative bound on the length of the trajectory required to decide the satisfiability of the formula $\phi$.

B. Extension to infinite trajectories

For certain types of formulas, we can stitch together trajectories of length $T$ using a receding horizon approach, to produce an infinite computation that satisfies the STL formula. An example of this is safety properties, i.e. $\phi = \Box(\varphi_{MPC})$ for some STL formula $\varphi_{MPC}$. For such formulas, at each step of the MPC computation, we will search for a finite trajectory of horizon length $T$ that satisfies $\varphi_{MPC}$.

VI. ROBUSTNESS AS AN OBJECTIVE

TODO: Say robustness how it can be computed recursively while computing MILP constraints. Use it to solve Problem 5 in the absence or in complement of a different cost function $J$.

VII. CASE STUDY I: BUILDING CLIMATE CONTROL

In this case study we consider the problem of controlling building indoor climate. We use the model that is proposed in [13]. First we present a summary of the building thermal modeling. We then present an MPC formulation for the control of building HVAC system to minimize the energy consumption of a building while satisfying the state and input constraints.

A. Building Mathematical Model

1) Heat Transfer: As shown in Fig. 1, a building is modeled as a circuit in which there are two types of nodes: walls and rooms. $n$ is the total number of nodes, $m$ of which represent rooms and the remaining $n - m$ nodes represent walls. We denote the temperature of room $r_i$ with $T_{r_i}$. The wall node and temperature of the wall between room $i$ and $j$ are denoted by $(i, j)$ and $T_{w_{i,j}}$, respectively. Temperature of the $(i, j)$ wall, and the $i^{th}$ room are governed by the following equation:

$$ C^{w}_{i,j} \frac{dT_{w_{i,j}}}{dt} = \sum_{k \in N_{w_{i,j}}} T_k - T_{w_{i,j}} \frac{T_k}{R_{i,j,k}} + r_{i,j} \alpha_{i,j} A_{w_{i,j}} Q_{rad_{i,j}} $$

$$ C^{r}_{i} \frac{dT_{r_{i}}}{dt} = \sum_{k \in N_{r_{i}}} T_k - T_{r_{i}} \frac{T_k}{R_{i,k}} + m_{r_{i}} c_a (T_{s_{i}} - T_{r_{i}}) + w_{i} \tau_{w_{i}} A_{win_{i}} Q_{rad_{i}} + \dot{Q}_{int_{i}} $$

where $C^{w}_{i,j}$, $\alpha_{i,j}$ and $A_{w_{i,j}}$ are heat capacity, radiative heat absorption coefficient and area of wall between room $i$ and $j$, respectively. $R_{i,j,k}$ is the total thermal resistance between the centerline of wall $(i, j)$ and the side of the wall where node $k$ is located. $Q_{rad_{i,j}}$ is the radiative heat flux density on wall $(i, j)$. $N_{w_{i,j}}$ is the set of all of neighboring nodes to node $w_{i,j}$. $r_{i,j}$ is wall identifier which is equal to 0 for internal walls, and equal to 1 for peripheral walls (i.e. either $i$ or $j$ is the outside node). $T_{r_{i}}$, $C^{r}_{i}$ and $m_{r_{i}}$ are the temperature, heat capacity and air mass flow into the room $i$, respectively. $c_a$ is the specific heat capacity of air, and $T_{s_{i}}$ is the temperature of the supply air to room $i$. $\pi_{i}$ is window identifier which is equal to 0 if none of the walls surrounding room $i$ have window, and is equal to 1 if at least one of them has a window. $\tau_{w_{i}}$ is the transmissivity of glass of window $i$, $A_{win_{i}}$ is the total area of window on walls surrounding room $i$, $Q_{rad_{i}}$ is the radiative heat flux density per unit area radiated to room $i$, and $\dot{Q}_{int_{i}}$ is the internal heat generation in room $i$. $N_{r_{i}}$ is the set of all of the neighboring room nodes to room $i$.

More details on building thermal model can be found in [13].

The heat transfer equations for each wall and room yield
the following system dynamics:

\[ \dot{x}_t = f(x_t, u_t, d_t) \]
\[ y_t = Cx_t \]  \tag{7}

where \( x_t \in \mathbb{R}^n \) is the state vector representing the temperature of the nodes in the thermal network, \( u_t \in \mathbb{R}^m \) is the input vector representing the air mass flow rate and discharge air temperature of conditioned air into each thermal zone, and \( y_t \in \mathbb{R}^m \) is the output vector of the system which represents the temperature of the thermal zones. \( l \) is the number of inputs to each thermal zone (e.g. air mass flow and supply air temperature). \( C \) is a matrix of proper dimension.

Assuming linear relation between \( T_{out}, Q_{int}, Q_{rad} \) and the building temperature rise we approximate the disturbance to the model with an affine function of these quantities, leading to:

\[ d_t = \alpha_3 Q_{rad}(t) + \alpha_2 \dot{Q}_{int}(t) + \alpha_1 T_{out}(t) + \alpha_0 \]  \tag{8}

where \( \alpha_i, \forall i = 1, \ldots, 3 \) are parameters of the model to be estimated. We approximate the values of \( Q_{rad}(t) \) and \( \dot{Q}_{int}(t) \) using \( Q_{rad}(t) = \beta_1 T_{out}(t) + \beta_0 \) and \( \dot{Q}_{int}(t) = \gamma_1 \Psi(t) + \gamma_0 \), where \( \Psi(t) \) is level of \( CO_2 \) concentration in the room. By substituting for \( Q_{rad}(t) \) and \( \dot{Q}_{int}(t) \), and rearranging the terms, we obtain:

\[ d_t = \bar{\alpha}_2 T_{out}(t) + \bar{\alpha}_1 \Psi(t) + \bar{\alpha}_0 \]  \tag{9}

Therefore, only measurements of outside air temperature and \( CO_2 \) concentration levels are needed to determine the disturbance to the model. The values of \( \bar{\alpha}_2, \bar{\alpha}_1, \text{ and } \bar{\alpha}_0 \) are estimated along with other parameters of the model [13].

B. MPC for Building Climate Control

We consider a commercial building that has an HVAC system controlled by an MPC. We adopt the MPC formulation proposed in [11] that has the objective of minimizing the total energy cost (in dollar value). Let the time be slotted with \( \tau \) as the length of each time slot, and let \( H \) be the prediction horizon (in number of time-slots) of the MPC. System dynamics is also discretized with a sampling time of \( \tau \). Typical values for \( \tau \) and \( H \) range from 15 min to 1 hr for \( \tau \) and a few hours to a few days, e.g., 3 to 72 time slots (with the assumption of 1 hr for each time slot) for \( H \). The choice of \( \tau \) depends on how far in the future the estimation of the predicted values have an acceptable accuracy, and also on how far in the future the required information (e.g. cost of energy) is available. Here we consider \( \tau = 1 \) hr and \( H = 24 \) hrs. At each time \( t \), the predictive controller solves the following optimal control problem to compute the optimal vector \( \bar{u}_t = [u_t, \ldots, u_{t+H-1}] \), where \( u_{t+k} \) is the air mass flow to the thermal zones of the building. The inputs to the optimal control problem are the states (i.e. zone temperatures) \( x_t \) (as initial condition), the set of electric energy prices \( \{\pi_{t+1}, \ldots, \pi_{t+H-1}\} \), the non-electric and non-fan energy prices, such as gas price for heating \( \pi_{ne,h} \), and cooling \( \pi_{ne,c} \) which are considered time-invariant, a set of constraints \( X_{t+k} \) on the system states of the type: \( x_{t+k} \) should be in \( X_{t+k} \) for all times \( t+k \) where \( k \in \{1, \ldots, H\} \); a set of constraints \( U_{t+k} \) on system inputs for all \( k \in \{1, \ldots, H-1\} \), and an estimate on unmodelled disturbances \( d_{t+k} \) for all \( k \in \{0, \ldots, H-1\} \) for the next \( H \) time slots.

The optimization problem is as follows:

\[
\begin{align*}
\min_{\bar{u}_t} & \quad \sum_{k=0}^{H-1} C_{hvac}(u_{t+k}, \pi_{e,t+k}, \pi_{ne,c}, \pi_{ne,h}, \pi_{out}) \\
\text{s.t.} \quad & x_{t+k+1} = f(x_{t+k}, u_{t+k}, d_{t+k}), \quad k = 0, \ldots, H - 1 \\
& x_{t+k} \in X_{t+k}, \quad k = 1, \ldots, H \\
& u_{t+k} \in U_{t+k}, \quad k = 0, \ldots, H - 1
\end{align*}
\]  \tag{10}

where \( f \) captures the system dynamics, \( T_{ref} \) is a reference temperature and \( ||x||_A = x^T Q x \). The total HVAC power consumption cost \( C_{hvac}(t) \) is the summation of fan power, cooling power and heating power, given by:

\[
C_{hvac}(t) = \pi_e(t) P_f(t) + \pi_{ne,c} P_c(t) + \pi_{ne,h} P_h(t)
\]  \tag{11}

where with the assumption of no recirculation of air, and without loss of generality, these power consumptions for time slot \([t, t+1]\) (with the assumptions of constant air mass flow and price of energy over \([t, t+1]\)) are calculated as follows:

\[
\begin{align*}
P_f(u_t) &= c_1 u_t^4 + c_2 u_t^2 + c_3 u_t + c_4 \\
P_c(u_t, T_{out}) &= c_5 u_t (T_{out} - T^*) / COP_c \\
P_h(u_t, T_{out}) &= c_6 u_t (T_{out} - T^*) / COP_h
\end{align*}
\]  \tag{12-14}

where \( c_1, c_2, c_3, \text{ and } c_4 \) are constants of the fan, \( c_p \) is the specific heat of air, and \( COP_h, \text{ and } COP_c \) are the coefficients of performance for the heating system and the cooling system, respectively, and the supply air temperature \( T^* \) is considered constant. To move the coolant fluid around, heating and cooling systems use pumps which consume electric power. However, we assume that electric power consumption of pumps is negligible compared to the non-electric heating and cooling powers of these systems.

We consider terminal cost and terminal constraint to guarantee stability and feasibility of the RHC. In the specific context of the MPC problem considered here, the terminal set ensures that sufficient energy is always stored in thermal storage elements (walls and room air) to counteract an unpredicted change in the supplied cooling or heating energy.

VIII. CASE STUDY II: REGULATION CONTROL FOR SMART GRID

The second case study we consider is the smart grid building model of the power system presented in [12]. The interconnection of power system components, including a governor, turbine and generator, is shown in the block diagram in Figure 2. In the diagram, \( \delta P_C \) is a control input which acts against an increase or decrease in power demand to regulate the system frequency, and \( \delta P_D \) denotes fluctuations in power demand, modeled as an exogenous input (disturbance).
A. Two-Area System Model

We consider a two-area interconnected system consisting of two buses connected by a tie line with reactance $X_{tie}$. The power flow on the tie line from area 1 to area 2 is denoted by $P_{tie}^{j}$. A positive $\Delta P_{tie}^{j}$ represents an increase in power transfer from area 1 to area 2. This is equivalent in effect to increasing the load of area 1 and decreasing the load of area 2. Each area consists of the subsystems shown in Figure 2. Next, we present the mathematical model of the two-area system. Note that for states, $x^{i}$, the superscript refers to the control area (i.e., $i = 1$, 2), and the subscript indexes the state in each area.

\begin{align}
\frac{dx_{1}^{i}}{dt} &= (D_{x}^{i}x_{1}^{i} + \Delta P_{D}^{i} - \Delta P_{tie}^{i} + \Delta P_{anc}^{i}) \\
\frac{dx_{2}^{i}}{dt} &= (x_{2}^{i} - x_{1}^{i})/T_{i}^{2} \\
\frac{dx_{4}^{i}}{dt} &= (x_{4}^{i} - x_{3}^{i})/T_{i}^{2} \\
\frac{dx_{5}^{i}}{dt} &= (P_{GV}^{i} - x_{5}^{i})/T_{i}^{2} \\
\frac{dx_{6}^{i}}{dt} &= (x_{6}^{i} - x_{4}^{i})/T_{i}^{2} \\
\frac{dx_{7}^{i}}{dt} &= (x_{7}^{i} - x_{1}^{i}/R^{i})/T_{i}^{2} \\
\frac{dx_{8}^{i}}{dt} &= (x_{8}^{i} - x_{2}^{i}/R^{i})/T_{i}^{2}
\end{align}

where $\Delta P_{M}^{i}$ and $P_{GV}^{i}$ are given by $\Delta P_{M}^{i} = K_{i}^{1}x_{1}^{i} + K_{i}^{2}x_{2}^{i} + K_{i}^{3}x_{3}^{i} + K_{i}^{4}x_{4}^{i}$, and $P_{GV}^{i} = (1 - T_{2}^{i}/T_{3}^{i})x_{5}^{i} + T_{2}^{i}/T_{3}^{i}x_{6}^{i}$. $D_{x}$ is the damping coefficient, $M$ is the machine inertia constant, $R$ is the speed regulation constant, $T_{i}$’s are time constants for power system components, and $K_{i}$’s are fractions of total mechanical power outputs associated with different operating points of the turbine. In formulation (15), the first state represents the frequency increment, $x_{1}^{i} = \delta_{x}^{i}$. All state dynamics are derived using the mathematical model of each subsystem, as presented in [12]. The state space model (15) can be discretized and written in compact form as:

\begin{equation}
x[k + 1] = Ax[k] + Bu_{\text{cont}}[k] + B_{2}u_{\text{anc}}[k] + Ed[k].
\end{equation}

We use this state update equation in Section VIII-C, where we present the MPC formulation. Input signals are $u_{\text{cont}} = [\Delta P_{M}^{1}, \Delta P_{M}^{2}]^{T}$, the ancillary inputs are $u_{\text{anc}} = [\Delta P_{anc}^{1}, \Delta P_{anc}^{2}]^{T}$, and the exogenous inputs (i.e., disturbances or variations in demands) are denoted by $d = [\Delta P_{tie}^{1}, \Delta P_{tie}^{2}]^{T}$.

B. Automatic Generation Control

In the classical AGC, a simple PI control is utilized to regulate the frequency of the grid. The Area Control Error (ACE) is defined as $ACE^{i} = \Delta P_{tie}^{i} + \beta_{i}x_{1}^{i}$, where $\Delta P_{tie}^{i} = P_{tie}^{i} - P_{tie}^{i_{\text{in scheduled}}}$, and $\beta_{i}$ is the bias coefficient of area $i$. The standard industry practice is to set the bias $\beta_{i}$ at the so-called Area Frequency Response Characteristic (AFRC), which is defined as $\beta_{i} = D_{i}^{i} + 1/R_{i}^{i}$. The integral of ACE is used to construct the speed changer position feedback control signal ($\Delta P_{C}^{i}$). In other words, the control input $\Delta P_{C}^{i}$ is given by $\Delta P_{C}^{i} = -K_{i}^{1}x_{i}^{i}$, where $K_{i}^{1}$ is the feedback gain and $\Delta x_{i}^{i} = ACE^{i}$. We propose a methodology for the ancillary services complementing the primary control of AGC, as described in VIII-C.

C. MPC for Ancillary Services

We present an MPC scheme to control the ancillary service to improve on the classical AGC practice. This optimization-based control framework is utilized as a higher-level control in a “hierarchical” scheme on top of the low-level classical AGC control [12]. We require that $u_{\text{anc}}$ satisfies $u_{\text{anc}}[k + j|k] \leq \pi_{\text{anc}}$ for some $u_{\text{anc}} < 0$ and $\pi_{\text{anc}} > 0$, and a maximum ramp constraint:

\begin{equation}
|u_{\text{anc}}[k + 1] - u_{\text{anc}}[k]| \leq \lambda, \text{for some } \lambda > 0.
\end{equation}

At each time step $k$, we thus solve the following problem:

\begin{equation}
\min_{u_{\text{anc}}[k]} \ J(ACE, U_{\text{anc}}) + ||x[k + H|k] - x_{\text{ref}}||_{Q}
\end{equation}

s.t.

\begin{equation}
x[k + j + 1|k] = Ax[k + j|k] + B_{2}u_{\text{anc}}[k + j|k] + Ed[k + j|k]
\end{equation}

\begin{equation}
u_{\text{anc}}[k + j|k] \leq \pi_{\text{anc}}
\end{equation}

\begin{equation}
u_{\text{anc}}[k + j + 1|k] - u_{\text{anc}}[k + j|k] \leq \lambda
\end{equation}

\begin{equation}x[k + H|k] \in X_{H}
\end{equation}

where

\begin{equation}U_{\text{anc}}[k] = \{u_{\text{anc}}[k|k], u_{\text{anc}}[k + 1|k], \ldots, u_{\text{anc}}[k + H - 1|k]\}
\end{equation}

is the vector of inputs from $k$ to $k + H$ and $H$ is the prediction horizon. The notation $x[k + j|k]$ denotes that predictions of $x$ for future times $k + j$ are obtained at each time step $k$. All the constraints of problem (18) that depend on $j$ should hold for $j = 0, 1, \ldots, H - 1$.

The cost function proposed in [12] minimizes the $\ell_{2}$ norm of the ACE signal in areas $i = 1, 2$, by exploiting the ancillary service available in each area, while taking into account the system dynamics and constraints. We propose to constrain the ACE signal to satisfy a specified set of STL properties, while minimizing the ancillary service used by each area. Thus we defined $J(ACE, U_{\text{anc}}) = \|U_{\text{anc}}\|_{\ell_{2}} = \sum_{i=1}^{2} \sum_{j=0}^{H-1} |U^{i}_{\text{anc}}[k + j|k]|2$, and an STL formula $\varphi$ which says that whenever $|ACE^{i}|$ is larger than 0.01, it should become less than 0.01 in less than $\tau$ s. More precisely we used $\varphi = \Box(\varphi_{t})$ with

\begin{equation}
\varphi_{t} = -((|ACE^{i}| < 0.01)) \Rightarrow (\Diamond_{[0, \tau]}(|ACE^{i}| < 0.01) \wedge (-(|ACE^{i} < 0.01)) \Rightarrow (\Diamond_{[0, \tau]}(|ACE^{i}| < 0.01)
\end{equation}

We encoded this formula and added the resulting constraints to the MPC problem as described in the previous sections, and solved it for different values of $\tau$. Results are shown in Figure 3, and demonstrate that the STL constraint is correctly enforced in the stabilization of the ACE signal.
IX. DISCUSSION

TODO: reword, include discussion of extensions

The main contributions of this paper are a BMC-inspired encoding for STL specifications as mixed integer linear constraints. We showed how this encoding can be used to generate signals that satisfy finite and infinite horizon STL properties in an open-loop fashion, and moreover to generate signals that are maximally robust. We also demonstrated how our MILP formulation of the STL synthesis problem can be used as part of existing controller synthesis frameworks like model predictive control, to compute feasible and optimal controllers for cyber-physical systems that must satisfy timed and temporal properties. We presented experimental results for controller synthesis on simplified models of a smart building-level micro-grid and HVAC system, and showed how the MPC schemes for controlling the ancillary service power flow and demand in these examples can be framed in terms of synthesis from an STL specification, and present simulation results to illustrate the effectiveness of our proposed synthesis methodology.

Future work: summarize history, formula decomposition, ties to online monitoring for STL. Contract-based design (if the two systems satisfy STL specs, what can we guarantee when they interact?).

REFERENCES


