Scalable Scheduling of Energy Control Systems

ABSTRACT

Peak power consumption is a universal problem across energy control systems in electrical grids, buildings, electric vehicles and industrial automation where the uncoordinated operation of multiple controllers result in temporally correlated electricity demand surges (or peaks). While there exist several different approaches to balance power consumption by load shifting and load shedding, they operate on coarse grained time scales and do not help in de-correlating energy sinks. The focus of this paper is on a a scalable approach for "Green Scheduling", fine-grained scheduling of control systems within an aggregate peak power envelop while ensuring the individual controllers are maintained within the desired ranges. What makes the green scheduling problem particularly hard is that its control variables are binary. The complexity grows exponentially with the scale of the system, making it impossible to handle systems with more than a few dozens variables, unless the simple periodic scheduling algorithm is used. While simple and scalable, periodic schedules are static and limited in their capability to handle dynamics, safety constraints, and disturbances.

The new results presented here novelly combine techniques from control theory and computer science to solve the green scheduling problem for large-scale systems with hundreds or thousands subsystems. The original system with binary control variables are approximated by an *averaged system* whose inputs are the *utilization values* of the binary inputs within a given period. The error between the two systems can be bounded, which allows us to derive a safety constraint for the averaged system so that the original system's safety is guaranteed. This relaxation already reduces the green scheduling mixed integer program to a standard program with real variables. However, for a large-scale system over a long prediction horizon, the number of optimization variables can be prohibitively large, which not only affects the computation performance but also requires commercial optimization solvers and powerful computing hardware. To alleviate this issue, we abstract the averaged system by a simple scalar dynamical system whose control input is the total demand of all the subsystems. This model abstraction is achieved by extending the notion of simulation relations between transition systems to allow for input constraints between the systems. We develop conditions to test for simulation relations as well as algorithms to compute such a model abstraction. As a consequence, we only need to solve a small linear program to compute an optimal trace of the total demand. The total demand is then broken down, by solving a linear program much smaller than the original program, to individual utilization values of the subsystems in an interval. The actual schedule in each interval is then obtained by a low-level scheduling algorithm. We propose two such algorithms: a simple ad-hoc algorithm that may switch on and off a subsystem multiple times, and a 2-dimensional packing algorithm that only switches any subsystem at most once in each interval.

Numerical simulations in Matlab show the effectiveness and scalability of our approach.

General Terms

Theory

Keywords

embedded control systems, peak power management, cyber-physical systems

1. INTRODUCTION

Peak power consumption is a universal problem across energy control systems in electrical grids, buildings, electric vehicles and industrial automation where the uncoordinated operation of multiple controllers result in temporally correlated electricity demand surges. In the case of the electrical grid, when the popular UK TV soap Eastenders comes to an end five times a week, the grid has to deal with around 1.75 million kettles requiring power at the same time (to prepare tea). ThatÕs an additional 3 gigawatts of power for the roughly 3-5 minutes it takes each kettle to boil. So big is the surge, caused by correlated human behavior, that backup power stations have to go on standby across the country, and thereÕs even additional power made available in France just in case the UK grid canÕt cope [1].

In the case of building systems such as heating, ventilating, air conditioning and refrigeration (HVAC&R) systems, chiller systems, and lighting systems operate independently of each other and frequently trigger concurrently, resulting

in temporally correlated power demand surges. Most commercial buildings are subject to peak demand pricing which can be 200-400 times that of the nominal power rate [2]. High peak loads also lead to a higher cost of production and distribution of electricity and lower reliability. Therefore, peaks in electricity usage are inefficient and expensive for both suppliers and customers.

In the case of electric and hybrid-electric vehicles, peak power consumption due to vehicle acceleration or hilly terrain results in a high current draw from the battery [3]. This increases the operating temperature which reduces the battery lifetime and capacity and requires additional resources for cooling. Effectively managing hybrid energy sources to minimize peak power is an open research issue for electric vehicles with fast dynamics and limited load forecasts [4].

While there exist several different approaches to balance power consumption by load shifting and load shedding, they operate on coarse grained time scales and do not help in de-correlating energy sinks. The focus of this paper is on a a scalable approach for "Green Scheduling", fine-grained scheduling of control systems within an aggregate peak power envelop while ensuring the individual controllers are maintained within the desired ranges.

While traditional real-time scheduling algorithms [5] may be applied to such resource sharing problems, they impose stringent constraints on the task model. Generally, real-time scheduling is restricted to tasks whose worst case execution times are fixed and known a priori [6]. While this simplifies the runtime complexity, for control systems it does not effectively capture the system's behavior whose operation is dependent on the plant dynamics and environmental conditions.

The contributions of this paper

2. ENERGY SYSTEM SCHEDULING

This section can be placed inside the Introduction. NEED TO BE REWRITTEN TO BE SELF-CONTAINED, NO REFERENCE TO PREVIOUS WORK.

We consider a variant of the original scheduling problem where the peak demand is determined based on the interval averaged total demand. Specifically, time is divided into equal intervals: $[0,T],[T,2T],\dots$ In each interval $[kT,(k+1)T]$ the averaged total demand is calculated as $d_k = \frac{E_k}{T}$ where E_k is the total energy consumption during interval k. This is the practical way to determine energy demands, for example in electricity bills and in demand response (DR) programs. The peak demand, for example to be used in calculating the demand charge, is the maximum interval demand over a given billing period: $\max_k d_k$. Our goal is to schedule the subsystems to minimize the peak interval demand while maintaining a safe operation. At the same time, it is desirable to schedule the subsystems in each interval so that the peak instantaneous total demand is also reduced, although this will not affect the demand charge.

As in our previous work, we consider a continuous-time linear system Σ_b of the form

$$(\Sigma_b) \qquad \dot{x} = Ax + Bu + Ew$$

where $x \in \mathbb{R}^n$ is the state vector, $u \in \{0,1\}^m$ are the binary control inputs that represent the schedules, and $w \in \mathbb{R}^p$ is the disturbance vector. The safety condition requires that the state $x(t)$ stays in a bounded safe set $\mathcal{X} \subset \mathbb{R}^n$ at all time. The disturbances are constrained in a known set $w(t) \in \mathcal{W}$ for all t. Both $\mathcal{X}$ and $\mathcal{W}$ are polyhedral.

At any time, the total demand is defined as a linear combination of the binary control inputs: $d(t) = \rho^T u(t)$ where ρ is the vector of the individual power demand of each subsystem. In each time interval $[kT,(k+1)T]$ the total energy consumption is $E_k = \int_{kT}^{(k+1)T} d(t)\mathrm{d}t$ and the averaged total demand is $d_k = \frac{E_k}{T}$. In this scheduling problem, we aim to schedule the subsystems, i.e., to compute the control inputs u, so that the energy cost over a finite horizon is minimized. The energy cost consists of a charge for the peak demand and possibly a charge for the energy consumtion. Specifically, we want to minimize the following cost function defined over a horizon of N time intervals:

$$\text{cost} = c_d \cdot \max_{0\leqslant k\leqslant N-1} d_k + \textstyle\sum_{0\leqslant k\leqslant N-1} c_{e,k}\cdot E_k \qquad (1)$$

where c_d is the fixed price for the peak demand and $c_{e,k}$ is the time-varying price for the interval energy consumption. Typically $c_d \gg c_{e,k}$ which gives customers incentive to reduce their peak demands.

The scheduling problem can be formulated as minimizing the above cost function subject to the constraints:

$$\begin{aligned}
&\dot{x}(t) = Ax(t) + Bu(t) + Ew(t)\\
&x(t) \in \mathcal{X}, \quad u(t) \in \{0,1\}^m, \quad w(t) \in \mathcal{W}\\
&E_k = \int_{kT}^{(k+1)T} \rho^T u(\tau)\mathrm{d}\tau,\ d_k = \frac{E_k}{T}
\end{aligned}$$

for all t and all $k = 0,\dots,N-1$. This optimization is intractable because $u(\cdot)$ is infinite dimensional. If we discretize Σ_b with sampling time T and assume that $u(t)$ and $w(t)$ are constant in each time interval, the optimization becomes:

$$\begin{aligned}
\underset{u_0,\dots,u_{N-1}}{\text{minimize}} \quad & c_d \cdot \max_{0\leqslant k\leqslant N-1} d_k + \textstyle\sum_{0\leqslant k\leqslant N-1} c_{e,k}\cdot E_k \qquad (2)\\
\text{subject to} \quad & x_{k+1} = A_T x_k + B_T u_k + E_T w_k\\
& d_k = \rho^T u_k\\
& x_k \in \mathcal{X}, \quad u_k \in \{0,1\}^m, \quad w_k \in \mathcal{W}\\
& E_k = T d_k
\end{aligned}$$

where the constraints hold for all $k = 0,\dots,N-1$. Here, the subscript k denotes the value of a variable at time step k, and matrices A_T, B_T, and E_T are of the discrete-time dynamical model. Note that the safety condition, that x_k stays inside $\mathcal{X}$ at all k, must be robust to the unknown but bounded disturbances w, and thereby results in conservative control inputs trading off performance for safety. In practice, disturbance forecasts are usually available, for example in the forms of weather forecast and occupancy schedules, which should be exploited to obtain more accurate predictions of future states and hence less conservative control inputs and better performance. Therefore, the disturbances are modeled as $w_k = \tilde{w}_k + \delta_k$ where $\tilde{w}_k \in \mathcal{W}$ is the forecast and δ_k is the forecast uncertainty. The forecast accuracy, as a bounded set of δ_k, is assumed to be known and certainly smaller than $\mathcal{W}$. The optimization (2) is a mixed-integer linear program (MILP) and can be solved by an MILP solver. However, except for small-scale systems with only a few control inputs and a short horizon N, the number of binary variables can be prohibitively large and the MILP is difficult to solve. For example, if Σ_b has $m = 20$ control inputs with sampling time $T = 5$min for a horizon of $N = 288$ steps

in 24 hours, the MILP will have 5,760 binary variables, not to mention the continuous state and disturbance variables. Solving such large MILPs often requires powerful computers with commercial optimization solvers. Moreover, due to the uncertainty caused by the disturbances, the control decisions need to be adjusted regularly by solving the optimization (2) repeatedly at every time step (model predictive control (MPC)). Therefore, for any practical size of the system, implementing this approach can be highly demanding, if even possible, in terms of run-time hardware and software requirements. Although there are techniques in MPC to reduce the complexity, e.g., move blocking, an MILP solver is still required. Certainly the controller cannot be implemented on an embedded processor with limited processing power and memory.

Our goal in this paper is to develop a scalable scheduling algorithm that can potentially run on embedded processors, even for systems of moderate practical size. We consider a slightly more general problem than (2):

$$\begin{aligned} \text{minimize}_{u(\cdot)} \quad & c_d \cdot \max_{0\leqslant k\leqslant N-1} d_k + \textstyle\sum_{0\leqslant k\leqslant N-1} c_{e,k}\cdot E_k \qquad (3)\\ \text{subject to} \quad & \dot{x}(t) = Ax(t) + Bu(t) + Ew(t) \quad \forall t \geq 0\\ & u(t) \in \{0,1\}^m \quad \forall t \geq 0\\ & x_k = x(kT) \in \mathcal{X}\\ & w(t) = w_k \in \mathcal{W} \quad \forall t \in [kT, (k+1)T)\\ & E_k = \int_{kT}^{(k+1)T} \rho^T u(\tau)\mathrm{d}\tau, \quad d_k = \frac{E_k}{T} \end{aligned}$$

where the constraints hold for all $k = 0, \ldots, N-1$. Problem (3) is more general than (2) because we consider a continuous-time schedule $u(\cdot)$, which is not necessarily constant during each time interval.

3. OVERVIEW OF THE APPROACH

We develop in this work a hierarchical design and run-time implementation approach for the above scheduling problem of energy control systems. A brief overview of the overall idea and results is described in this section and in fig. 1, while the technical details will be presented in the subsequent sections.

The complexity of the scheduling problem (3) comes mainly from the *binary* control inputs u. To alleviate this issue, we approximate the original model Σ_b by the averaged system Σ_a with state $\bar{x}$ for each time interval:

$$(\Sigma_a) \qquad \dot{\bar{x}}(t) = A\bar{x}(t) + B\eta_k + Ew(t), \quad t \in [kT, (k+1)T)$$

where the control input $\eta_k = \frac{1}{T}\int_{kT}^{(k+1)T} u(t)\mathrm{d}t$ is the *average*, also called the *utilization*, of $u(t)$ during the interval $[kT, (k+1)T]$. Clearly, η_k is a vector of real numbers between 0 and 1: $\eta_k \in [0,1]^m$. Because the control input η_k of Σ_a is continuous, if we use Σ_a in place of Σ_b, the complexity of the optimization is greatly reduced to that of an LP. However, to ensure the safety constraint, we need to bound the deviation of $\bar{x}(kT)$ from $x(kT)$ for all possible continuous-time binary input signals $u(t)$ that satisfy the utilization equation. In section 4, we will derive a tight bound on the error between x and $\bar{x}$. As we will see, the error can be unbounded or become too large as k increases, leading to overly conservative control inputs to ensure safety, or even infeasibility. To keep this error as small as possible, we reset the state $\bar{x}$ of Σ_a to the measured state x of Σ_b at each kT, so that their error is

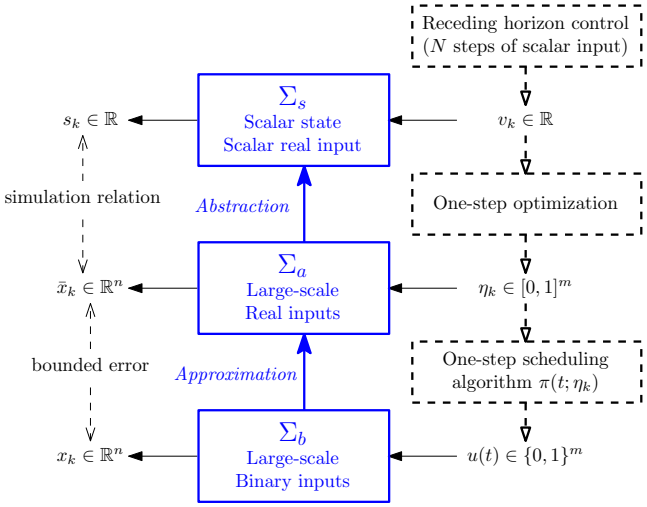

Figure 1: Overview of the scheduling approach. The blue boxes represent the different models at different scales resulted from the offline design process: the original large-scale binary-input model Σ_b is reduced to the single-state single-input model Σ_s. The dashed boxes represent the run-time components, which include two lightweight optimizations and one simple scheduling algorithm.

reset to 0. Since the input η_k is constant in each interval, Σ_a can be discretized with sampling time T. With the averaged system in place, we can reformulate the optimization (3) as minimizing the cost function (1) subject to

$$\dot{x}(t) = Ax(t) + B\pi(t;\eta_k) + Ew_k \quad \forall t \in [kT, (k+1)T) \qquad (4a)$$

$$A_T\bar{x}_k + B_T\eta_k + E_Tw_k + e_{k+1} \in \mathcal{X} \quad \forall e_{k+1} \in \mathcal{E}(\eta_k) \qquad (4b)$$

$$\bar{x}_k = x(kT) \qquad (4c)$$

$$\eta_k \in [0,1]^m, \quad w_k \in \mathcal{W}, \quad E_k = Td_k, \quad d_k = \rho^T\eta_k \qquad (4d)$$

where the constraints hold for all $k = 0, \ldots, N-1$. In the continuous-time dynamics (4a), $\pi(t;\eta_k)$ is a scheduling algorithm that calculates the continous-time schedule $u(t)$ during the k-th interval so that its utilization is η_k. Such a scheduling algorithm is the topic of section TODO: ???Update??? At each instant kT the state $\bar{x}_k$ of the averaged system is reset to the actual state $x(kT)$ (constraint (4c)). Let e_{k+1} be the error between $x((k+1)T)$ and $\bar{x}_k$, which is unknown but bounded in the η_k-dependent set $\mathcal{E}(\eta_k)$ (cf. section 4). To ensure that $x(kT) \in \mathcal{X}$, constraint (4b) robustly restricts the next state in $\mathcal{X}$ for all possible errors e_{k+1}.

In problem (4), the number of optimization variables $\eta_k \in \mathbb{R}^m$, $k = 0, \ldots, N-1$, is mN, which can be large for a large-scale system over a long horizon. For instance, with $m = 20$ control variables and sampling time $T = 5$min over a 24-hour horizon, the number of optimization variables is 5,760. Constraint (4b) involves high-dimensional dynamical equations and set operations over a long horizon, resulting in many constraints. More importantly, constraint (4a) requires execution of a continuous-time dynamical model. For these reasons, problem (4) is still difficult to solve, and its computational burden in run-time is still prohibitive for systems of practical size over a long horizon[1]. An embedded

[1]In our targeted energy control applications, the horizon is

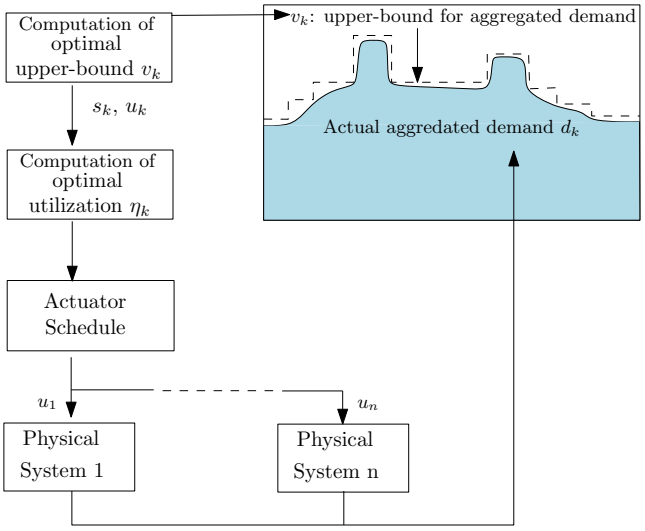

Figure 2: BlaBla

implementation of the controller is therefore unlikely realizable. To reduce the complexity even further, we employ a model abstraction technique based on the concept of simulation relations between transition systems [7, 8].

A transition system is a generalization of a dynamical system, whose state can transition into a new state either autonomously or under the influence of exogenous inputs. Intuitively speaking, a transition system $\mathcal{T}_2$ is said to *simulate* another transition system $\mathcal{T}_1$ if every move of $\mathcal{T}_1$ can be tracked by $\mathcal{T}_2$ with respect to a symbolic relation between their states [8]. Formal definitions of transition systems and simulation relations will be presented in section 5. The original definitions of a simulation relation in the literature do not take into account any relation between the two systems' inputs. In this paper, we extend this concept so that $\mathcal{T}_2$ simulates $\mathcal{T}_1$ while satisfying not only its state and input constraints but also a constraint between its input and that of $\mathcal{T}_1$.

Suppose that there exists a scalar discrete-time system:

$$(\Sigma_s) \quad s_{k+1} = \alpha s_k + v_k + \beta^T w_k$$
$$s_k \in \mathcal{S} \subseteq \mathbb{R}, v_k \in \mathcal{V} \subseteq \mathbb{R}, (s_k, v_k) \in \Omega \subseteq \mathcal{S} \times \mathcal{V}$$

and a relation $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{S}$ such that Σ_a simulates Σ_s subject to the constraint $T\rho^T \eta_k \leqslant v_k$ between their inputs. The intuition is that s_k represents an aggregated state of all the energy subsystems at time step k, and v_k is the total energy demand input to the system (more precisely, an upper-bound thereof). Effectively, the scalar system Σ_s abstracts the higher-dimensional system Σ_a by aggregating all the energy states and inputs. The constraints $\mathcal{S}$, $\mathcal{V}$, and Ω are to ensure compatibility with the state and input constraints of Σ_a. We will show that any admissible trajectory of Σ_s can be tracked by an admissible trajectory of Σ_a with equal or less energy cost (with respect to the cost function (1)). Therefore, instead of solving the large-scale optimization (4), we can optimize the sequence of $\{v_0, \ldots, v_{N-1}\}$ for the scalar system Σ_s, then recover the input η_k by solving a simple optimization for each time step k. The benefit of our approach is that all the involved optimization programs have much fewer variables and much less complexity, hence it is scalable. This advantage is more significant when a receding horizon control approach is employed to compute η_k one step at a time. In that case, we only solve two small-scale programs at each step, compared to a large-scale and complex program if we solve (4) directly. Details will be presented in the subsequent sections.

Our approach is best illustrated by fig. 1. There are two processes involved:

- In the **offline** controller design process (blue boxes and text, left side, from bottom up) we start with the original system model Σ_b and approximate it with the averaged system model Σ_a. The error bounds between the states of the two are also calculated as functions of η_k. Then Σ_a is abstracted by a scalar system Σ_s, and a simulation relation $\mathcal{R}$ between them is constructed.

- In the **online** controller execution process (dashed boxes, right side, from top down) a receding horizon control framework is used with the abstract model Σ_s to compute an optimal sequence of aggregate energy demands. Each energy demand v_k is disaggregated into utilization values η_k for the individual control inputs by a one-step optimization. A simple one-step scheduling algorithm π schedules the subsystems so that each achieves the corresponding utilization.

4. APPROXIMATION BY AVERAGING

Consider the averaged system Σ_a of Σ_b for each time interval $t \in [kT, (k+1)T)$: $\dot{\bar{x}}(t) = A\bar{x}(t) + B\eta_k + Ew(t)$. Recall that the *utilization* $\eta_k = \frac{1}{T}\int_{kT}^{(k+1)T} u(\tau)\mathrm{d}\tau$ is the average value of $u(t)$ in the interval, and is a vector of real numbers in $[0, 1]$. Initially $\bar{x}_0 = \bar{x}(0) = x(0)$. Generally, as k increases, $\bar{x}_k$ may deviate further and further from $x(kT)$, which is undesirable since the state constraint $\mathcal{X}$ needs to be maintained at all time. For this reason, as mentioned earlier, we reset the value of $\bar{x}$ to the measured state x at each time instant kT, to keep the deviation within a tight bound. The deviation $e(t) = x(t) - \bar{x}(t)$ follows the dynamics $\dot{e}(t) = Ae(t) + B(u(t) - \eta_k)$ for all $t \in [kT, (k+1)T)$ and $e(kT) = 0$. At the next instant $(k+1)T$, the error is given by the solution of this differential equation:

$$e_{k+1} = e((k+1)T) = \int_{kT}^{(k+1)T} \mathrm{e}^{A((k+1)T-s)} B(u(s) - \eta_k)\mathrm{d}s.$$

A tight bound on e_{k+1} can be obtained, which is dependent on the value of η_k. In this section, we make a practical assumption that the state matrix A is diagonalizable with real eigenvalues, that is $VA = DV$ where D is a diagonal matrix of the eigenvalues of A and V is a non-singular matrix whose columns are the corresponding right eigenvectors. We can then transform the state with V by defining a new state $z = Vx$, whose dynamics are

$$\dot{z} = V\dot{x} = VAx + VBu + VEw = Dz + VBu + VEw.$$

This new model is in the form of (Σ_b) with a diagonal state matrix. Let λ_i be the i-th diagonal element of D, for $i = 1, \ldots, n$. The following theorem provides bounds on e_{k+1}, which depend on η_k.

THEOREM 1. *Let b_i be the i^{th} row of B, $\bar{b}_i \geqslant 0$ the sum of all positive elements of b_i, and $\underline{b}_i \leqslant 0$ the sum of all negative*

typically a day, even a week or longer.

elements of b_i. For each $i = 1, \ldots, n$, define

$$\xi_i = \begin{cases} 0 & \text{if } \lambda_i = 0 \\ \frac{1}{\lambda_i}(\mathrm{e}^{\lambda_i T} - 1) - T & \text{if } \lambda_i > 0 \\ \frac{1}{\lambda_i}(\mathrm{e}^{\lambda_i T} - 1) - T\mathrm{e}^{\lambda_i T} & \text{if } \lambda_i < 0 \end{cases}$$

and $\bar{\varepsilon}_i = \xi_i \bar{b}_i$, $\underline{\varepsilon}_i = \xi_i \underline{b}_i$. Define $\tilde{B} = V^{-1}\Xi B$ where Ξ is the diagonal matrix of all ξ_i. Then the error e_{k+1} is bounded by

$$\underline{\varepsilon} \leqslant V(e_{k+1} + \tilde{B}\eta_k) \leqslant \bar{\varepsilon} \quad \textit{(element-wise)}$$

where $\bar{\varepsilon}$ and $\underline{\varepsilon}$ are the vectors of all $\bar{\varepsilon}_i$ and $\underline{\varepsilon}_i$ respectively. Equivalently, $e_{k+1} = \varepsilon - \tilde{B}\eta_k$ for some ε bounded element-wise by $\underline{\varepsilon} \leqslant \varepsilon \leqslant \bar{\varepsilon}$.

The proof can be found in appendix A.1. We note that the utilization η_k, i.e., the control input of Σ_a, enters affinely in the bounds on e_{k+1}.

REMARK 1. *We can still use our approach when A is not diagonalizable, however the error bounds obtained in theorem 1 must be generalized. For instance, we can use the Jordan normal form to transform A into a block diagonal matrix, and obtain bounds in a way similar to that in theorem 1. The made assumption simplifies the presentation of our results but does not limit the practicality of our approach.*

5. MODEL ABSTRACTRION WITH SIMULATION RELATIONS

As we discussed in section 3, to further reduce the optimization program (4), we employ the notion of model abstraction and simulation relations. We first distinguish model abstraction and model order reduction. Model order reduction is a common technique in control theory to reduce the order of a large-scale system so that it becomes manageable. The reduced model, however, still has the same input and output as the original model. Under the same input signal, the reduced model should produce an output signal which approximates that of the original model, within some error bound. The left diagram in fig. 3 illustrates this concept, where $\mathcal{M}_2$ is a reduced-order model of $\mathcal{M}_1$. On the other hand, model abstraction derives a new model, usually of a lower order, with completely different input and output than the original model. However, there exists a relation between the states of the two models that can be maintained along their evolutions. In other words, one model can track the behavior of the other under this relation. The right diagram in fig. 3 depicts two models with a symbolic relation $\mathcal{R}$ between their states. If for any admissible input $u_2(\cdot)$ to $\mathcal{M}_2$ there exists an admissible input $u_1(\cdot)$ to $\mathcal{M}_1$ so that the state relation is always maintained along the traces of the two models, we have a model abstraction. Furthermore, we may require that a symbolic relation $\mathcal{R}_y$ between their outputs (observations) is also maintained. A nice property of model abstraction is that we can design a controller for $\mathcal{M}_1$ by first designing a supposedly simpler controller for $\mathcal{M}_2$ and then refining it for $\mathcal{M}_1$ using the symbolic state relation.

5.1 Simulation Relations

In this section we review the concept of simulation relations between transition systems. The readers are referred to [8, 7] for more thorough treatments of the subject.

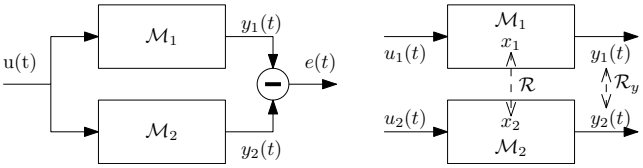

Figure 3: Model abstraction (right) is different from model order reduction (left): the models $\mathcal{M}_1$ and $\mathcal{M}_2$ do not share the same inputs nor outputs; instead they maintain a relation between their states along their evolutions.

The framework of transition systems allows us to unify the modeling of discrete and continuous, deterministic and non-deterministic dynamical systems. As we do not concern with the observable outputs of the systems in this paper, we remove the observation aspect from the definition of transition systems in [8]:

DEFINITION 1 (TRANSITION SYSTEM, [8]). *A transition system is a tuple $T = (Q, \Sigma, \rightarrow, Q^0)$ that consists of*

- *a possibly infinite set Q of states;*
- *a possibly infinite set Σ of labels;*
- *a transition relation $\rightarrow \subseteq Q \times \Sigma \times Q$;*
- *a possibly infinite set $Q^0 \subseteq Q$ of initial states.*

A transition from state q to state q^+ under the label σ, i.e., $(q, \sigma, q^+) \in \rightarrow$, is denoted $q \xrightarrow{\sigma} q^+$. We assume that the transition systems are *nonblocking*, meaning that for all $q \in Q$, there exists at least one transition starting from q. If for any $q \in Q$ and any $\sigma \in \Sigma$, there is at most one transition $q \xrightarrow{\sigma} q^+$ of T and Q^0 is a singleton, then T is called *deterministic*. Otherwise, it is called *nondeterministic*.

EXAMPLE 1. *The framework of transition systems generalizes dynamical systems. As an example, consider a nonlinear discrete-time dynamical system $x^+ = f(x, u)$ where $x \in \mathbb{R}^n$ is the state and $u \in \mathbb{R}^m$ is the input, with initial state x_0. This system can be represented by a deterministic transition system T with $Q = \mathbb{R}^n$, $\Sigma = \mathbb{R}^m$, $\rightarrow = \{(x, u, x^+) | x \in Q, u \in \Sigma, x^+ = f(x, u) \in Q\}$, and $Q^0 = \{x_0\}$. If the system is subject to disturbance: $x^+ = f(x, u, w)$ where $w \in \mathcal{W} \subseteq \mathbb{R}^p$ is the disturbance, then T becomes nondeterministic where $\rightarrow = \{(x, u, x^+) | x \in Q, u \in \Sigma, w \in \mathcal{W}, x^+ = f(x, u, w) \in Q\}$.*

A simulation relation between two transition systems $T_1 = (Q_1, \Sigma_1, \rightarrow_1, Q_1^0)$ and $T_2 = (Q_2, \Sigma_2, \rightarrow_2, Q_2^0)$ is a stronger notion of system refinement, which allows one system to "track" the other system while maintaining a certain relation between their states. A simulation relation is defined in definition 2. Note that in [8], it is required that T_1 and T_2 have the same label sets (and the same observation sets), however we modify the definition to relax this requirement, i.e., Σ_1 and Σ_2 can be different.

DEFINITION 2 (SIMULATION). *A relation $\mathcal{R} \subseteq Q_1 \times Q_2$ is called a simulation relation of T_1 by T_2 if and only if for all $(q_1, q_2) \in \mathcal{R}$ and for all transitions $q_1 \xrightarrow{\sigma_1}_1 q_1^+$, there exists a transition $q_2 \xrightarrow{\sigma_2}_2 q_2^+$ such that $(q_1^+, q_2^+) \in \mathcal{R}$.*

T_2 is said to simulate T_1, denoted $T_1 \preceq T_2$, if there exists a simulation relation $\mathcal{R}$ of T_1 by T_2 such that for all $q_1 \in Q_1^0$, there exists $q_2 \in Q_2^0$ such that $(q_1, q_2) \in \mathcal{R}$. Intuitively, if $T_1 \preceq T_2$ then every state trajectory of T_1 can be "tracked" by T_2 with respect to the state relation $\mathcal{R}$. Simulation relations and their variants are a powerful tool for safety verification and hierarchical controller design [9, 10, 11, 8].

5.2 Input-Constrained Simulation Relations

Conventionally, simulation relations as defined in section 5.1 do not explicitly take into account disturbances. Moreover, there is no relation between the inputs (or labels) to the two systems as required for the total demand bound in our problem. Therefore, this section extends the concept of simulation relations to account for these ingredients.

REMARK 2. *For nondeterministic systems, the simulation relation as defined in definition 2 is not robust to the nondeterminism. In particular, it is implicitly assumed that the transition from any q_1 under any σ_1, although nondeterministic, is detectable so that an appropriate transition $q_2 \xrightarrow{\sigma_2} q_2^+$ can be selected. T_2 is not nondeterministic anymore because here $q_2 \xrightarrow{\sigma_2} q_2^+$ can be chosen to satisfy $(q_1^+, q_2^+) \in \mathcal{R}$. Furthermore, it assumes no correlation between the nondeterminism of the two systems. In reality, their nondeterminism may be correlated, for example if they are subject to the same disturbances in different ways. Our extended definition of simulation relations provides options for robustness to disturbances and nondeterminism correlation.*

REMARK 3. *A relation between the labels of T_1 and T_2 generalizes the simulation relation definition in [8], where σ_1 and σ_2 are related by $\sigma_1 = \sigma_2$ (provided that $\Sigma_1 = \Sigma_2$).*

In this section, we generalize the label σ of a transition system T as a pair $\sigma = (\upsilon, \delta)$ of a control label $\upsilon \in \mathcal{U}$ and a disturbance label $\delta \in \mathcal{D}$. Here, $\mathcal{U}$ and $\mathcal{D}$ are respectively the (possibly infinite) sets of control labels and disturbance labels. The definition of the transition relation is modified accordingly as $\rightarrow \subseteq Q \times \mathcal{U} \times \mathcal{D} \times Q$, and a transition $(q, \upsilon, \delta, q^+) \in \rightarrow$ is denoted $q \xrightarrow{(\upsilon,\delta)} q^+$. Given two transition systems T_1 and T_2, we impose a constraint between their control labels υ_1 and υ_2 as a relation $\mathcal{R}_u \subseteq \mathcal{U}_1 \times \mathcal{U}_2$, and a constraint between their disturbance labels $\mathcal{R}_d \subseteq \mathcal{D}_1 \times \mathcal{D}_2$. Note that the constraint $\mathcal{R}_u$ is desirable but not required for the execution of the transition systems, i.e., an admissible execution of the transition systems may violate the constraint. The relation $\mathcal{R}_d$ takes effect whenever T_1 and T_2 are executed simultaneously in the same environment. For system T_i, the set of all admissible successor states from a state q_i under label $\sigma_i = (\upsilon_i, \delta_i)$ is denoted by $\mathrm{succ}_i(q_i, \upsilon_i, \delta_i) = \{q_i^+ \mid (q_i, \upsilon_i, \delta_i, q_i^+) \in \rightarrow_i\}$.

DEFINITION 3 (INPUT-CONSTRAINED SIMULATION). *A relation $\mathcal{R} \subseteq Q_1^S \times Q_2^S$ is an input-constrained simulation relation of T_1 by T_2 if and only if for all $(q_1, q_2) \in \mathcal{R}$ and all $(\upsilon_1, \delta_1) \in \mathcal{U}_1 \times \mathcal{D}_1$ such that $\mathrm{succ}_1(q_1, \upsilon_1, \delta_1) \neq \emptyset$, there exists $\upsilon_2 \in \mathcal{U}_2$ such that*

1. *$(\upsilon_1, \upsilon_2) \in \mathcal{R}_u$; and*
2. *for all $\delta_2 \in \mathcal{D}_2$ such that $(\delta_1, \delta_2) \in \mathcal{R}_d$, $\mathrm{succ}_2(q_2, \upsilon_2, \delta_2) \neq \emptyset$ and $(q_1^+, q_2^+) \in \mathcal{R}$ $\forall (q_1^+, q_2^+) \in \mathrm{succ}_1(q_1, \upsilon_1, \delta_1) \times \mathrm{succ}_2(q_2, \upsilon_2, \delta_2)$.*

The existence of an input-constrained simulation relation $\mathcal{R}$ of T_1 by T_2 allows T_2 to track any state sequence of T_1 with respect to $\mathcal{R}$, as long as their initial states are in $\mathcal{R}$. We define a trajectory of T_i as a (potentially infinite) sequence of admissible states, labels, and transitions: $q_i^0 \xrightarrow{\upsilon_i^0, \delta_i^0}_i q_i^1 \xrightarrow{\upsilon_i^1, \delta_i^1}_i q_i^2 \cdots$. The next lemma follows directly from the definition 3. It essentially says that if an input-constrained simulation relation $\mathcal{R}$ of T_1 by T_2 exists then T_2 can be controlled to track any admissible trajectory of T_1, with respect to $\mathcal{R}$, while keeping the input constraint $\mathcal{R}_u$.

LEMMA 1. *Suppose $\mathcal{R}$ is an input-constrained simulation relation of T_1 by T_2, whose initial states satisfy $(q_1^0, q_2^0) \in \mathcal{R}$. Let $\kappa_2 : Q_1 \times \mathcal{U}_1 \times \mathcal{D}_1 \times Q_2 \rightarrow \mathcal{U}_2$ be any feedback law such that for all admissible quadruples $(q_1, \upsilon_1, \delta_1, q_2) \in Q_1 \times \mathcal{U}_1 \times \mathcal{D}_1 \times Q_2$, $\upsilon_2 = \kappa_2(q_1, \upsilon_1, \delta_1, q_2)$ satisfies the conditions in definition 3. Such a feedback law always exists. Then for any trajectory of T_1, any corresponding trajectory of T_2 with $\upsilon_2^k = \kappa_2(q_1^k, \upsilon_1^k, \delta_1^k, q_2^k)$ satisfies $(q_1^k, q_2^k) \in \mathcal{R}$ and $(\upsilon_1^k, \upsilon_2^k) \in \mathcal{R}_u$ for all k.*

6. MODEL ABSTRACTION FOR ENERGY SYSTEM SCHEDULING

Going back to the energy system scheduling problem, we apply the framework of input-constrained simulation relations to abstract the high-dimensional model Σ_a by a low-dimensional model Σ_s. We then derive a feedback control law that allows Σ_a to track the state of Σ_s while maintaining their simulation relation, all their state and input constraints, as well as the input upper-bounds set by Σ_s. In particular, we consider the scalar discrete-time system Σ_s:

$$(\Sigma_s) \quad \begin{aligned} &s_{k+1} = \alpha s_k + v_k + \beta^T w_k \\ &s_k \in \mathcal{S} \subseteq \mathbb{R}, v_k \in \mathcal{V} \subseteq \mathbb{R}, (s_k, v_k) \in \Omega \subseteq \mathcal{S} \times \mathcal{V} \end{aligned} \tag{5}$$

The intuition is that s_k represents an aggregated state of all the subsystems at time step k and v_k specifies an upper-bound of the total energy demand. For example, consider an energy system consisting of n rooms and a heater in each room, where the heaters have fixed heating powers and can only be switched on and off. At time k, s_k can represent the total enthalpy of all rooms, while v_k bounds the total heating energy of all heaters during that interval. Because the system is subject to heat loss and disturbances, the dynamics of s have the form of (5).

The abstraction of Σ_a, i.e., the input-constrained simulation relation $\mathcal{R}$, depends on how the control input v_k is computed and how the disturbances are represented in Σ_s. In practice, forecasts of the disturbances, denoted $\tilde{w}_k$, with known bounded accuracies are often available and used in computing the control inputs for both Σ_s and Σ_a. The forecast accuracies are represented by a vector $\zeta \geq 0$ in $\mathbb{R}^p$ such that the error between w_k and $\tilde{w}_k$ are always bounded element-wise as $-\zeta \leqslant w_k - \tilde{w}_k \leqslant \zeta$ for all k.

We consider two control approaches in this paper:

1. **Feedforward approach:** The control sequence v_0, $v_1, \ldots, v_{N-1}$ of Σ_s are computed using the disturbance forecasts $\tilde{w}_0, \tilde{w}_1, \ldots, \tilde{w}_{N-1}$ as nominal disturbances for the horizon. These control inputs are not adjusted until the end of the horizon; in other words, the control law for Σ_s is feedforward. Consequently, the trajectory of Σ_s is nominal without taking into account actual

disturbances. The control input η_k for Σ_a is computed based on the actual state $\bar{x}_k$ of Σ_a and the nominal state, control, and disturbances of Σ_s, hence it is a feedback law.

2. **Feedback approach:** At each time step k, v_k is decided using the actual state s_k, which is computed from the observed actual state $\bar{x}_k$. The state trajectory of Σ_s is therefore actual, not nominal. The control input η_k is calculated from the actual states s_k and $\bar{x}_k$, and v_k and $\tilde{w}_k$. Hence, both control laws for Σ_a and Σ_s are feedback.

We derive the abstraction Σ_s of Σ_a for each approach in three steps:

1. Σ_s and Σ_a are formulated as transition systems T_1 and T_2 respectively;
2. Assuming a certain form of $\mathcal{R}$, we derive conditions for $\mathcal{R}$ to be an input-constrained simulation relation;
3. Based on the conditions, we develop algorithms to compute the parameters of Σ_s, its joint constraint Ω, and the simulation relation $\mathcal{R}$.

6.1 Feedforward Approach

6.1.1 Transition Systems

In the feedforward approach, the evolution of Σ_s is nominal, where the (nominal) disturbances are known and deterministic. We formulate T_1 as follows:

- State $q_1 \equiv s$ and state set $Q_1 = \mathcal{S}$.
- Control label $\upsilon_1 \equiv v$ with $\mathcal{U}_1 = \mathcal{V}$.
- Disturbance label $\delta_1 \equiv \tilde{w}$ is the disturbance forecast with $\mathcal{D}_1 = \mathcal{W}$ being the set of possible disturbances.
- Transition set $\rightarrow_1 = \{(s, v, \tilde{w}, s^+) \,|\, (s, v) \in \Omega, \tilde{w} \in \mathcal{W}, s^+ = \alpha s + v + \beta^T \tilde{w} \in \mathcal{S}\}$.

The transition system T_2 of Σ_a is defined as:

- State is the averaged state $q_2 \equiv \bar{x}$ with $Q_2 = \mathcal{X}$.
- Control label $\upsilon_2 \equiv \eta$ with $\mathcal{U}_2 = [0, 1]^m$.
- Disturbance label: Σ_a is influenced by the actual disturbances w_k, therefore $\delta_2 \equiv w$ with $\mathcal{D}_2 = \mathcal{W}$.
- Transition set: recall that at every time step $k+1$, $\bar{x}_{k+1}$ is reset to $x((k+1)T)$, which is within a bounded error e_{k+1} from $A_T\bar{x}_k + B_T\eta_k + E_T w_k$ (cf. theorem 1). This reset can be modeled as a non-deterministic but bounded perturbation of the state. Therefore we define $\rightarrow_2 = \{(\bar{x}, \eta, w, \bar{x}^+) \,|\, \bar{x} \in \mathcal{X}, \eta \in \mathcal{U}_2, w \in \mathcal{W}, \bar{x}^+ = A_T\bar{x} + B_T\eta + E_T w + e \in \mathcal{X}, \underline{\varepsilon} \leqslant V(e + \tilde{B}\eta) \leqslant \bar{\varepsilon}\}$.

Because v_k specifies an upper-bound of the total energy demand $T\rho^T\eta_k$, we define the joint input constraint $\mathcal{R}_u = \{(v, \eta) \in \mathcal{U}_1 \times \mathcal{U}_2 \,|\, T\rho^T\eta \leq v\}$. Finally, the forecast accuracy between nominal $\tilde{w}$ of Σ_s and actual w of Σ_a is modeled by the disturbance relation $\mathcal{R}_d = \{(\tilde{w}, w) \in \mathcal{D}_1 \times \mathcal{D}_2 \,|\, -\zeta \leqslant w - \tilde{w} \leqslant \zeta\}$.

6.1.2 Conditions for $\mathcal{R}$

To derive the model abstraction of Σ_a, the input-constrained simulation relation $\mathcal{R}$ must be computed. Specifically, we need to be able to certify if any given relation $\mathcal{R}$ of s and $\bar{x}$ is an input-constrained simulation relation of T_1 by T_2. The following theorem provides such a necessary and sufficient condition for $\mathcal{R}$. Here, the product between a matrix $M \in \mathbb{R}^{m\times n}$ and a set $\mathcal{P} \subseteq \mathbb{R}^n$ is defined as the image of $\mathcal{P}$ under the linear map M, i.e., $M\mathcal{P} = \{y \in \mathbb{R}^m \,|\, y = Mx, x \in \mathcal{P}\}$. The identity matrix is denoted by $\mathbb{I}$, whose dimensions are often omitted when they are obvious from the context. The symbol $\mathbf{0}$ denotes a zero vector or matrix.

THEOREM 2. *A set $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{X}$ is an input-constrained simulation relation of T_1 by T_2 for the feedforward approach if and only if*

$$M_l\mathcal{P}_l \subseteq M_r\mathcal{P}_r \tag{6}$$

where

$$M_l = \begin{bmatrix} 0 & \mathbf{0} & 1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \alpha & \mathbf{0} & 1 & \mathbf{0} & \beta^T & \mathbf{0} \\ 0 & A_T & 0 & E_T & E_T & \mathbb{I} \end{bmatrix}$$

$$M_r = \begin{bmatrix} 0 & \mathbf{0} & 1 & & \mathbf{0} & \\ 1 & \mathbf{0} & 0 & & \mathbf{0} & \\ 0 & \mathbb{I} & 0 & \tilde{B} - B_T & & \end{bmatrix}$$

and

$$\begin{aligned} \mathcal{P}_l = &\big\{(s, \bar{x}, v, \delta, \tilde{w}, \varepsilon) \,\big|\, (s, \bar{x}) \in \mathcal{R},\ (s, v) \in \Omega, -\zeta \leqslant \delta \leqslant \zeta, \\ &\qquad \tilde{w} \in \mathcal{W}, \tilde{w} + \delta \in \mathcal{W}, \alpha s + v + \beta^T\tilde{w} \in \mathcal{S}, \underline{\varepsilon} \leqslant \varepsilon \leqslant \bar{\varepsilon}\big\} \\ \mathcal{P}_r = &\big\{(s^+, \bar{x}^+, \bar{v}, \eta) \,\big|\, \bar{x}^+ \in \mathcal{X}, (s^+, \bar{x}^+) \in \mathcal{R}, \eta \in [0, 1]^m, \\ &\qquad T\rho^T\eta \leqslant \bar{v}\big\} \end{aligned}$$

The proof can be found in appendix A.2.

Although theorem 2 is useful for verifying whether a given $\mathcal{R}$ is an input-constrained simulation relation, it does not allow us to directly compute $\mathcal{R}$ since $\mathcal{R}$ appears on both sides of (6). Furthermore, computing the image of a set under a linear map and checking for a set inclusion are generally difficult. Therefore, we will impose a certain structure on the relation $\mathcal{R}$ that results in polyhedral sets $\mathcal{P}_l$ and $\mathcal{P}_r$. We will also derive a simpler condition for $\mathcal{R}$, which can be verified by checking the feasibility of a linear program.

As mentioned in the overview in section 3 and again at the beginning of this section, s represents an aggregated state of all the subsystems, for example the total enthalpy of an energy system. Thus, we can choose s to be approximately a linear combination of all the (energy) states of the subsystems. Let $s \approx c^T\bar{x}$, where c is a constant vector in $\mathbb{R}^n$. The set $\mathcal{R}$ is chosen to represent this relationship between s and $\bar{x}$ as

$$\mathcal{R} = \{(s, \bar{x}) \in \mathcal{S} \times \mathcal{X} \,|\, |c^T\bar{x} - s| \leqslant \gamma\}, \tag{7}$$

where $\gamma \geqslant 0$ is a design parameter. Intuitively, in an energy system, the set of $\bar{x} \in \mathcal{X}$ such that $(s, \bar{x}) \in \mathcal{R}$, for any $s \in \mathcal{S}$, is the set of all states that have roughly the same aggregated energy level s (up to an error of γ).

We have assumed in section 2 that $\mathcal{X}$ and $\mathcal{W}$ are polyhedra. Suppose also that Ω is a polyhedron. Let their hyperplane representations be

$$\mathcal{X} = \{x \in \mathbb{R}^n \,|\, H_x x \leqslant k_x\}, \qquad \mathcal{W} = \{w \in \mathbb{R}^p \,|\, H_w x \leqslant k_w\}$$
$$\Omega = \{(s, v) \in \mathcal{S} \times \mathcal{V} \,|\, H^s_\Omega s + H^v_\Omega v \leqslant k_\Omega\}$$

where H_x, k_x, H_w, k_w, H^s_Ω, H^v_Ω, k_Ω are matrices and vectors of appropriate dimensions. Because s and v are scalars, we can bound them as $\underline{s} \leqslant s \leqslant \bar{s}$ and $\underline{v} \leqslant v \leqslant \bar{v}$, where $\underline{s}$, $\bar{s}$, $\underline{v}$, $\bar{v}$ are respectively the lower- and upper-bounds of s and v. Note that Ω is a subset of $\mathcal{S} \times \mathcal{V} = [\underline{s}, \bar{s}] \times [\underline{v}, \bar{v}]$. With these assumptions, $\mathcal{P}_l$ and $\mathcal{P}_r$ become polyhedral sets. We can now state a sufficient condition for $\mathcal{R}$ in the next theorem.

THEOREM 3. *A set $\mathcal{R}$ as defined in (7) is an input-constrained simulation relation of T_1 by T_2 for the feedforward approach if there exist a real matrix Z of non-negative elements, a real matrix Q, and a real vector q such that*

$$M_r Q = \mathbb{I} \tag{8a}$$
$$M_r q = \mathbf{0} \tag{8b}$$
$$Z H_l = H_r Q M_l \tag{8c}$$
$$Z k_l \leqslant k_r - H_r q \tag{8d}$$

in which

$$H_l = \begin{bmatrix} & H_x & & & & \\ -1 & c^T & & & & \\ 1 & -c^T & & & & \\ H^s_\Omega & & H^v_\Omega & & & \\ & & & H_w & & \\ & & & H_w & H_w & \\ & & & \mathbb{I} & & \\ & & & -\mathbb{I} & & \\ \alpha & & 1 & & \beta^T & \\ -\alpha & & -1 & & -\beta^T & \\ & & & & & V \\ & & & & & -V \end{bmatrix}, \; k_l = \begin{bmatrix} k_x \\ \gamma \\ \gamma \\ k_\Omega \\ k_w \\ k_w \\ \zeta \\ \zeta \\ \bar{s} \\ -\underline{s} \\ \bar{\varepsilon} \\ -\underline{\varepsilon} \end{bmatrix}$$

$$H_r = \begin{bmatrix} & H_x & & \\ -1 & c^T & & \\ 1 & -c^T & & \\ & & -1 & T\rho^T \\ & & & \mathbb{I} \\ & & & -\mathbb{I} \end{bmatrix}, \; k_r = \begin{bmatrix} k_x \\ \gamma \\ \gamma \\ 0 \\ \mathbf{1} \\ \mathbf{0} \end{bmatrix}.$$

In the above matrices and vectors, the blank spaces are zero blocks and $\mathbf{1}$ denotes vector of 1's. The theorem is proved in appendix A.3.

Although theorem 3 is only sufficient, it reduces the condition to checking the feasibility of the linear program (8), which can be solved rather efficiently by any LP solver such as Gurobi, MOSEK, CPLEX, CLP.

We note that the design parameters, which are (α, β, c, γ, H^s_Ω, H^v_Ω, k_Ω), are multiplied by the elements of Z in (8). Consequently, if we want to solve (8) to also find the design parameters, the problem becomes a bilinear program, which is intractable to solve for any significant size. We will discuss a way to compute these parameters in section 6.3, after we derive similar results for the feedback control approach in the next section.

6.2 Feedback Approach

6.2.1 Transition Systems

In the feedback approach, the evolution of Σ_s is not nominal as it is influenced by the actual disturbances, which are unknown but within known accuracies from their forecasts. To model this phenomenon, we will include the disturbance forecasts in the control labels of both T_1 and T_2, while their disturbance labels are the errors between the forecasts and the actual disturbances. Specifically, we define T_1 as follows:

- State $q_1 \equiv s$ and state set $Q_1 = \mathcal{S}$.
- Control label $\upsilon_1 \equiv (v, \tilde{w}_1)$ with $\mathcal{U}_1 = \mathcal{V} \times \mathcal{W}$.
- Disturbance label δ_1 is the disturbance forecast error $\delta_1 = w - \tilde{w}_1$; $\mathcal{D}_1 = \{\delta_1 \in \mathbb{R}^p \mid -\zeta \leqslant \delta_1 \leqslant \zeta\}$.
- Transition set $\rightarrow_1 = \{(s, v, \tilde{w}_1, \delta_1, s^+) \mid (s, v) \in \Omega, \tilde{w}_1 \in \mathcal{W}, \delta_1 \in \mathcal{D}_1, \tilde{w}_1 + \delta_1 \in \mathcal{W}, s^+ = \alpha s + v + \beta^T(\tilde{w}_1 + \delta_1) \in \mathcal{S}\}$.

The transition system T_2 of Σ_a is defined as:

- State $q_2 \equiv \bar{x}$; $Q_2 = \mathcal{X}$.
- Control label $\upsilon_2 \equiv (\eta, \tilde{w}_2)$ with $\mathcal{U}_2 = [0, 1]^m \times \mathcal{W}$.
- Disturbance label δ_2 is also the forecast error; $\mathcal{D}_2 = \{\delta_2 \in \mathbb{R}^p \mid -\zeta \leqslant \delta_2 \leqslant \zeta\}$.
- Transition set: we again model the state reset as a non-deterministic but bounded perturbation of $\bar{x}$. Define $\rightarrow_2 = \{(\bar{x}, \eta, \tilde{w}_2, \delta_2, \bar{x}^+) \mid \bar{x} \in \mathcal{X}, \eta \in \mathcal{U}_2, \tilde{w}_2 \in \mathcal{W}, \delta_2 \in \mathcal{D}_2, \tilde{w}_2 + \delta_2 \in \mathcal{W}, \bar{x}^+ = A_T \bar{x} + B_T \eta + E_T(\tilde{w}_2 + \delta_2) + e \in \mathcal{X}, \underline{\varepsilon} \leqslant V(e + \tilde{B}\eta) \leqslant \bar{\varepsilon}\}$.

The joint input constraint $\mathcal{R}_u$ needs to take into account that the disturbance forecasts are the same for T_1 and T_2, hence $\mathcal{R}_u = \{((v, \tilde{w}_1), (\eta, \tilde{w}_2)) \in \mathcal{U}_1 \times \mathcal{U}_2 \mid T\rho^T \eta \leq v, \tilde{w}_1 = \tilde{w}_2\}$. Finally, because T_1 and T_2 are subject to the same actual disturbances, we define the disturbance relation $\mathcal{R}_d = \{(\delta_1, \delta_2) \in \mathcal{D}_1 \times \mathcal{D}_2 \mid \delta_1 = \delta_2\}$.

6.2.2 Conditions for $\mathcal{R}$

The only differences between the feedback and feedforward approaches are that in the feedback approach: (a) the control input v for T_1 must robustly drive s to $\mathcal{S}$ regardless of the forecast error δ; (b) s is updated with the actual disturbance $\tilde{w} + \delta$. Therefore, the necessary and sufficient condition in theorem 2 still holds for the feedback approach with modified matrix M_l and set $\mathcal{P}_l$:

$$M_l = \begin{bmatrix} 0 & \mathbf{0} & 1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \alpha & \mathbf{0} & 1 & \beta^T & \beta^T & \mathbf{0} \\ 0 & A_T & 0 & E_T & E_T & \mathbb{I} \end{bmatrix}$$

$$\begin{aligned} \mathcal{P}_l = \big\{ & (s, \bar{x}, v, \delta, \tilde{w}, \varepsilon) \,\big|\, (s, \bar{x}) \in \mathcal{R}, \; (s, v) \in \Omega, -\zeta \leqslant \delta \leqslant \zeta, \\ & \tilde{w} \in \mathcal{W}, \tilde{w} + \delta \in \mathcal{W}, \alpha s + v + \beta^T \tilde{w} \in \mathcal{S} \ominus \beta^T \mathcal{D}_1, \\ & \underline{\varepsilon} \leqslant \varepsilon \leqslant \bar{\varepsilon} \big\} \end{aligned}$$

where $\ominus$ denotes the Pontryagin difference between two sets. Similarly, theorem 3 is still valid when k_l is modified accordingly:

$$k_l = \begin{bmatrix} k_x \\ \gamma \\ \gamma \\ k_\Omega \\ k_w \\ k_w \\ \zeta \\ \zeta \\ \bar{s} - \max_{\delta \in [-\zeta, \zeta]} \beta^T \delta \\ -\underline{s} + \min_{\delta \in [-\zeta, \zeta]} \beta^T \delta \\ \bar{\varepsilon} \\ -\underline{\varepsilon} \end{bmatrix}$$

6.3 Computation of Design Parameters

The design parameters of the model abstraction, which we aim to compute, are (α, β, c, γ, H^s_Ω, H^v_Ω, k_Ω). As mentioned above, they enter the sufficient condition (eq. (8)) in bilinear terms, hence it is intractable to compute them (optimally) using this condition. In this section, we present a method to design these parameters in a suboptimal but tractable way.

6.3.1 Computing α, β, c

We compute these parameters based on the intuition that s is the aggregated state of all the subsystems: $s \approx c^T \bar{x}$. From the discrete-time dynamics of $\bar{x}$ we have

$$s_{k+1} \approx c^T \bar{x}_{k+1} = c^T A_T \bar{x}_k + c^T B_T \eta_k + c^T E_T w_k$$

while at the same time

$$s_{k+1} = \alpha s_k + v_k + \beta^T w_k \approx \alpha c^T \bar{x}_k + T\rho^T \eta_k + \beta^T w_k.$$

Matching the two equations, we aim to find α and c so that $c^T B_T = T\rho^T$ while minimizing $\|c^T A_T - \alpha c^T\|$, where $\|\cdot\|$ denotes some vector norm. If B_T is full row rank, c can be uniquely computed by solving a linear equation, and the minimization becomes a simple unconstrained program. If B_T is not full row rank, c and α can be found by solving the above nonlinear optimization, which can be made unconstrained by rewriting c in terms of the basis vectors of the kernel of B_T^T. We then calculate $\beta = E_T^T c$.

6.3.2 Computing γ and Ω

Once α, β, and c are chosen, we can compute the parameter γ of the relation $\mathcal{R}$ and the constraint set Ω of (s, v). We first choose the ranges $[\underline{s}, \bar{s}]$ and $[\underline{v}, \bar{v}]$ of s and v that satisfy

$$\underline{s} \geqslant \min_{x \in \mathcal{X}} c^T x, \quad \bar{s} \leqslant \max_{x \in \mathcal{X}} c^T x$$

$$\underline{v} \geqslant \min_{\mathbf{0} \leqslant \eta \leqslant \mathbf{1}} T\rho^T \eta, \quad \bar{v} \leqslant \max_{\mathbf{0} \leqslant \eta \leqslant \mathbf{1}} T\rho^T \eta$$

based on the relationships between s and $\bar{x}$, v and η. Because X is bounded, $\underline{s}$ and $\bar{s}$ are finite; and so are $\underline{v}$ and $\bar{v}$.

Choosing γ and Ω is difficult because γ appears in both k_l and k_r, and Ω is a set. Obviously, it is unnecessary for γ to be larger than $\bar{s} - \underline{s}$ due to the relation $|c^T \bar{x} - s| \leqslant \gamma$. Therefore we can select any value between 0 and $\bar{s} - \underline{s}$ for γ, with the apparent intuition that as γ gets smaller, the constraint between s and $\bar{x}$ becomes stricter, keeping $(s, \bar{x}$ in $\mathcal{R}$ is more difficult, hence $\mathcal{R}$ is less likely an input-constrained simulation relation. For Ω, which is a subset of $[\underline{s}, \bar{s}] \times [\underline{v}, \bar{v}]$, the following result allows us to construct Ω from smaller sets.

LEMMA 2. *Suppose that all the parameters are fixed except for Ω. Let Ω_1 and Ω_2 are two subsets of $[\underline{s}, \bar{s}] \times [\underline{v}, \bar{v}]$ and suppose that they satisfy the sufficient condition in theorem 3. Then their union $\Omega_1 \cup \Omega_2$ satisfies the necessary and sufficient condition in theorem 2.*

This lemma can be proved by noting that Ω_1 and Ω_2 satisfy theorem 2, and Ω appears only in the left-hand side of the set inclusion (eq. (6)). Using this result, we can check the sufficient condition in theorem 3 for small sets, then construct the final Ω set from their union. This algorithm is outlined below.

1. We first grid the finite intervals of s and v as $\underline{s} = s^0 < s^1 < \cdots < s^{l_s} = \bar{s}$ and $\underline{v} = v^0 < v^1 < \cdots < v^{l_v} = \bar{v}$. This effectively grids the box $[\underline{s}, \bar{s}] \times [\underline{v}, \bar{v}]$ into $l_s l_v$ small cells.

2. For each cell $\Omega_{i,j} = [s^i, s^{i+1}] \times [v^j, v^{j+1}]$, for $0 \leqslant i \leqslant l_s - 1$ and $0 \leqslant j \leqslant l_v - 1$, check if it satisfies theorem 3.

3. Construct Ω as the union of all cells $\Omega_{i,j}$ that satisfy theorem 3. If the projection of Ω onto s covers the entire range $[\underline{s}, \bar{s}]$ then take Ω. This is to ensure that we can always find an admissible control input v for any valid value of s. We can simplify Ω by a convex polyhedron contained in the union, from which H_Ω^s, H_Ω^v and k_Ω are extracted.

REMARK 4. *The constructed set Ω may not satisfy theorem 3 because the condition is only sufficient. However, Ω is valid because it satisfies the stronger necessary and sufficient condition in theorem 2, due to lemma 2.*

7. SCALABLE SCHEDULING: DESIGN AND ALGORITHMS

This section summarizes the results developed in the previous sections and discusses the design process and the run-time implementation of our approach.

7.1 Design Process of Scalable Scheduling

Given the scheduling problem (3):

1. We first discretize the dynamical model with sampling time T to obtain the matrices A_T, B_T, and E_T.

2. Compute $\underline{\varepsilon}$, $\bar{\varepsilon}$, and $\tilde{B}$ for the error between x and $\bar{x}$, using theorem 1.

3. Compute parameters α, β, c of the scalar model Σ_s (section 6.3.1).

4. Compute parameters γ and Ω of the relation $\mathcal{R}$ (section 6.3.2). The construction of Ω requires checking the feasibility of the LP (8) for many times. For example, if we grid the ranges of s and v by 100 segments each, we must solve the feasibility problem 10,000 times.

7.2 Run-time Implementation

8. CASE STUDY: ROOM HEATING SYSTEM

9. CONCLUSIONS

10. REFERENCES

[1] Andrew Marr. *Tea Time in Britain, Britain from Above*, 2008 (accessed March 30, 2015).

[2] TRFund Energy Study. Understanding PECO's general service tariff.

[3] A.A. Pesaran, A. Vlahinos, and S.D. Burch. Thermal performance of ev and hev battery modules and packs. *Proceedings of the 14th International Electric Vehicle Symposium*, 1997.

[4] A. Khaligh and Z. Li. Battery, ultracapacitor, fuel cell, and hybrid energy storage systems for electric, hybrid electric, fuel cell and plug-in hybrid electric vehicles: State of the art. *IEEE Tran. on Vehicular Tech.*, 2008.

[5] Jane W. S. Liu. *Real-time Systems*. Prentice Hall, 2000.

[6] Tullio Facchinetti, Enrico Bini, and Marko Bertogna. Reducing the peak power through real-time scheduling techniques in cyber-physical energy systems. In *Proceedings of the First International Workshop on Energy Aware Design and Analysis of Cyber Physical Systems (WEA-CPS)*, 2010.

[7] R. Alur, T.A. Henzinger, G. Lafferriere, and G.J. Pappas. Discrete abstractions of hybrid systems. *Proc. IEEE*, 88(7):971–984, 2000.

[8] A. Girard and G. J. Pappas. Approximation metrics for discrete and continuous systems. *IEEE Trans. Autom. Control*, 52(5):782–798, 2007.

[9] Edmund M Clarke, Orna Grumberg, and Doron Peled. *Model checking*. MIT press, 1999.

[10] George J. Pappas, Gerardo Lafferriere, and Shankar Sastry. Hierarchically consistent control systems. *IEEE Trans. Autom. Control*, 45(6):1144–1160, 2000.

[11] Antoine Girard and George J. Pappas. Verification using simulation. In *Hybrid Systems: Computation and Control*, pages 272–286. Springer, 2006.

APPENDIX

A. PROOFS

A.1 Proof of theorem 1

Because A is diagonal, each element $e_i((k+1)T)$ can be calculated independently

$$e_i((k+1)T) = \int_{kT}^{(k+1)T} \mathrm{e}^{a_i((k+1)T-s)} b_i(u(s)-\eta_k)\mathrm{d}s.$$

If $a_i = 0$ then

$$e_i((k+1)T) = \int_{kT}^{(k+1)T} b_i(u(s)-\eta_k)\mathrm{d}s = b_i\left(\int_{kT}^{(k+1)T} u(s)\mathrm{d}s - T\eta_k\right) = 0$$

where the last equality comes from the definition of η_k. Consider the case $a_i \neq 0$. Observe that $\int_{kT}^{(k+1)T} \lambda_i b_i(u(s) - \eta_k)\mathrm{d}s = 0$ for any constant λ_i. Consequently

$$e_i((k+1)T) = \int_{kT}^{(k+1)T} (\mathrm{e}^{a_i((k+1)T-s)} - \lambda_i) b_i(u(s)-\eta_k)\mathrm{d}s$$

where $\lambda_i = 1$ if $a_i > 0$ and $\lambda_i = \mathrm{e}^{a_i T}$ if $a_i < 0$. It is straightforward to show that $\mathrm{e}^{a_i((k+1)T-s)} - \lambda_i \geqslant 0$ for all $s \in [kT, (k+1)T]$. Because $u(s) \in \{0,1\}^m$ for all s, $\underline{\varepsilon}_i \leqslant b_i u(s) \leqslant \bar{\varepsilon}_i$ with $\underline{\varepsilon}_i$ and $\bar{\varepsilon}_i$ defined as above. Therefore $(\mathrm{e}^{a_i((k+1)T-s)} - \lambda_i)(\underline{\varepsilon}_i - b_i\eta_k) \leqslant (\mathrm{e}^{a_i((k+1)T-s)} - \lambda_i) b_i(u(s) - \eta_k) \leqslant (\mathrm{e}^{a_i((k+1)T-s)} - \lambda_i)(\bar{\varepsilon}_i - b_i\eta_k)$. It follows that

$$\xi_i(\underline{\varepsilon}_i - b_i\eta_k) \leqslant \int_{kT}^{(k+1)T} (\mathrm{e}^{a_i((k+1)T-s)} - \lambda_i) b_i(u(s)-\eta_k)\mathrm{d}s \leqslant \xi_i(\bar{\varepsilon}_i - b_i\eta_k)$$

where $\xi_i = \int_{kT}^{(k+1)T} (\mathrm{e}^{a_i((k+1)T-s)} - \lambda_i)\mathrm{d}s = \frac{1}{a_i}(\mathrm{e}^{a_i T} - 1) - \lambda_i T$. The theorem is proved.

A.2 Proof of theorem 2

A.3 Proof of theorem 3