From Control to Scheduling: An Elastic Execution Model

Madhur Behl, Willy Bernal, Truong Nghiem, Miroslav Pajic and Rahul Mangharam
Dept. of Electrical and Systems Engineering
University of Pennsylvania
{mbehl, willyg, nghiem, pajic, rahulm}@seas.upenn.edu

***Abstract*—We present an elastic execution model for scheduling control systems while maintaining an acceptable level of service. Scheduling allows for coordination, composition and optimization across multiple interacting and non-interacting control systems. Unlike traditional real-time scheduling theory, where the execution time, and hence the schedule, are a function of the system variables only, elastic execution time models found in Cyber-Physical Systems are a function of the physical variables and the dynamics of the system that is being controlled. A task-set is constructed by extracting the temporal parameters of a system from its dynamics. We then provide the conditions for feasibility, optimality and admissibility for a set of tasks. One application will be to implement an 'energy-router', which coordinates the electrical demand from multiple control systems to minimize the peak-to-average ratio of energy consumption in buildings. Our work is a step towards developing scheduling theory for cyber-physical systems.**

I. Introduction

Cyber Physical Systems (CPS) are the next generation of real-time and embedded systems where control, communication and computation are tightly coupled with the underlying physical substrates. CPS operate within the constraints of their environment and must maintain safety and efficacy of the physical processes. In traditional, computing system-centric real-time systems, a set of tasks are scheduled to use the available resources efficiently while ensuring critical tasks always meet their deadlines. We now ask the question: Can traditional scheduling algorithms and task models be extended to Cyber-Physical Systems? If so, then how well does the system perform within the context of its environment? In CPS, in addition to the computing and communication constraints, the timeliness requirements of the tasks are a function of the physical process dynamics and the operating environment.

There is a fundamental difference between how traditional real-time theory treats timing constraints of a system and the timings restrictions that exist in a cyber physical system. In traditional real-time scheduling theory, the concept of periods, release times, execution time and deadlines of tasks is well specified as system-centric functional and timing requirements. Often some of these parameters are assumed to be fixed while carrying out a feasibility analysis of a given set of tasks. For instance, many scheduling algorithms for scheduling periodic and aperiodic tasks assume an upper bound on the execution time of a task. This can be estimated using a worst-case execution time analysis, among other techniques. Thus, given an upper bound on the execution time of a task, one can derive the demand-bound function for the task set, prove the schedulability of the task set and provide an admission control policy such that all tasks meet their deadlines.

In a cyber physical system, the execution time is a function of the system dynamics and the environment. Not only do the control and scheduling decisions influence the physical parameters of the system but the physical parameters can also influence the scheduling parameters. The fact that scheduling parameters are no longer determined by system specifications alone, suggests that we cannot use traditional real time scheduling algorithms to schedule such systems. Our goal is to explore alternate task execution models to better address generic CPS scheduling problems. To illustrate the models, we later focus on the problem of coordinating multiple energy sinks in a building to provide for a more energy-efficient operation with a minimal peak-to-average ratio. In order to achieve this goal we define task models and the set of conditions for which this class of cyber physical systems is schedulable.

The elastic execution should not be confused with elastic scheduling, in which task utilization can be changed by varying the period of the tasks. In our model, the execution time is a function of the response time and initial conditions of the control task, which further might be a function of more physical parameters such as human occupancy, weather and the price of energy.

The goal is to develop an 'energy-router' for buildings to provide a switching schedule for the HVAC (Heating, Ventilation and Air Conditioning) and refrigeration systems in a building based on their current states. The energy-router models the building as a hierarchy of control loops and implements an elastic execution scheduling model that optimizes user comfort and reduces the Peak to Average Ratio (PAR) of energy consumed. The US Department of Energy (DoE) estimates that 73% of the electricity usage is consumed by commercial buildings and residential housing [1]. HVAC systems account for 50% of the total energy budget in buildings [2]. Energy pricing is based on peak demand as there is no way to efficiently store and buffer energy. Thus spikes in energy demand can lead to brownouts and unstability across the power grid.

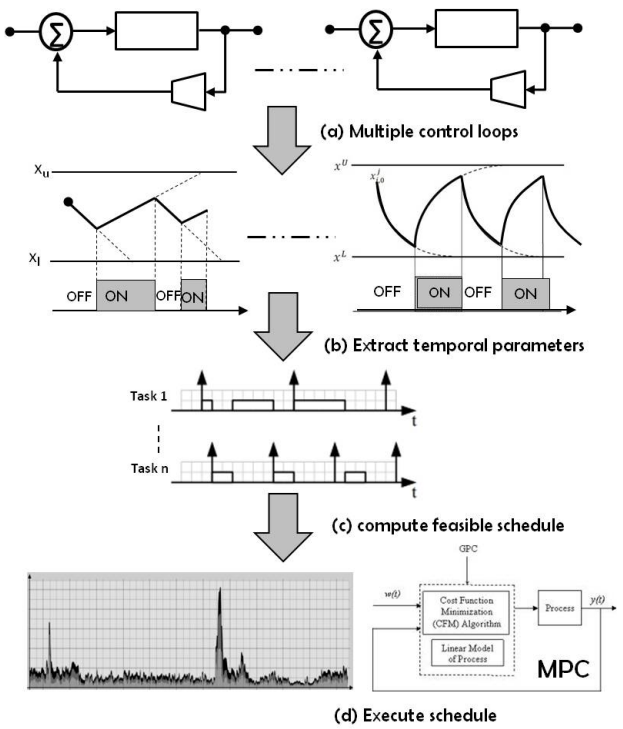

Figure 1. Different stages involved from going towards scheduling from control systems

For example, consider a residential building, whose occupants are watching live coverage of the Superbowl from their homes. During commercial breaks, a large number of viewers will tend to open their refrigerators to grab a beverage. On doing so, the compressor of the refrigerator *kicks* in and causes a small spike in the power consumption of that house. But if a large number of residents do the same thing, these small spikes accumulate and cause a considerably large voltage spike for the particular building. Considering that hundreds of thousand of viewers are watching the game in a city and millions across the nation,this often leads to a huge spike in the voltage level for a short duration. In this duration, the power companies need to meet the massive demand in order to prevent an brownout. The correlation between human behavior and the environment can easily drive an uncoordinated system towards instability.

In general, given a set of control loops that control a system or a process, our goal is to find a schedule for the controllers such that we always operate within the acceptable region of performance while minimizing resources consumed by the system. This idea is depicted in Fig. 1, where the first step is to form a state-time profile for the system being controlled, as shown in Fig. 1(a). The temporal parameters are extracted from the profiles to form a set of tasks (Fig. 1(b)). For a given set of tasks, we check conditions for schedulability and if the task set is schedulable then we form a feasible schedule using some scheduling policy (Fig. 1(c)). Lastly, as shown in Fig. 1(d), we execute our scheduling algorithm using a finite horizon dynamic programming or predictive algorithms while minimizing a resource/performance parameter.

The rest of the paper is structured in the following manner. In Section 2, we formulate how temporal parameters can be extracted from the systems dynamics. We provide the analysis for the linear case in Section 3. Section 4 describes the application of the elastic execution model to optimize energy consumption for a building. Section 5 concludes this initial work with a roadmap of our future work.

II. Task Model

In order to create a schedule for the dynamic systems that are being controlled, we first abstract the timing requirements of multiple control systems as a set of tasks. The temporal parameters of a task can be extracted from the state-time profile of the system being controlled. In this paper we only consider systems which have linear or exponential time profiles. Each task has a state that grows or decays linearly or exponentially based on the system dynamics. For example, in the case of a room's temperature, the profile is most likely to be exponential. Each system can be viewed as one task. Every task has an upper and lower threshold value for the state. These thresholds correspond to the deadband of states between which the system should operate at any time. A task T_i is said to be feasible *iff* the state of the task at any time is in between the thresholds. We want to find a switching schedule for a set of tasks such that all tasks remain feasible subject to the condition that at most one task will remain on at any time.

To define a feasible schedule, we need a notion of deadlines of the tasks. The deadline of the task $d_{i,l}$ is defined as the maximum time up to which the task can remain OFF after which it will fall below its lower threshold , and likewise if

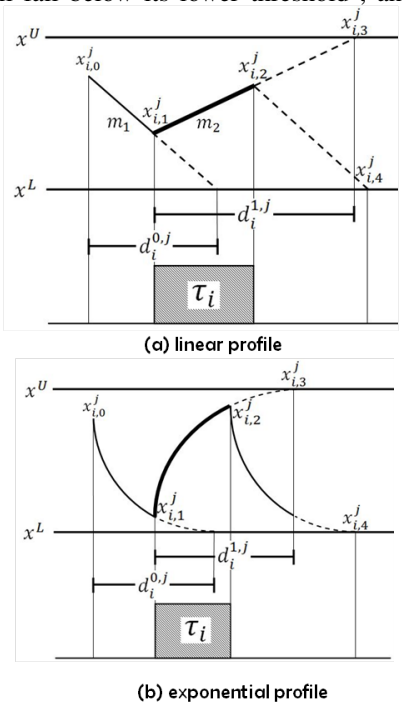

Figure 2. Extraction of temporal parameters from system dynamics

the task was already ON, then it has a deadline $d_{i,u}$ which corresponds to the maximum time it can remain ON, after which it will exceed the upper threshold and become non-feasible. The deadlines can be calculated with the knowledge of the growth and decay rates for the tasks. An example of how this can be done is shown in Figure 2. For part (a) we consider a system with a linear profile, with a decay rate of m_1 and a growth rate of m_2. When the task is at its initial state, it can have two deadlines based on if remains ON or OFF from this point onwards. These deadlines are depicted in the figure as $d_i^{0,j}$ and so on. We also define the execution time e_i of a task as the duration for which the tasks remains ON. As can be seen from the figure, the execution time is not fixed and is a function of when we switch on the task and is therefor a function of the response time and the initial conditions. Part (b) of the figure shows similar analysis for a system with exponential growth and decay rates. We observe that the slope of the system response, and hence the effective execution time, depends on the response time and initial conditions.

Once we have have extracted the temporal parameters of a system we can look at the constraints that these parameters must satisfy in order for the tasks to be schedulable. In this paper we provide the analysis for the simple case of systems with linear profiles like the one shown in figure 2.

III. Analysis for Linear Tasks

Consider a set of n linear tasks $\{T_i\}$. Each task T_i has a state variable $x_i \in \mathbb{R}$. At any time, each task can be in one of two modes: ON (or scheduled) or OFF. In ON mode, the dynamics of T_i is given by the differential equation $\frac{\mathrm{d}x_i}{\mathrm{d}t} = -a_i$ where $a_i > 0$ is the downward slope (or discharge rate) of T_i. In OFF mode, the dynamics of T_i is given by the differential equation $\frac{\mathrm{d}x_i}{\mathrm{d}t} = b_i$ where $b_i > 0$ is the upward slope (or charge rate) of task T_i. At any time, only one task can be scheduled (being charged) as we assume a finite power supply. The stability constraint requires that the state variable x_i of task T_i must stay within a range $[l_i, h_i]$, that is $l_i \leq x_i \leq h_i$ at all time. Time starts at $t = 0$ and the initial value of x_i is $x_i(0) = x_{0,i}$, where $l_i \leq x_{0,i} \leq h_i$.

A. Infeasibility result

At time $t \geq 0$, let $t_{\mathrm{on},i}$, $0 \leq t_{\mathrm{on},i} \leq t$, be the total time since the beginning that T_i is on. Then the total time since the beginning that T_i is off is $t_{\mathrm{off},i} = t - t_{\mathrm{on},i}$. The resource constraint requires that

$$\sum_{i=1}^{n} t_{\mathrm{on},i} \leq t. \tag{1}$$

The state value of task T_i at time t can be computed as

$$x_i(t) = x_{0,i} - a_i t_{\mathrm{off},i} + b_i t_{\mathrm{on},i} = x_{0,i} - a_i t + (a_i + b_i) t_{\mathrm{on},i}.$$

Define new variable $\hat{x}_i = \frac{x_i - x_{0,i}}{a_i + b_i}$ we have that

$$\hat{x}_i(t) = -\frac{a_i}{a_i + b_i} t + t_{\mathrm{on},i} = -d_i t + t_{\mathrm{on},i} \tag{2}$$

in which $d_i = \frac{a_i}{a_i + b_i} > 0$. The stability constraint requires that $\hat{l}_i \leq \hat{x}_i \leq \hat{h}_i$ where $\hat{l}_i = \frac{l_i - x_{0,i}}{a_i + b_i}$ and $\hat{h}_i = \frac{h_i - x_{0,i}}{a_i + b_i}$.

Taking the sum of all $\hat{x}_i(t)$ gives

$$\hat{x}(t) = \sum_{i=1}^{n} \hat{x}_i(t) = -\sum_{i=1}^{n} d_i t + \sum_{i=1}^{n} t_{\mathrm{on},i} \tag{3}$$

with the constraint that $\hat{l} = \sum_{i=1}^{n} \hat{l}_i \leq \hat{x}(t) \leq \hat{h} = \sum_{i=1}^{n} \hat{h}_i$. Using (1) we obtain the following inequality

$$\hat{x}(t) \leq \left(1 - \sum_{i=1}^{n} d_i\right) t = (1 - d)t \tag{4}$$

in which $d = \sum_{i=1}^{n} d_i > 0$. Inequality (4) is very important since it gives us the condition on the schedulability of the set of tasks.

Theorem III.1. *If $d > 1$, the set of tasks is not schedulable (by any scheduling policy).*

Proof: If $d > 1$ then $1 - d < 0$. By inequality (4), $\hat{x}(t)$ is bounded above by a strictly decreasing function $(1 - d)t$. By time $\hat{l}/(1 - d) \geq 0$ at the latest, $\hat{x}(t)$ will violate the stability condition. ∎

Particularly, in the case $n = 2$, if $a_1 > b_1$ (i.e., discharge rate is faster than charge rate) and $\frac{b_2}{a_2} < \frac{a_1}{b_1}$ then $d > 1$ and the tasks are not schedulable.

B. Feasibility result

In this section, we present feasibility result for a set of linear tasks on an ideal platform where there is no technical constraint on when a task can be scheduled. Hence, on this platform, a task can be switched ON and OFF arbitrarily fast or slow as long as it satisfies the stability constraint.

A feasibility condition on the initial states $x_{0,i}$ is stated in Proposition III.2, which is very straightforward.

Proposition III.2. *If there are two different tasks T_i and T_j, $i \neq j$, such that $x_{0,i} = l_i$ and $x_{0,j} = l_j$ then the set of tasks is not schedulable.*

If more than one task starts at their lower thresholds then there exists no valid schedule because at least one task will violate its stability constraint regardless of the schedule.

We proceed with the main feasibility theorem.

Theorem III.3. *Given a set of tasks $\{T_i\}_{i=1\ldots n}$. If d_i is rational for all $i = 1 \ldots n$, $\sum_{i=1}^{n} d_i \leq 1$, and at most one task starts at its lower threshold then the set of tasks is schedulable on an ideal platform.*

Proof: We can assume that at the beginning, all tasks do not start at their thresholds, i.e., $l_i < x_{0,i} < h_i$ for $i = 1 \ldots n$. Indeed, if there are tasks starting at their thresholds, among which at most one can start at its lower threshold, we can always schedule the tasks for a short period of time $\tau > 0$ so that $l_i < x_i(\tau) < h_i$, $i = 1 \ldots n$.

For each task T_i, since d_i is rational, it can be written as $d_i = \frac{m_i}{n_i}$ where $m_i, n_i \in \mathbb{N}$. Let N be a common

multiple of all n_i, for $i = 1 \dots n$, we can write $d_i = \frac{M_i}{N}$ for some $M_i \in \mathbb{N}$. We will discretize time with period $\Delta_T > 0$ and design a periodic schedule for the set of tasks on the discrete-time platform. The schedule is specified by an ordering sequence $\rho = (\rho_1, \dots, \rho_N)^\omega$ of length N, where each $\rho_k \in \{0, 1, \dots, n\}$ indicates which task is ON during interval k, and the superscript ω means that the sequence is repeated indefinitely. If $\rho_k = 0$ then all tasks will be OFF at the start and during interval k, otherwise task T_{ρ_k} will be ON and all the other tasks will be OFF. Note that the ordering sequence ρ and the time period Δ_T are decoupled and independent of each other.

Choose any $\Delta_T > 0$, for example $\Delta_T = 1\,\text{sec}$. Consider the sequence ρ of N numbers. We assign to each ρ_k, $k = 1 \dots N$, an integer between 0 and n so that for each integer $i \in \{1, \dots, n\}$, the number of times it appears in ρ is exactly M_i. Because $\sum_{i=1}^n d_i \leq 1$, $\sum_{i=1}^n M_i \leq N$. It follows that we can always construct ρ to satisfy the above condition.

The state of task T_i at instant $t_k = k\Delta_T$, $k = 0, 1, \dots, N$, can be computed as

$$\begin{aligned} x_i(t_k) &= x_{i,0} + n_{\text{on}}\Delta_T b_i - (k - n_{\text{on}})\Delta_T a_i \\ &= x_{i,0} + \Delta_T \left[n_{\text{on}} b_i - (k - n_{\text{on}}) a_i\right] \end{aligned}$$

where n_{on} is the number of time intervals from the beginning until step k during which T_i is ON. If the same schedule ρ is used but a different time period $\Delta'_T > 0$ is chosen, the new states at instant $t'_k = k\Delta'_T$ can be written as:

$$x'_i(t'_k) = x_{i,0} + \Delta'_T \left[n_{\text{on}} b_i - (k - n_{\text{on}}) a_i\right].$$

We then have the equation:

$$x'_i(t'_k) = x_{0,i} + \frac{\Delta'_T}{\Delta_T}(x_i(t_k) - x_{0,i}).$$

For each $k = 1, 2, \dots, N-1$, define $\Delta_{T,k}$ as follows:

- If $l_i \leq x_i(t_k) \leq h_i$ then $\Delta_{T,k} = +\infty$;
- If $x_i(t_k) < l_i$ then $\Delta_{T,k} = \Delta_T \frac{l_i - x_{0,i}}{x_i(t_k) - x_{0,i}}$ where $\Delta_{T,k} > 0$ because $x_{0,i} > l_i$;
- If $x_i(t_k) > h_i$ then $\Delta_{T,k} = \Delta_T \frac{h_i - x_{0,i}}{x_i(t_k) - x_{0,i}}$ where $\Delta_{T,k} > 0$ because $x_{0,i} < h_i$.

Let $\Delta^\star_T = \min\{\Delta_{T,1}, \dots, \Delta_{T,N-1}\} > 0$. If the same schedule ρ is used with time period $\Delta^\star_T$, which yields states $x^\star_i(t^\star_k)$ where $t^\star_k = k\Delta^\star_T$, it is straightforward to show that:

- At the end of the sequence ρ, $x^\star_i(t^\star_N) = x_{i,0} + \Delta_T \left[M_i b_i - (N - M_i) a_i\right] = x_{i,0}$;
- For all $k = 0, 1, \dots, N$, $l_i \leq x^\star_i(t^\star_k) \leq h_i$.

Thus, for all $0 \leq t \leq N\Delta^\star_T$, $x_i(t)$ stays within the stability region, and it goes back to the initial state at the end of the sequence ρ. It follows that ρ with time period $\Delta^\star_T$ is a valid schedule for the set of tasks. ■

IV. Case study: Energy Router

A primary design goal for energy-efficient buildings is the *fine-grained coordination of energy demand* so as to minimize the peak power consumption while maintaining acceptable functionality and comfort. As explained earlier with our *superbowl* example, when major appliances in buildings are all switched on at the same time, it causes spikes in the power consumptions which results in exorbitant electricity bills. Electricity producing companies do not charge by average amount of power consumed but by the peak energy usage in short 15-30 minute spans. Just as most commercial buildings have an IP router for managing data traffic, our algorithms can be easily incorporated within an "energy router" meant for coordinating and constraining energy demands to maintain a minimum Peak-to-Average Ratio (PAR) at fine time steps.

V. Roadmap

So far we have established feasibility conditions for a set of linear tasks. As future work we want to formulate theory which provides constraints on initial conditions. We also want to develop tighter constraints on schedulability of the system. Some of the theorems that we want to formulate are:

1) **Critical Instant Theorem:** Given a set of feasible tasks, they are schedulable by the scheduling policy S, *iff* every task is schedulable at its critical instant. We can have another theorem that defines when a critical instant of a task occurs.
2) **Optimality:** For a certain set of tasks then it is schedulable by scheduling policy S if and only if it has a feasible schedule.
3) **Acceptability test:** For a feasible set of tasks, being scheduled according to scheduling policy S, we can add another task T_k in that schedule *iff* that task satisfies the acceptability criteria.
4) Develop a scheduling algorithm that can schedule a set of feasible tasks.
5) Provide utilization/performance bounds for the above algorithm.
6) Schedule with respect to additional constraints on operational efficiency of the system being scheduled.

References

[1] Department of Energy, *Buildings Energy Data Book*, Department of Energy, March 2009.

[2] F. W. Payne. *Energy management control system handbook*. Fairmont Press, 1984.