Survey of Weakly-hard Constraints on Distributed Embedded Control Systems

Martin Meng
The University of North Carolina at Chapel Hill
martinmq@live.unc.edu

***Abstract*—Weakly hard models have been utilized to optimize the design of real-time distributed embedded control systems. This survey serves as an overview of research topics related to weakly hard models with an emphasis on their applications to control systems. Specifically, this survey proposes an approach to leverage the weakly hard model to enhance the robustness of distributed embedded control systems in automatic vehicles. This survey also summaries various methods to compute weakly hard guarantees, studies the impact of deadline miss on the stability of control systems, introduces cases in which weakly hard models are adopted to improve the security and fault tolerance of systems, and studies Mix Integer Programming problems in the context of real-time distributed systems.**

***Index Terms*—Distributed Systems, Embedded Systems, Controller Design, Real-time Systems, Weakly Hard Models, Cyber-Physical Systems, Security**

I. Introduction and Literature Review

A. Embedded Systems

In contrast to general-purpose computers such as personal computers which are designed to perform heterogeneous functions on one machine, embedded systems are designed to perform specific functions and are often components of larger systems. Embedded computers are the most prevalent type of computers in the current society with a wide range of applications including, but not limited to, control systems such as temperature control system in air conditioners, cruise control system in modern vehicles, and air-traffic control systems in airports. Because embedded systems are ubiquitous, it is important to optimize the design of embedded systems in terms of quality of service, energy consumption, security, robustness, and fault tolerance, etc, while guaranteeing the correctness of the systems. Because embedded systems are often parts of larger systems in which computation and communication resources are limited, it is important to optimize the resource usage for embedded systems. This paper studies a way to perform these optimization on the embedded systems designs by relaxing timing constraints imposed on them.

B. Real-time Constraints

A characteristic of embedded systems is that they ought to satisfy real-time constraints. In the literature of real-time systems, a timing constraint for a task is defined as the deadline for each instance of the task. A constraint is *hard* if the deadline has to be met, and *soft* if the deadline can be missed occasionally. Correspondingly, a *hard* real-time model must meet all of its deadlines whereas a *soft* real-time model can allow occasional deadline misses [1].

However, the dichotomous classification between hard and soft constraints is not sufficient to formulate real-time embedded systems. On one hand, a soft real-time model cannot provide guarantees to performance and correctness requirements of some systems such as the stability requirement of some embedded controllers. On the other hand, a hard real-time model is too pessimistic for some systems that can tolerant bounded deadline misses. Given the limited computation and communication resources of embedded systems, it is a waste of resources to model such systems with hard real-time constraints. Moreover, it is possible that a tasks set that is not schedulable under the hard real-time model may become schedulable under a less-restricted real-time model. Therefore, we need to consider a timing constraint that is in the middle of hard and soft constraint. This paper studies *weakly-hard* constraints, timing constraints that are weaker than hard constraints in that they allow deadline misses, but also harder than soft constraints in that the number of deadline misses within a time interval is bounded.

C. Weakly Hard Constraints

Weakly-hard constraints are constraints that allow deadline misses in a predictable manner. Weakly-hard systems are real-time systems whose distribution of met and missed deadlines is precisely bounded. A common model of weakly-hard systems is called (m-k) model, first proposed by Hamdaoui et al. [2]. In the (m, k) model, there are at least m deadlines that are previously met in any consecutive sequence of k tasks (see Definition 1). Later, Bernat et al. [3] generalized the (m, k) model to four types of weakly-hard models that exhaust all possible real-life scenarios (see Section II for more details). Since then, works have been done in computing and verifying weakly-hard guarantees [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], as well as applying the weakly-hard model to optimize the design and implementation of embedded systems [13], [14], [15].

D. Weakly-hard Guarantees

Various methods are proposed to compute the schedulability on weakly-hard systems. Response time analysis [16] is a common way to verify whether a task set is schedulable in the worst case. Broster et al. [4] utilized the technique of response time analysis to provide schedulability guarantees on weakly-hard models. However, the weakly-hard guarantee calculated by the response time analysis is pessimistic. Broster et al. [4] used simulation based experiments to show that many weakly hard constraints are not violated even at a much higher error rate than the limit calculated by the response time based schedulability analysis. Therefore, a better method to compute the weakly hard guarantees is needed.

Typical Worst Case Analysis (TWCA) is a timing analysis technique that can provide a tighter bound than response time analysis for weakly hard models [17]. Hammadeh et al. utilized the TWCA method as well as an Integer Linear Programming (ILP) formulation to calculate the maximum occurrences of deadline misses for tasks with static priority scheduling policy [5]. Xu et al. improved the TWCA technique to calculate the deadline miss model by considering possible scenarios of sporadic overloads leading to occasional deadline misses [18]. TWCA was then applied to calculating weakly-hard guarantees for systems with dependent task chain [19] and with weighted round-robin scheduling policy [8]. Ahrendts et al. combined TWCA with Compositional Performance Analysis (CPA) to compute weakly-hard guarantees for distributed embedded systems with multiple resources [7].

There are methods other than TWCA that are used to calculate weakly-hard schedulability guarantees. Frehse et al. [20] proposed a model checking based hybrid technique for schedulability analysis on weakly-hard systems, but its scalability is limited by the computation complexity of model checking. Sun et al. [6] formulated the weakly-hard schedulability problem as Mixed Integer Linear Programming (MILP) problem for periodic tasks with static priorities without the knowledge of task offsets. However, this method's scalability is limited by the size of tasks because of the complexity of the MILP problem [6]. Pazzaglia et al. extended the scope of this method to include systems allowing jitters and resource sharing [11], but this method still needs a significantly large amount of time to obtain an optimal solution to the large, complex MILP problem. Choi et al. improved the running time of the schedulability analysis with a new scheduling policy that classifies each task based on the number of deadlines that are previously met [12].

To show that the methods for computing weakly-hard constraints are sound, it is necessary to verify the safety and functional correctness of these weakly-hard guarantees on embedded systems. Duggirala et al. verified the safety of weakly hard guarantees on linear control systems by reducing the original verification problem to a software verification problem [21]. Huang et al. extended the scope of the verification from linear systems proposed in [21] to nonlinear system by adopting an over-approximation technique [9]. Huang et al. also developed SAW, a tool that implements their verification method [10].

E. Usage of Weakly Hard Model

Weakly hard models have various application to embedded systems including, but not limited to, intermittent computing systems, multi-rate distributed systems, image processing systems, and control systems. Intermittent computing systems are embedded systems whose computation may be occasionally suspended because of unreliable power supply, resulting in occasional deadline misses [22], [23], [24], [25]. The lack of stable power supply is possible for embedded systems because of the potential scarcity of resource for embedded systems. For example, embedded systems in an automatic vehicle that is crossing a desert are likely to not have continuous, stable power supply for embedded systems on the vehicle. An unstable power supply degenerates the quality of resources which are used by embedded systems. Therefore, design optimization is needed for embedded systems to overcome the challenges imposed by unstable power supply.

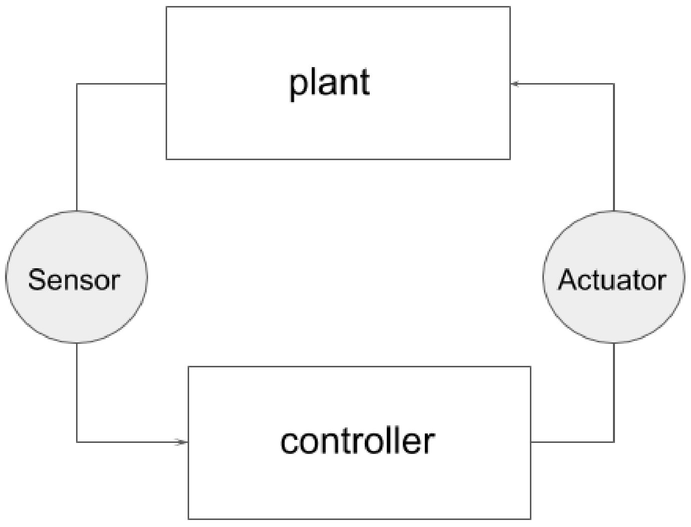
Other than embedded systems with unreliable power supply, weakly hard model is useful for multi-rate distributed embedded systems that are sensitive to message losses. Distributed embedded systems with periodic scheduling policy are possible to have various rates for different components of the systems. For instance, in a distributed embedded control system, the sensor and the controller may operate at different rates. Specifically, bounded deadline misses will occur if the sensor has a higher operating frequency than the controller [26]. Another type of embedded system that can have bounded deadline misses is an image processing system whose deadline misses are caused by the slow rate of human perception [27]. Because these systems suffer from bounded deadline misses, the weakly hard model can provide flexible timing guarantees for these systems.

F. Control Systems

This survey focuses on the application of weakly hard timing constraints on yet another type of embedded systems: control systems. As shown in Figure 1, a control system consists of a physical plant and a controller which ensures certain behaviors of the physical plant. The controller consists of sensors which read data from the physical plant as input, electric computing units (ECUs) which compute control inputs, and actuators which perform operations on the physical plants indicated by the control inputs. We consider a common type of control systems in this paper: linear time-invariant control systems with feedback loops. This type of system is used to maintain the stability of the physical plant and ensure the states of the model of the physical plant to approach a referenced value. Therefore, scheduling policies ought to guarantee the stability and performance requirements of control systems.

The weakly hard model can be a right fit for control systems because of the following intrinsic property of control systems: the stability and performance requirements of control systems can be satisfied despite occasional deadline misses, provided that no more than m deadlines misses can happen in any sequence of k deadlines, which is defined as the m-K model of weakly hard systems (see Definition 3) [2], [28]. Goswami et al. formalized this property by proposing an analytical upper bound on deadline misses that allows distributed embedded control systems to maintain the stability as well as

Fig. 1. Control system model



meet the performance requirements [29]. Control performance under deadline misses is also studied on an optimal linear quadratic regulator controller [30]. Linsenmayer et al. studies methods to design stable controllers given the weakly hard constraints [31]. The degradation of performance caused by the deadline misses on weakly hard systems is studied as well. Pazzaglia et al. showed the impact of various deadline miss patterns on the control performance of the weakly hard system [32]. Lastly, Maggio et al. [15] studied the stability of control systems with a different type of weakly hard models (m-K model, see Definition 3 in Section II). Therefore, the reason why weakly hard model is applicable to control systems is formalized.

G. *Applying Weakly-hard Model to Control Systems*

Since reasons of applying weakly hard model to control systems were discussed, works have been done to exploit the application of weakly hard model on scheduling control tasks to enhance the security and robustness of the system. Liang et al. proposed a co-design approach to use weakly hard constraints on control tasks to improve systems' capability for hosting security monitoring tasks, thus enhancing the security of the system designs [13]. Liang et al. derived the weakly hard constraints on control tasks that satisfy the stability and performance requirement of the control system by applying quantitative analysis techniques. Then, based on the new set of tighter timing constraints, Liang et al. optimized the allocation, priority, and period assignment of the security monitoring tasks. Liang et al. then improved the capability of fault tolerance of the system by utilizing weakly hard constraints as well [14]. These two works have shown that weakly hard model can be utilized to make control systems more secure. This paper will discuss a potential method to make distributed embedded control systems more robust based on the weakly hard model (see Sectio V).

H. *Optimization Problem*

Techniques from optimization problems, such as Integer Linear Programming (ILP) and Mixed Integer Programming (MIP) [33], are widely used in computing weakly hard guarantees and applying weak hard models to embedded systems. Sun et al. [6] and Pazzaglia et al. [11] used ILP formulation to compute the schedulability for periodic tasks on weakly hard systems. Recently, Kumar et al. [34] proposed an ILP based framework to schedule weakly hard tasks on a heterogeneous multiprocessor system in order to minimize energy consumption. However, ILP based model generally does not scale well. For an ILP problem with a significantly large number of constraints, it takes a large amount of time and memory to compute the optimal solution. Techniques from artificial intelligence such as heuristic search may be used to solve this problem, but future work on this topic is needed. An alternative method of ILP is MIP. Zhang [35] proposed a MIP based framework to schedule application tasks and communication tasks in the Ethernet-based systems. His framework allows multiple objectives to be optimized. However, the MIP formulation has not yet been applied to weakly hard models.

I. *Outline*

The rest of the paper is organized as follows. Section II introduces the weakly hard models. Section III illustrates an example to increase the robustness of distributed embedded control systems based on weakly hard models. Section IV presents a framework that uses MIP to solve scheduling problems on distributed systems. Section V discusses a potential method to make distributed embedded control systems more robust. Section VI concludes the paper.

II. Weakly Hard Model

Weakly hard model is a formalized description for systems that can sporadically miss deadlines. Bernat et al. [3] proposed four definitions on weakly hard models.

Definition 1. A task τ *"meets any n in m deadlines"*, if for any sequence of m consecutive jobs of τ, there are at least n jobs that meet the deadline.

Definition 2. A task τ *"meets row n in m deadlines"*, if for any sequence of m consecutive jobs of τ, there are at least a sequence of n consecutive jobs that meet the deadline.

Definition 3. A task τ *"misses any n in m deadlines"*, if for any sequence of m consecutive jobs of τ, there are at most n deadline misses.

Definition 4. A task τ *"misses row n in m deadlines"*, if for any sequence of m consecutive jobs of τ, there are less than n consecutive deadline misses.

These four types of weakly hard models are classified by two metrics. The first metric considers whether the model considers deadlines that are met or missed. The other metric considers whether the missed deadlines or deadlines that are met are required to be consecutive or not. Table I shows this two-dimensional classification.

TABLE I
CLASSIFICATION OF WEAKLY HARD MODELS

Metrics	Met deadlines	Missed deadlines
Consecutive	Definition 2	Definition 4
Any order	Definition 1	Definition 3

The research community has focused on the model *"misses any n in m deadlines"* defined in Definition 3, also called the m-K model. For example, Frehse et al. used model checking technique to analysis the schedulability under m-K model [20]. The analysis of Sun et al. on periodic tasks with static priorities with free offsets is based on m-K model as well [6]. Moreover, designing stable controller on the m-K model has been studied [31].

However, m-K model does not have any constraint on the consecutiveness of the deadline misses, but industrial cases often require consideration of the consecutiveness of deadline misses [15]. Therefore, Maggio et al. considered the model in Definition 4 which bounds the maximum number of consecutive deadline misses. They analyzed the stability of control systems based on this model by considering joint spectral radius [15].

The relationship between the models "misses row n in m deadlines" (Definition 4) and "misses any n in m deadlines" (Definition 3, m-K model) is that the model "misses row n in m deadlines" imposes additional constraints about consecutiveness on the task set and thus is a tighter model. Therefore, if a task set is schedulable on the model "misses any n in m deadlines" (Definition 3), then the task set is guaranteed to be schedulable on the model "misses row n in m deadlines" (Definition 4). However, there exist cases in which a task set is schedulable on the model "misses row n in m deadlines" (Definition 4) but not on the model "misses any n in m deadlines" (Definition 3). For instance, if a task misses n deadlines in a sequence of m consecutive deadlines but the n deadline misses are not consecutive, then this task satisfies the model "misses row n in m deadlines" (Definition 4) but fails to satisfy the model "misses any n in m deadlines" (Definition 3). Because the "misses any n in m deadlines" model (Definition 3) includes the model "misses row n in m deadlines" (Definition 4) in terms of schedulability, and is more popular in the research community, the rest of this survey will consider the m-K model only.

III. Weakly Hard Model and Control Systems

A. Control Systems

This section studies an example [13] to enhance the security and robustness of control systems by leveraging weakly hard constraints. We consider a feedback controller design for physical plants with linear time-invariant model. The dynamics of the system can be formulated by the following state-space model:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \\ u(t) &= Ky(t) \end{aligned}$$

where x(t), y(t), and u(t) are vectors representing the states, the output, and the control input of the system at time t, respectively. A, B, and C are system matrices.

This continuous model of control systems needs to be discretized before being implementing on digital platforms. We assume that computation delay and communication delay are negligible in the system. A typical model of such discrete system is as follows:

$$\begin{aligned} z[k+1] &= A_{aug}z[k] + B_{aug}u[k] \\ y[k] &= C_{aug}z[k] \\ u[k] &= -Kz[k] \end{aligned}$$

where z[k], y[k], and u[k] denote the augmented states, the output, and the control input of the system at time h_k, respectively, where h_k denotes the time t at the k-th cycle. K is the feedback gain of the system that ensures the stability of the system. A_{aug} is a matrix $\in \ R^n \times R^n$, B_{aug} is a matrix $\in R^n \times 1$, and C_{aug} is a matrix $\in \ 1 \times R^n$.

B. Stability and Control Performance

To optimize the design of control systems under the weakly hard model, we need to study the impact of deadline miss pattern on the stability and control performance of the system. A discrete control system as described in the previous section is stable if all the closed-loop poles lie in the unit circle in a complex plane. This system dynamics is described as follows:

$$\begin{aligned} z[k+1] &= (A_{aug} - B_{aug}K)z[k] \\ A_{cl} &= A_{aug} - B_{aug}K \end{aligned}$$

The stability property is formalized as a study of the eigenvalues of the matrix A_{cl}. The closed-loop system is stable if and only if all eigenvalues of A_{cl} are within the unit circle in the complex plane, i.e. their lengths are all less than 1.

While stability considers whether the system can be stabilized, control performance measures how quickly the controller can bring the system back to the equilibrium state after a disturbance, assuming the the system can be stabilized. The control performance is quantified by a metric H. H stands for the minimal number of sampling cycles to bring a disturbance J back to a certain predefined threshold J_{th}. This property is formalized as the following equation:

$$\begin{aligned} &\forall r \geq H \\ &\quad J_r \leq J_th \end{aligned}$$

C. Weakly Hard Constraints

The weakly hard model we take into consideration in this example is the m-K model that allows a task to "misses any n in m deadlines" (Definition 3). We quantitively formalize this timing requirement for a task τ as

$$(k^j, N^j)$$

which represents that for a sequence of N^j consecutive activations of task τ, there are at most k^j deadline misses.

We also denote

$$dmm(N^j)$$

as the number of deadline misses in the worst case for task τ in any sequence of N^j consecutive activations.

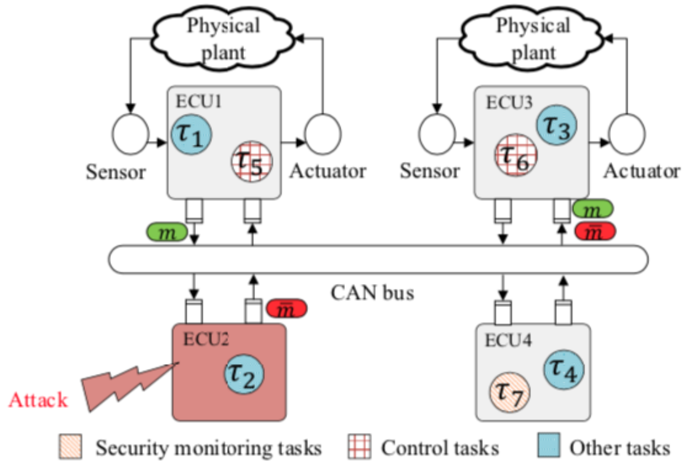
With these notions in hand, we can formally define a task τ satisfying its weakly hard constraint (k^j, N^j) as follows:

$$dmm(N^j) \leq k^j, \forall j$$

D. Security Monitoring Tasks

The goal of this example [13] is to improve the security of the embedded control systems by optimizing the allocation, priority, and period assignments of the security monitoring tasks leveraging the weakly hard constraints. Security monitoring tasks read the CAN messages to detect intrusion. Security monitoring tasks are periodic tasks. For each period, an activation of the security monitoring task reads the CAN messages it monitors and check for anomaly intrusions. Figure 2 shows a model of system with security monitoring tasks.

Fig. 2. System model with security monitoring tasks [13]



Security monitoring tasks impose constraints on the system design. These constraints are of two types. The first type deals with the coverage of security monitoring tasks. For instance, a constraint asserts that all security critical tasks are monitored by at least one security monitoring task. The second type of constraint is about redundancy. To avoid the single-point failure, a security critical CAN message may require to be monitored by multiple security monitoring tasks on different ECUs.

E. Optimization Problem

This example formulates the design problem as a multi-objective optimization problem. The objective of the optimization problem describes the trade-off between control performance and security. The constraints include the stability constraints, control performance constraints, constraints imposed by security monitoring tasks, and schedulability constraints imposed by the weakly hard model.

IV. MIP Formulation

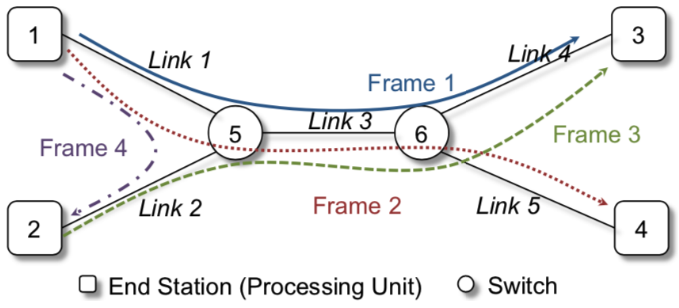
A. Mix Integer Programming

Mix Integer Programming (MIP) is a common optimization technique that is widely used by the research community to study weakly hard models. An optimization problem is called an integer programming problem if its decision variables are constrained to integer values. MIP is a special form of integer programming problem in that some variables, but not all, are constrained to be integers. A common type of decision variables in MIP is binary variable whose value is restricted to be 0 or 1, which formulates yes/no question. However, integer programming is NP-Hard because it is not convex, so it takes a large amount of computation time and memory resources to find an optimal solution. In spite of the complexity, industrial solvers can solve small size MIP problem relatively fast.

B. Ethernet-based Switch Network

This section considers the application-level scheduling problem on a time-triggered, Ethernet based distributed system [35]. Specifically, a switch network is considered. This network can be model as an undirected graph. The nodes of the graph are end stations and switches that forward messages between end stations. An edge between two nodes represents a full-duplex link that allows simultaneous message transmissions between two nodes. An example of this graph-based presentation of a Ethernet-based switch network is shown in Figure 3.

Fig. 3. A model of Ethernet-based Switch Network [35]



C. MIP Formulation

Zhang [35] formulates the application-level scheduling problem on a time-trigger, Ethernet-based switch network as an MIP problem. The tasks to be scheduled on the distributed system can be classified into application tasks that run on the end stations and communication tasks that should be scheduled on the Ethernet network.

The first constraint on the MIP program formulation requires that application tasks are collision-free. The assumption for the distributed system is that each end station only consists of a single processor, so this constraint ensures that only one application task can be run on any end station at any given time stamp. The second constraint ensures that communication tasks are collision-free as well. At most one communication task can be transmitted on one direction of any link at any time stamp.

Path dependency and data dependency are formulated as constraints as well. Path dependency for a communication task is formulated such that each frame has to be forwarded in the correct temporal order defined by its path to ensure that the message reaches the correct end station. Data dependency in application tasks requires application tasks and their corresponding communication tasks to be schedule in the correct temporal order.

Constraints on response time and end-to-end latency on application tasks have also been exploited. Both the response time and the end-to-end latency are formulated as hard constraints, but adopting weakly hard constraints can provide looser constraints on response time, thus expanding the scope of schedulable task sets. Future work can be done on this direction.

Zhang [35] considered multiple objectives in this problem formulation. These objectives include response time and end-to-end latency in the average case and in the worst case, as well as certain timing requirements on specific tasks. The objective function, as a result, is a sum of each weighted sub-objective.

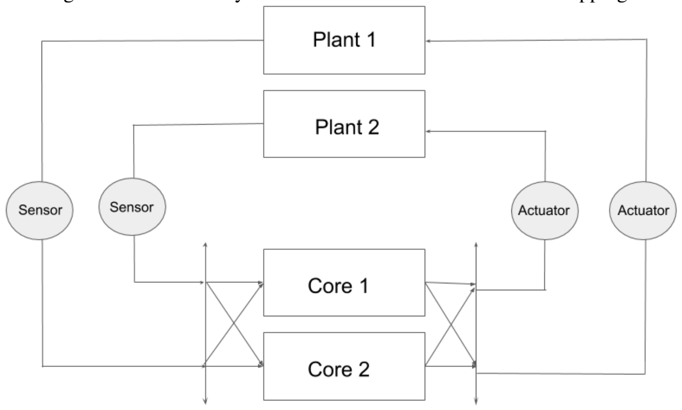
V. Open Question

A. Overview

This section considers an open question: how to enhance the robustness of distributed embedded control systems utilizing the weakly hard model. The solution of this question can be applied to improving the security and robustness of automatic vehicles. A classical method to enhance the robustness of distributed system is to increase the redundancy of the system components so that single-point failure can be avoided. However, embedded control systems, e.g. control systems on automatic vehicles, have limited resources and therefore cannot afford a high level of redundancy. On the other hand, control systems can tolerant a certain level of occasional, bounded deadline misses as demonstrated in [13], [15], [28], [29], [30], [31], [32]. Therefore, we can utilize this property to optimize the assignments of application-level control tasks on processors.

The approach is as follows: first, we analyze the deadline miss pattern for each control task in the m-K weakly hard model (Definition 3). Then, we map the control tasks to processors in a way that the tasks are rotating to a different processor in the next time stamp. Each time a processor encounters fatal errors and stops working, we adjust the mapping to fulfill the stability and control performance requirements for each control task, based on its deadline miss pattern. The mapping problem can be formulated as a MIP problem that optimizes the stability and control performance of the task set. Figure 4 presents a model of two control tasks that are mapped onto two processor cores.

Fig. 4. A model of systems with rotational control tasks mapping



B. Deadline Miss Pattern

We consider the deadline miss pattern for each control task in the m-K weakly hard model (Definition 3). We assume that the system platform adopts a static-priority based, preemptive policy. A task set is defined as follows:

$$\tau = \{\tau_1, \tau_2, ..., \tau_n\}$$

$\forall i \in 1, 2, ..., n$, τ_i is a tuple of the following form:

$$\tau_i = (c_{\tau_i}, d_{\tau_i}, t_{\tau_i}, p_{\tau_i})$$

where c_{τ_i} is the worst-case execution time (WECT), d_{τ_i} is the deadline, t_{τ_i} is the period, and p_{τ_i} is the priority of τ_i.

We can use analysis technique proposed in [13], [29] to analytically calculate an upper bound on the deadline miss pattern for each control task that ensures the stability of the task.

C. Objective Function

We formulate the control tasks mapping problem as an MIP problem. The objective of this MIP formulation is to optimize the stability and control performance of the control tasks. To optimize the stability, we define a variable S such that

$$S = \sum_{i=1}^{n} b_i$$
$$b_i = \begin{cases} 0 & \text{if } \tau_i \, is \, stable \\ 1 & \text{if } \tau_i \, is \, unstable \end{cases}$$

Here, b_i is a binary decision variable that represents whether task τ_i is table or not. Then, by minimizing S, we can have the maximum number of tasks that fulfill the stability requirement.

For control performance requirement, we can use the method described in Section III-B, as well as in [13], to describe the number of minimum required sample intervals needed to bring the system which is suffered from a disturbance back to the equilibrium state. We use H to describe the control performance for all tasks. Thus, the objective function is:

$$minimize\ \alpha S + \beta H$$

where α and β are predefined weights that are used to study the trade-off between two objectives.

D. Constraints

We now attempt to create constraints for the MIP formulation. The first type of constraints ensures that control tasks and communication tasks are collision-free, so that at any given time stamp, at most one control task will be running on any processor core, and at most one communication task will be transmitted on any communication bus. The formulation of a similar type of constraint is proposed in [35].

We then formulate constraints that encode the data dependency requirement for control tasks as well as path dependency requirement for communication tasks. Data dependency ensures that control tasks and their corresponding communication tasks are executed in the correct temporal order. Path dependency similarly impose temporal order to communication tasks so that they can be transmitted to the correct destination. Zhang [35] provides a general type of dependency constraint that can be of use in this solution.

Next, we encode timing constraints for the response time and end-to-end latency of each task. We use the method introduced in Section III-C [13] to set weakly hard constraints on tasks, and for those tasks not schedulable on the weakly hard model, hard constraints with a predefined maximum time requirement can be used.

Constraints are needed to ensure that the control tasks mapping will be changed from a time stamp to the next one. If we run the mapping algorithm every time stamp, so on each time stamp, we can encode the current mapping to be different from a predefined constant N which represents the number of past mappings in the most recent time stamps. However, the time to run the mapping algorithm on each time stamp is expensive in terms of time, and we will need to take this delay into our consideration. We can use heuristic search algorithm to aid the solving process of the MIP problem. An alternative approach is to analytically solve the MIP problem once with a default mapping to be in a round-robin manner. We predefine a temporal order of processor cores that each task by default is mapped to for each round. In this way, we don't need to run the MIP problem in run time and we do not need to have constraints about the order of processors each task is mapping to at every time stamp.

VI. Conclusion

This survey has reviewed recent works related to weakly hard constraints with an emphasis on its application to control systems. Various methods to compute and verify weakly-hard guarantees are studied. We focus on weakly hard model's applications to distributed embedded control systems such that the security and robustness of the systems are enhanced. We also study MIP formulation, an optimization technique common in the community of real-time systems. Lastly, we propose a new approach to leverage the weakly hard model to enhance the robustness of distributed embedded control systems in automatic vehicles.

References

[1] J. Liu, *Real-Time Systems*. Prentice Hall, 2000. [Online]. Available: https://books.google.com/books?id=855QAAAAMAAJ

[2] M. Hamdaoui and P. Ramanathan, "A dynamic priority assignment technique for streams with (m, k)-firm deadlines," *IEEE transactions on Computers*, vol. 44, no. 12, pp. 1443–1451, 1995.

[3] G. Bernat, A. Burns, and A. Liamosi, "Weakly hard real-time systems," *IEEE Transactions on Computers*, vol. 50, no. 4, pp. 308–321, 2001.

[4] I. Broster, G. Bernat, and A. Burns, "Weakly hard real-time constraints on controller area network," in *Proceedings 14th Euromicro Conference on Real-Time Systems. Euromicro RTS 2002*. IEEE, 2002, pp. 134–141.

[5] Z. A. Hammadeh, S. Quinton, and R. Ernst, "Extending typical worst-case analysis using response-time dependencies to bound deadline misses," in *Proceedings of the 14th International Conference on Embedded Software*, 2014, pp. 1–10.

[6] Y. Sun and M. D. Natale, "Weakly hard schedulability analysis for fixed priority scheduling of periodic real-time tasks," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 16, no. 5s, pp. 1–19, 2017.

[7] L. Ahrendts, S. Quinton, T. Boroske, and R. Ernst, "Verifying weakly-hard real-time properties of traffic streams in switched networks," in *ECRTS 2018-30th Euromicro Conference on Real-Time Systems*, 2018, pp. 1–22.

[8] Z. A. Hammadeh and R. Ernst, "Weakly-hard real-time guarantees for weighted round-robin scheduling of real-time messages," in *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1. IEEE, 2018, pp. 384–391.

[9] C. Huang, W. Li, and Q. Zhu, "Formal verification of weakly-hard systems," in *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, 2019, pp. 197–207.

[10] C. Huang, K.-C. Chang, C.-W. Lin, and Q. Zhu, "Saw: A tool for safety analysis of weakly-hard systems," *arXiv preprint arXiv:2005.07159*, 2020.

[11] P. Pazzaglia, Y. Sun, and M. Di Natale, "Generalized weakly hard schedulability analysis for real-time periodic tasks."

[12] H. Choi, H. Kim, and Q. Zhu, "Job-class-level fixed priority scheduling of weakly-hard real-time systems," in *2019 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2019, pp. 241–253.

[13] H. Liang, Z. Wang, D. Roy, S. Dey, S. Chakraborty, and Q. Zhu, "Security-driven codesign with weakly-hard constraints for real-time embedded systems," in *2019 IEEE 37th International Conference on Computer Design (ICCD)*, 2019, pp. 217–226.

[14] H. Liang, Z. Wang, R. Jiao, and Q. Zhu, "Leveraging weakly-hard constraints for improving system fault tolerance with functional and timing guarantees," *arXiv preprint arXiv:2008.06192*, 2020.

[15] M. Maggio, A. Hamann, E. Mayer-John, and D. Ziegenbein, "Control-system stability under consecutive deadline misses constraints," in *32nd Euromicro Conference on Real-Time Systems (ECRTS 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[16] M. Joseph and P. Pandya, "Finding response times in a real-time system," *The Computer Journal*, vol. 29, no. 5, pp. 390–395, 1986.

[17] S. Quinton, M. Hanke, and R. Ernst, "Formal analysis of sporadic overload in real-time systems," in *2012 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2012, pp. 515–520.

[18] W. Xu, Z. A. H. Hammadeh, A. Kröller, R. Ernst, and S. Quinton, "Improved deadline miss models for real-time systems using typical worst-case analysis," in *2015 27th Euromicro Conference on Real-Time Systems*, 2015, pp. 247–256.

[19] Z. A. H. Hammadeh, R. Ernst, S. Quinton, R. Henia, and L. Rioux, "Bounding deadline misses in weakly-hard real-time systems with task dependencies," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, 2017, pp. 584–589.

[20] G. Frehse, A. Hamann, S. Quinton, and M. Woehrle, "Formal analysis of timing effects on closed-loop properties of control software," in *2014 IEEE Real-Time Systems Symposium*. IEEE, 2014, pp. 53–62.

[21] P. S. Duggirala and M. Viswanathan, "Analyzing real time linear control systems using software verification," in *2015 IEEE Real-Time Systems Symposium*, 2015, pp. 216–226.

[22] V. Talla, B. Kellogg, B. Ransford, S. Naderiparizi, S. Gollakota, and J. R. Smith, "Powering the next billion devices with wi-fi," in *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, 2015, pp. 1–13.

[23] D. Balsamo, A. S. Weddell, A. Das, A. R. Arreola, D. Brunelli, B. M. Al-Hashimi, G. V. Merrett, and L. Benini, "Hibernus++: a self-calibrating and adaptive system for transiently-powered embedded devices," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 12, pp. 1968–1980, 2016.

[24] A. Colin and B. Lucia, "Chain: tasks and channels for reliable intermittent programs," in *Proceedings of the 2016 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications*, 2016, pp. 514–530.

[25] B. Lucia, V. Balaji, A. Colin, K. Maeng, and E. Ruppel, "Intermittent computing: Challenges and opportunities," in *2nd Summit on Advances in Programming Languages (SNAPL 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[26] W. Li, L. Gérard, and N. Shankar, "Design and verification of multi-rate distributed systems," in *2015 ACM/IEEE International Conference on Formal Methods and Models for Codesign (MEMOCODE)*. IEEE, 2015, pp. 20–29.

[27] M. Yang, J. Crenshaw, B. Augustine, R. Mareachen, and Y. Wu, "Adaboost-based face detection for embedded systems," *Computer Vision and Image Understanding*, vol. 114, no. 11, pp. 1116–1125, 2010.

[28] P. Ramanathan, "Overload management in real-time control applications using (m, k)-firm guarantee," *IEEE Transactions on parallel and distributed systems*, vol. 10, no. 6, pp. 549–559, 1999.

[29] D. Goswami, R. Schneider, and S. Chakraborty, "Relaxing signal delay constraints in distributed embedded controllers," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 6, pp. 2337–2345, 2014.

[30] E. P. van Horssen, A. R. B. Behrouzian, D. Goswami, D. Antunes, T. Basten, and W. P. M. H. Heemels, "Performance analysis and controller improvement for linear systems with (m, k)-firm data losses," in *2016 European Control Conference (ECC)*, 2016, pp. 2571–2577.

[31] S. Linsenmayer and F. Allgower, "Stabilization of networked control systems with weakly hard real-time dropout description," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 4765–4770.

[32] P. Pazzaglia, L. Pannocchi, A. Biondi, and M. Di Natale, "Beyond the weakly hard model: Measuring the performance cost of deadline misses," in *30th Euromicro Conference on Real-Time Systems (ECRTS 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

[33] A. Schrijver, *Theory of linear and integer programming*. John Wiley & Sons, 1998.

[34] N. Kumar, J. Mayank, and A. Mondal, "An ilp framework for energy optimized scheduling for weakly-hard real-time systems: work-in-progress," in *Proceedings of the International Conference on Embedded Software Companion*, 2019, pp. 1–2.

[35] L. Zhang, "Synthesizing communication-centric automotive cyber-physical systems," 2018.