An Adaptive Approach to Reduce Control Delay Variations

Shengyan Hong and Xiaobo Sharon Hu
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556
{shong3,shu}@nd.edu

M.D. Lemmon
Department of Electrical Engineering
University of Notre Dame
Notre Dame, IN 46556
lemmon@nd.edu

***Abstract*—For many control systems, control performance is strongly dependent on delay variations of the control tasks. Such variations can come from a number of sources including task preemptions, variations in task workloads and perturbations in the physical environment. Existing work has considered improving control task delay variations due to task preemption only. This paper presents a general adaptive framework that incorporates a powerful heuristic aiming to further reduce delay variations. Preliminary results indicate that the heuristic significantly improves existing approaches.**

I. Introduction

For many cyber-physical systems, intelligent coordination between control design and its corresponding computer implementation can lead to improved control performance and/or reduced resource demands [1], [11], [14]. A prime example that benefits from such coordination is regulating delay variations (jitter) in control tasks. For many control systems, control performance strongly depends on delay variations in control tasks. Such variations can come from numerous sources including task preemptions, variations in task workloads and perturbations in the physical environment, and can cause degraded control system performance, such as sluggish response and erroneous behavior.

There are a number of published papers related to reducing delay variations. In [2], [5], the authors proposed a task decomposition based approach where each task is partitioned into three subtasks and the delay variation of the final subtask (corresponding to control update) is minimized. A somewhat indirect way of reducing delay variations is to reduce task deadlines, which has been investigated by many researchers, e.g., [3], [4], [7], [12]. A common theme of all these methods is to focus on reducing deadlines of either tasks or subtasks. Because deadlines are only allowed to be reduced, these methods cannot effectively explore the design space where deadlines of certain tasks/subtasks may be increased (within some upper bounds) to reduce the overall delay variations.

The task-decomposition based methods [2], [5] suffer less, but still obvious performance degradation (compared with direct deadline reduction methods) when deadlines are only allowed to be decreased greedily. The decomposition task model is acceptable for control tasks where only a small amount of data needs to be passed to control update subtasks, otherwise context switching cost could be prohibitive. In addition, these methods require repeated worst-case response time computation under EDF, which can be time consuming and unsuitable for on-line use. On-line adjustment is needed to reduce delay variations when parameters such as task periods change in response to environment perturbations.

In this paper, we present an on-line adaptive approach which directly minimizes delay variations for both decomposable and non-decomposable control tasks simultaneously. The approach leverages the IMF based unified model for both types of tasks and formulates the delay variation minimization problem as an optimization problem. An efficient algorithm is designed based on the generalized elastic scheduling heuristic [10]. The efficiency of the algorithm readily supports an adaptive framework which can adjust deadlines of control tasks on-line in response to dynamic changes in workloads.

II. Preliminaries

In this section, we first introduce necessary notation and scheduling properties and then present some motivation for the problem to be solved.

A. System Model

We consider a computer system which needs to handle a set Γ of N real-time control tasks, $\{\tau_1, \tau_2, \cdots, \tau_N\}$, each with the following attributes: (C_i, D_i, P_i), where C_i is the worst case execution time (WCET) of τ_i, D_i is τ_i's deadline, P_i is its period, and $C_i \leq D_i \leq P_i$. Without loss of generality, we adopt the IMF task modeling approach introduced in [5]. Specifically, we let τ_i be composed of three subtasks, the initial part τ_{ii} for sampling input data, the mandatory part τ_{im} for executing the control algorithm, and the final part τ_{if} to deliver the control action. Thus, a task set Γ_{IMF} consists of $3N$ subtasks $(\tau_{1i}, \tau_{1m}, \tau_{1f}, ..., \tau_{Ni}, \tau_{Nm}, \tau_{Nf})$, each with the following parameters

$$\tau_{ii} = \{C_{ii}, D_{ii}, P_i, O_{ii}\}$$
$$\tau_{im} = \{C_{im}, D_{im}, P_i, O_{im}\}$$
$$\tau_{if} = \{C_{if}, D_{if}, P_i, O_{if}\}$$

where $O_{i\star}$ is the offset of the corresponding subtask. Note that in order for the IMF model to faithfully represent the original task set, each τ_{ii} must be executed before τ_{im}, which must in turn be executed before τ_{if}. For a non-decomposable task, say τ_i, we simply have $C_{ii} = C_{im} = D_{ii} = D_{im} = 0$, and $C_{if} = C_i$. Some tasks may also be partially decomposable, i.e., we may have non-zero C_{ii} and D_{ii} but $C_{im} = D_{im} = 0$.

To achieve desirable control performance, control actions should be delivered at regular time intervals. However, preemptions, variations in task workloads, and perturbations in the physical environment make each instance of the control actions experience different delays. Similar to [5], we define the delay variation as the difference between the worst and best case response times of the same final subtask relative to its period, i.e.,

TABLE I: A motivational example containing four tasks with the first two tasks being indecomposable.

Task name	Computation Exec. time	Deadline	Period	Original Delay Variations(%)	Delay Variations DRB / TBB / ADVR(%)	Delay Variations before Reassignment DRB / TBB / ADVR(%)	Delay Variations after Reassignment DRB / TBB / ADVR(%)
Speed	5000	27000	27000	18.52	38.98 / 18.52 / 3.7	38.98 / 18.52 / 3.7	41.48 / 18.52 / 3.7
Strength	8000	30000	320000	1.56	2.89 / 1.56 / 3.37	28.91 / 15.63 / 33.68	31.25 / 15.63 / 22.81
Position	10000	45000	50000	32	31.75 / 26 / 0.27	31.75 / 26 / 0.27	34 / 26 / 2.34
Sense	13000	60000	70000	40	32.86 / 20 / 8.57	32.86 / 20 / 8.57	32.86 / 20 / 8.57

$$DV_i = \frac{WCRT_{if} - BCRT_{if}}{P_i}, \tag{1}$$

where $WCRT_{if}$, $BCRT_{if}$ are the worst case response time and best case response time, respectively. Our problem then is to minimize the delay variations of all the final subtasks.

We use Earliest Deadline First (EDF) scheduling algorithm. A necessary and sufficient condition for a synchronous task set to be schedulable under EDF is given below.

Theorem 1. *A set of synchronous periodic tasks with relative deadlines less than or equal to periods can be scheduled by EDF if and only if* $\forall L \in K \cdot P_i + D_i \leq \min(L_{ip}, H, B_p)$ *the following constraint is satisfied,*

$$L \geq \sum_{i=1}^{N} (\lfloor \frac{L - D_i}{P_i} \rfloor + 1) \cdot C_i \tag{2}$$

where $L_{ip} = \frac{\sum_{i=1}^{N}(P_i - D_i)U_i}{1-U}, U_i = \frac{C_i}{P_i}, U = \sum_{i=1}^{N} \frac{C_i}{P_i}$, $K \in N$*(the set of natural numbers including 0),* H *is the hyperperiod, and* B_p *is the busy period [6], [9].*

For an asynchronous task set, the condition in Theorem 1 can be used as a sufficient condition [6].

B. Motivation

We use a simple robotic example, similar to the one in [5], to illustrate the deficiencies of existing approaches for delay variation reduction. The example contains four control tasks, i.e., the speed, strength, position and sense tasks. The tasks and the original delay variations under EDF are shown in columns 1 to 5 of Table I. We consider two representative methods for delay variation reduction: a deadline reduction based method from [4], denoted as DRB, and a task decomposition based method from [5], denoted as TBB.

Suppose that decomposing the speed and strength tasks would cause non-negligible context switch overhead and we opt to only partition the position and sense tasks according to the IMF model. Assume that the IMF decomposition is made considering that the initial and final subtasks consume 10% of the execution time of corresponding control task. By applying the DRB and TBB methods, new delay variation values are obtained and are shown as the first two values in column 6 of Table I. It is easy to observe that the TBB method is more effective in reducing delay variations than the DRB method (which does not even provide much improvement over the original delay variations). However, with the TBB method, two tasks still suffer about 20% or more delay variation.

Now, assume that at some time interval, the execution rate of the strength task increases by 10 times. If the same deadline assignments are used for the tasks/subtasks, the delay variation of the strength task increases to 28.91% and 15.63% for DRB and TBB, respectively (see the first two values in column 7 of Table I). Suppose we apply the DRB and TBB methods online in response to the period change, the new delay variation values are shown in column 8 of Table I. It turns out that the TBB does not improve the delay variations while DRB actually gives worse delay variations.

With our proposed approach (ADVR), better delay variations can be obtained for all the cases considered above. In particular, for each respective scenario, we have applied our approach and the delay variation values are shown as the third number in columns 6-8 of Table I. Though for some tasks, delay variations see a small increase, most of the tasks which suffer from large delay variations due to the other methods are now having much smaller delay variations.

III. Our Approach

From the previous section, one can see that delay variations could be improved significantly if more appropriate deadline assignments can be identified. In this section, we describe our proposed adaptive delay variation reduction (ADVR) approach. ADVR is built on three basic elements. First, the general IMF model as given in the last section is used for both decomposable and non-decomposable tasks. Second, the delay variation reduction problem is formulated as an optimization problem. Third, an efficient heuristic is developed to solve the optimization problem. The heuristic is then incorporated into a simple adaptive framework.

We adopt the generalized IMF model described in Section II to represent the task set under consideration. The general IMF model allows both decomposable and non-decomposable tasks to be treated equivalently. Given an IMF task set, there may exist numerous sets of feasible deadlines (D_{ii}, D_{im}, D_{if}) which allow the original task set to be schedulable. However, different sets of deadlines could lead to different delay variations of the original tasks. To find the particular subtask deadline assignment that results in the minimum delay variation, we formulate the deadline selection problem as a constrained optimization problem. Though existing work such as [10], [13] has considered the deadline selection problem as an optimization problem, there are two major differences between our present formulation and theirs. First, our formulation directly minimizes delay variations. Second and more importantly, our formulation leverages special properties of the IMF task model and thus allows much more effective delay variation reduction.

The delay variation minimization problem is to minimize the total delay variation bounds of (1) subject to the schedulability constraints as given in (2) while considering the IMF task model. Specifically, we have

$$\text{min: } \sum_{i=1}^{N} w_i (D_{if} - C_{if})^2 \tag{3}$$

$$\text{s.t. } \sum_{i=1}^{N} [(\lfloor \frac{L - D_{ii}}{P_i} \rfloor + 1) \cdot C_{ii} + (\lfloor \frac{L - D_{im}}{P_i} \rfloor + 1) \cdot C_{im}$$

$$+(\lfloor\frac{L-D_{if}}{P_i}\rfloor+1)\cdot C_{if}] \leq L, \forall L \in K \cdot P_i + D_i \leq L_{ip}, \quad (4)$$

$$D_{ii} = D_{im}, \quad (5)$$

$$C_{if} \leq D_{if} \leq \min(D_{im}, D_i - D_{im}), \quad (6)$$

$$C_{im} \leq D_{im} \leq D_i - C_{if}, \quad (7)$$

where L_{ip} and K is defined in Theorem 1. If task τ_i is not decomposable, (6) and (7) are replaced by

$$C_{if} \leq D_{if} \leq D_i, \quad (8)$$

$$D_{im} = 0. \quad (9)$$

To see why the above formulation can lead to valid deadline assignments that minimizes delay variations, first note that deadline D_{if} is the upper bound of the WCRT of the final subtask of τ_i, and C_{if} is the lower bound of the BCRT of the final subtask of τ_i. By setting $w_i = \frac{1}{P_i^2}$, the objective function in (3) is the upper bound on the delay variation squares as defined in (1). (By using w_i instead of P_i^2 directly, we can also capture the relative importance of control tasks in the objective function.)

To guarantee schedulability under the IMF model, we have introduced a set of constraints in our formulation. Constraint (4) helps ensure the schedulability of the task set according to Theorem 1. However, this constraint alone is not sufficient since there exist dependencies when executing the initial, mandatory and final subtasks of any decomposable task. Note that ensuring the subtask dependencies during task execution is straightforward. The difficulty lies in capturing this in the schedulability test without being overly pessimistic.

To handle the unique challenges due to the IMF task model, we have added several more constraints in addition to (4). To capture the fact that τ_{ii} is always executed before τ_{im}, we can set $D_{ii} \leq D_{im}$ (and hence τ_{ii} has a higher priority than τ_{im} as long as $O_{ii} = O_{im} = 0$). For simplicity, we let $D_{ii} = D_{im}$, assuming that a tiebreak goes to τ_{ii}, which leads to constraint (5). This simplification is based on the observation that constraint (4) is satisfied by a task set with $(D_{ii} = D_{im}, D_{im}, D_{if})$ for $i = 1, ..., N$ if it is satisfied by a task set with $(D'_{ii}, D_{im}, D_{if})$ for $D'_{ii} \leq D_{im}$ and $i = 1, ..., N$.

Since τ_{if} must start after τ_{im} is completed, we let $O_{if} = D_{im}$. Furthermore, to guarantee that task τ_i finishes by its deadline D_i, we must have $O_{if} + D_{if} \leq D_i$. We automatically have $D_{if} \leq D_i - D_{im}$, which leads to one part of (6). The other part of (6), i.e., $D_{if} \leq D_{im}$ reflects the desire that smaller deadlines should be assigned to the final subtask compared to that of the mandatory subtask so as to help the delay variation reduction of the final subtask (as this would be the delay variation of interests). (7) constrains the space of D_{im} and is obtained simply by combining $D_{im} \leq D_i - D_{if}$ and $D_{if} \geq C_{if}$.

Solving the optimization problem specified in (3)-(7) is not trivial as it involves dealing with a discontinuous function (the floor function). Heuristic techniques such as the one presented in [10] can be used to solve the problem, but the computation time can be quite long and the solutions may not always of high quality. We have devised an improved heuristic by making use of some observations specific to the particular problem at hand and also by designing a more effective search method. Due to the page limit, we only discuss some of the key observations below.

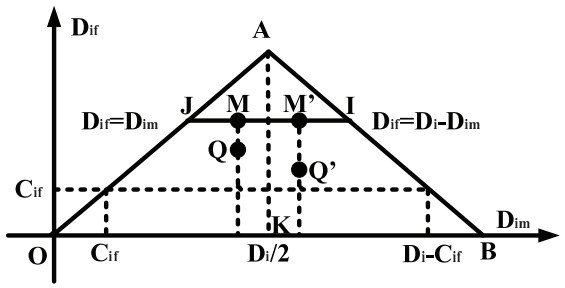

Fig. 1: Feasible deadline region for mandatory/final tasks.

Based on constraints (5)-(7), Figure 1 depicts the feasible region of (D_{im}, D_{if}), which is bounded by $\triangle\ ABO$. However, in $\triangle\ ABO$, there are a large number of solution points that cannot satisfy (4). To make our search more efficient, we would like to reduce our search region as much as possible without sacrificing the optimization solution quality. Theorem 2 provides the basis for reducing the search region.

Theorem 2. *Given a set Γ_{IMF} of N tasks. If the necessary and sufficient condition for schedulability in Theorem 1 is satisfied for a synchronous task set Γ_{IMF} with $(D_{ii} = D_{im}, D_{im}, D_{if})$ for $i = 1, ..., N$, then the same condition is satisfied for a synchronous task set Γ'_{IMF} with $(D'_{ii}, D'_{im}, D_{if})$, where $D'_{ii} = D'_{im} \geq D_{im}$ for $i = 1, ..., N$.*

Applying Theorem 2 to the search region depicted in Figure 1, one can readily see that point M' on the segment MI leads to a schedulable solution if point M leads to a schedulable solution. Since D_{im} corresponding to M' is larger than that of M, M' is a more desirable solution than M as it leads to a smaller D_{if}. Based on this observation, we can reduce the search region by 1/2 by replacing constraints (6)-(7) in the optimization problem by the following:

$$C_{if} \leq D_{if} \leq D_i - D_{im}, \quad (10)$$

$$\frac{D_i}{2} \leq D_{im} \leq D_i - C_{if}, \quad (11)$$

If task τ_i is not decomposable, constraint (11) is replaced by

$$D_{im} = 0. \quad (12)$$

The optimization problem defined by (3) together with (4), (5), (10), (11) and (12) is similar to the problem studied in [10]. However, the simplified sufficient condition adopted by [10] either fails to find a solution or finds a very pessimistic solution for task sets with high utilization. It can also take more iterations to reach convergence. We have developed a better heuristic to avoid such problems. Our heuristic first replaces constraint (4) by two related constraints

$$\sum_{i=1}^{N}[(\lfloor\frac{L-D_{ii}}{P_i}\rfloor+1)\cdot C_{ii} + (\lfloor\frac{L-D_{im}}{P_i}\rfloor+1)\cdot C_{im}$$

$$+(\lfloor\frac{L-D_{if}}{P_i}\rfloor+1)\cdot C_{if}] \leq L, \forall L \in K \cdot P_i + D_i < L_{ip}, \quad (13)$$

$$\sum_{i=1}^{N}[(\frac{L-D_{ii}}{P_i}+1)\cdot C_{ii} + (\frac{L-D_{im}}{P_i}+1)\cdot C_{im} + (\frac{L-D_{if}}{P_i}+1)\cdot C_{if}] \leq L, \forall L = L_{ip}, \quad (14)$$

where L_{ip} and K are as defined in Theorem 1. It is easy to see that (4) is equivalent to (13) and (14). By leveraging the KKT Theorem [15], Theorem 3 defines the set of optimized final task deadlines for any fixed values of $L = L_{ip}$ and mandatory task deadlines. The schedulability of an optimized task set is checked at any scheduling point $L < L_{ip}$ by Theorem 1.

Theorem 3. *Given the constrained optimization problem as specified in (3), (5), (10), (14), for fixed values of $L = L_{ip}$ and mandatory task deadlines $D_{im}, \forall i$. Let*

$$\tilde{D} = \sum_{i=1}^{N} L \cdot U_i - \sum_{i=1}^{N} D_{im} \cdot (U_{ii} + U_{im}) + \sum_{i=1}^{N} C_i - L - \sum_{D_{if} \neq D_{ifmax}} D_{ifmin} U_{if} - \sum_{D_{if}=D_{ifmax}} D_{ifmax} U_{if}, \quad (15)$$

$$\tilde{S} = \sum_{D_{if} \neq D_{ifmax}} \frac{U_{if}^2}{w_{if}}, \quad (16)$$

A solution, D_{if}^*, is optimal, if and only if

$$D_{if}^* = \frac{\tilde{D} U_{if}}{\tilde{S} w_{if}} + D_{ifmin}, \quad (17)$$

where $U_i = \frac{C_i}{P_i}$, $U_{ii} = \frac{C_{ii}}{P_i}$, $U_{im} = \frac{C_{im}}{P_i}$, $U_{if} = \frac{C_{if}}{P_i}$, $D_{ifmin} = C_{if}$ and $D_{ifmax} = D_i - D_{im}$.

Based on Theorem 3, our heuristic solves the optimization problem as follows. For an initial solution $(D_{ii} = D_{im}, D_{if})$, the value of L is computed, and an updated D_{if} is obtained by applying Theorem 3. Then, D_{im} is updated by using $D_{im} = D_i - D_{if}$ which satisfies (10). The updated D_{im} and D_{if} are used to find a new L. This process is repeated until the algorithm converges. We refer to this heuristic as ADVR.

As we have seen from the motivational example, dynamic workload changes could cause larger delay variations if the original task/subtask deadlines were used. It is desirable to deploy an on-line adaptive framework to adjust task/subtask deadlines when workloads change significantly. The key to such an adaptive framework is an efficient method of solving the optimization problem posed earlier. Based on experimental results, our heuristic, ADVR, seems to satisfy such a requirement. Hence, we propose an adaptive framework built around ADVR. The framework is similar to the one in [8] and is shown in Figure 2.

An on-line monitoring mechanism in Kernel measures the mean execution time $\hat{c}_i$ and the maximum execution time $\hat{C}_i$, and transfers the data to Execution Time Estimator. Execution Time Estimator estimates and provides the approximate execution time Q_i for Trigger. Meanwhile, Plant also reports its error and task period P_i to Trigger. When the error and the changes of Q_i and P_i reach some thresholds, Trigger will signal ADVR algorithm to recompute the deadlines and send the results to Kernel again. With these new results, Kernel adjusts Plant so as to reduce delay variations.

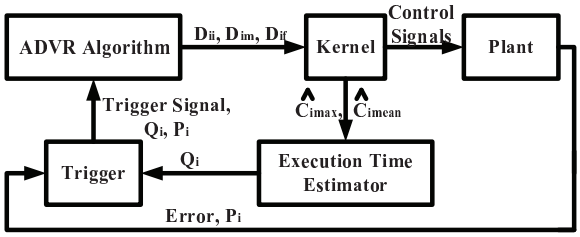

Fig. 2: Event-driven based arch. for deadline adaptation.

IV. Summary and Future Work

We have presented a new approach to reduce delay variations of control tasks. The approach formulates the delay variation reduction problem as an optimization problem that can effectively handles both decomposable and non-decomposable tasks. Based on several key observations, we devised an efficient heuristic to solve the optimization problem. The efficiency of the heuristic leads to an adaptive framework that can dynamically readjust task/subtask deadlines to keep delay variations small in the presence of environment perturbations. As future work, we will implement and evaluate our approach in a real-time operating system to control an actual application. Besides, our approach will be extended to adapt to various perturbations in physical environment.

References

[1] P. Albertos, A. Crespo, I. Ripoll, M. Valles, and P. Balbastre, "Rt control scheduling to reduce control performance degrading," in *ICDC '00*.

[2] P. Balbastre, I. Ripoll, and A. Crespo, "Control tasks delay reduction under static and dynamic scheduling policies," in *RTCSA '00*.

[3] ——, "Optimal deadline assignment for periodic real-time tasks in dynamic priority systems," in *ECRTS '06*.

[4] ——, "Minimum deadline calculation for periodic real-time tasks in dynamic priority systems," *IEEE Trans. Comput.*, vol. 57, no. 1, pp. 96–109, 2008.

[5] P. Balbastre, I. Ripoll, J. Vidal, and A. Crespo, "A task model to reduce control delays," *Real-Time Syst.*, vol. 27, no. 3, 2004.

[6] S. K. Baruah, L. E. Rosier, and R. R. Howell, "Algorithms and complexity concerning the preemptive scheduling of periodic, real-time tasks on one processor," *Real-Time Syst.*, vol. 2, no. 4, 1990.

[7] E. Bini and G. Buttazzo, "The space of edf deadlines: the exact region and a convex approximation," *Real-Time Syst.*, vol. 41, no. 1, pp. 27–51, 2009.

[8] G. Buttazzo and L. Abeni, "Adaptive workload management through elastic scheduling," *Real-Time Syst.*, vol. 23, no. 1/2, pp. 7–24, 2002.

[9] G. C. Buttazzo, "Hard real-time computing systems: Predictable scheduling algorithms and applications." Springer, 2005.

[10] T. T. Chantem, X. S. Hu, and M. D. Lemmon, "Generalized elastic scheduling for real-time tasks," *IEEE Trans. Comput.*, vol. 58, no. 4, pp. 480–495, 2009.

[11] A. Crespo, I. Ripoll, and P. Albertos, "Reducing delays in rt control: The control action interval, decision and control," in *IFAC '99*.

[12] H. Hoang, G. Buttazzo, M. Jonsson, and S. Karlsson, "Computing the minimum edf feasible deadline in periodic systems," in *RTCSA '06*.

[13] T. Kim, H. Shin, and N. Chang, "Deadline assignment to reduce output jitter of real-time tasks," in *IFAC '00*.

[14] M. Lluesma, A. Cervin, P. Balbastre, I. Ripoll, and A. Crespo, "Jitter evaluation of real-time control systems," in *RTCSA '06*.

[15] S. G. Nash and A. Sofer, "Linear and nonlinear programming." McGraw-Hill, 1996.