Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Optimal Control of Multi-layer Discrete Event Systems with Real-Time Constraint Guarantees *

Jianfeng Mao Christos G. Cassandras

Dept. of Manufacturing Engineering and Center for Information and Systems Engineering, Boston University, Brookline, MA 02446, USA (email: jfmao@bu.edu, cgc@bu.edu)

Abstract: We consider Discrete Event Systems (DES) involving tasks with dependability requirements in the form of real-time constraints. We seek to control their processing times so as to satisfy these constraints while also minimizing a given cost function. When tasks are processed by a single resource, it has been shown that there are structural properties of the optimal state trajectory for this problem that lead to a very efficient *Critical Task Decomposition Algorithm* (CTDA). For a DES with multiple resources, we consider a multi-layer network where each layer contains multiple nodes, each node may have multiple inputs and multiple outputs, and tasks are processed so that the real-time constraints apply on an end-to-end basis. Extending earlier results (where each layer contained a single node), we derive structural properties of the optimal solution that lead to the idea of introducing "virtual"deadlines at each node (except for the last layer) and decouple nodes so that the CTDA for single-node problems can be used. We prove that an appropriately constructed sequence of solutions of these simpler problems converges to the global optimum of the original problem and hence obtain an efficient scalable Multi-Layer Virtual Deadline Algorithm (MLVDA). We illustrate the efficiency of the MLVDA through numerical examples.

1. INTRODUCTION

A large class of Discrete Event Systems (DES) involves the control of resources allocated to tasks according to certain operating specifications. In designing *dependable* DES, a key dependability requirement is that tasks must be executed within given *real-time constraints*. However, such dependability comes at a cost, giving rise to a fundamental dependability-efficiency trade-off. Examples arise in manufacturing systems, where the operating speed of a machine can be controlled to trade off between energy costs and requirements on timely job completion (Pepyne and Cassandras [2000]), or in wireless networks where severe battery limitations call for new techniques aimed at minimizing energy consumption while still providing performance guarantees (Miao and Cassandras [2006], Gamal et al. [2002]). Thus, if the cost of guaranteeing real-time constraints is so high as to exhaust the batteries of a wireless system, then this system is obviously no longer dependable. This trade-off leads to a class of optimization problems where a cost function must be minimized subject to the usual DES dynamics in max-plus form for task completion times along with real-time (dependability) constraints imposed on these completion times.

A single-resource DES can be modelled as a single-input single-output (SISO) single-server queueing system as in Fig. 1, whose dynamics are given by the well-known max-plus equation

$$x_i = \max(x_{i-1}, a_i) + s_i(u_i) \tag{1}$$

where a_i is the arrival time of task $i = 1, 2, \dots$, x_i is the time when task i completes service, and $s_i(u_i)$ is its service time which is controllable through its processing rate u_i. Real-time constraints imposed on tasks are represented through $x_i \le d_i$ where d_i is a given "deadline" associated with task i. In order to meet such constraints, one typically has to incur a higher cost associated with s_i (since the server must operate at higher speed). This class of problems has been extensively studied, mostly in the computer science literature: preemptive tasks are considered, for example, in Yao et al. [1995], Aydin et al. [May 2004], nonpreemptive periodic tasks in Jeffay et al. [1991], and nonpreemptive aperiodic tasks in Gamal et al. [2002], Miao and Cassandras [2006], Mao et al. [June 2007]. The latter case is of particular interest in wireless communications where nonpreemptive scheduling is necessary to execute aperiodic packet transmission tasks which also happen to be highly energy-intensive. In a broader context, we are interested in studying optimization problems of the general form

$$\min_{u_1,\dots,u_N} \sum_{i=1}^{N} \theta_i(u_i) \tag{2}$$
$$\text{s.t.} \quad x_i = \max(x_{i-1}, a_i) + s_i(u_i) \le d_i, \qquad i = 1, ..., N$$

where $\theta_i(u_i)$ is the cost of operating the resource at a high enough rate to complete task i within s_i and guarantee that $x_i \le d_i$. In general, the controls $u_1, ..., u_N$ may be time-varying. However, as shown in Miao and Cassandras [Sep. 2005], when $\theta_i(\cdot)$ is monotonically increasing and convex and all a_i, d_i are known, then the optimal control of each task is constant during the processing time $s_i(u_i)$. We

* Supported in part by NSF under Grants DMI-0330171 and EFRI-0735974, by AFOSR under grants FA9550-07-1-0213 and FA9550-07-1-0361, and by DOE under grant DE-FG52-06NA27490.

will consider such cases and also assume that $s_i(u_i) \geq 0$ is monotonically decreasing in u_i for all $i = 1, ..., N$.

Problem (2) is generally a hard nonlinear optimization problem, complicated by the inequality constraints $x_i \leq d_i$ and the nondifferentiable max operator involved. Although the max operator can be removed by introducing auxiliary variables w_i, $i = 1, ..., N$, and adding the constraints $x_i = w_i + s_i(u_i)$, $w_i \geq x_{i-1}$, $w_i \geq a_i$, we note that this makes the problem even more inefficient to solve since it doubles its dimensionality and also introduces $2N$ inequality constraints. Despite these difficulties, it has been shown in Mao et al. [June 2007] that the *Critical Task Decomposition Algorithm* (CTDA) provides a highly efficient solution procedure which reduces a complex optimization problem to a simple procedure for identifying a set of "critical tasks". Numerical examples show that standard nonlinear programming software fails to produce solutions to problems for $N > 400$. In contrast, CTDA is scalable in N and gives exact solutions extremely fast (compared to specialized convex programming solvers). Speed and scalability are particularly crucial for applications where small, inexpensive, often wireless devices are required to perform on-line computations with minimal on-board resources.

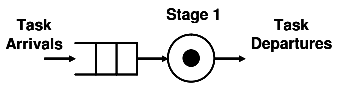

Fig. 1. A single-resource system

In the case of a multi-resource DES where tasks must be sequentially processed over a number of stages as in Fig. 2, each stage is characterized by dynamics of the max-plus form coupled to those of its neighboring stages (1), i.e., tasks at the first stage satisfy

$$x_{i,1} = \max(x_{i-1,1}, a_i) + s_{i,1}(u_{i,1}) \quad (3)$$

and at the following stages $j = 2, ..., M$:

$$x_{i,j} = \max(x_{i-1,j}, x_{i,j-1}) + s_{i,j}(u_{i,j}),\ l = 2, ..., M \quad (4)$$

In addition, the real-time constraints apply on an end-to-end basis, i.e., only tasks at the last stage satisfy the constraints $x_{i,M} \leq d_i$. This problem is much more complicated and cannot be dealt with by merely extending the CTDA because the decomposition properties characterizing an optimal sample path of (2) no longer hold and the coupling in (4) significantly complicates any solution methodology. Incidentally, the same complications arise even in the absence of real-time constraints: extending such single-stage problems solved in Cho et al. [2001] even to two stages becomes significantly more difficult Gokbayrak and Cassandras [2000], Cassandras et al. [1999]. Nonetheless, exploiting structural properties of the optimal solution, we have developed a highly efficient *Virtual Deadline Algorithm* (VDA) Mao and Cassandras [2006] to solve this problem. The main idea is to introduce a "virtual" deadline at each stage $1, ..., M-1$, so that the M-stage problem is replaced by $\sum_{l=1}^{M} V_l$ single-stage problems of the form (2), which we know can be very efficiently solved through the CTDA in Mao et al. [June 2007].

In this paper, we study a more general *multi-layer* network environment as shown in Fig. 3, where each resource is represented by a node which may have multiple inputs and multiple outputs (queues are included in the nodes but are omitted from the figure). Examples arise in assembly and/or disassembly manufacturing systems, or in some clustering-based protocols for wireless sensor networks, such as LEACH (Heinzelman et al. [2002]). In such multi-layer systems, each node may have multiple outputs (e.g., disassembled parts, packet copies) each one routed to a specified downstream node. A node may also have multiple inputs so that the time when the ith task can be initiated is a function of the departure times of the ith task at all its predecessor nodes; in particular, we consider this function to be the the maximum value of these departure times, which corresponds to applications such as assembly operations in manufacturing systems or computations requiring multiple data in computer systems. The main contribution of the paper is the construction of a sequence of solutions to much simpler single-resource (i.e., single-node) problems, a proof that this sequence converges to to the global optimum of the multi-layer problem, and the development of an efficient algorithm (which results from this analysis) to solve this problem.

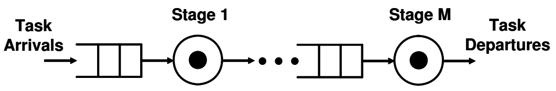

Fig. 2. A multi-resource system with M sequential stages

The paper is organized as follows. In Section 2, we formulate the M-layer problem with strict end-to-end real-time constraints. In Section 3, we establish two structural properties of the optimal solution extending results from multi-stage systems as in Fig. 2. These lead, in Section 4, to the construction of a single-node problem solution sequence and a proof that this sequence converges to the global optimum of the multi-layer problem. We present a multi-layer VDA in Section 5, followed by numerical examples in Section 6, and conclude with Section 7.

2. MULTI-LAYER PROBLEM FORMULATION

We consider a multi-layer DES, as illustrated in Fig 3, where there are M layers with V_l nodes in the l-th layer for $l = 1, ..., M$. Let (l, n) denote the n-th node in the l-th layer. Node (l, n) can only receive tasks from the (l-1)th layer and send tasks to the (l+1)th layer; communication among nodes in the same layer is not allowed. There is a sequence of N tasks arriving at known times $0 \leq a_{1,n} \leq \cdots \leq a_{N,n}$ at each node $(1, n)$ in the first layer with known hard end-to-end deadlines $d_{1,n}, \ldots, d_{N,n}$ for each node (M, n) in the last layer.

Let $P_{l,n}$ denote the set of predecessors of node (l, n). For example, in Fig 3, $P_{2,1} = \{(1,1), (1,2), (1,V_1)\}$. Since the predecessors of node (l, n) must be in the (l-1)th layer, we can simplify $P_{l,n}$ by removing the layer index, e.g., rewrite $P_{2,1}$ as $P_{2,1} = \{1, 2, V_1\}$. Similarly, we define $B_{l,n}$ as the set of successors of node (l, n); for example, $B_{1,1} = \{1, 2\}$.

The dynamics of the multi-layer system are significantly different from the single and multi-stage systems in Figs. 1, 2. In the latter, we are dealing with single-input and single-output nodes, while the nodes in the multi-layer system generally have multiple inputs and multiple outputs. In the multi-stage system, each node only has one predecessor

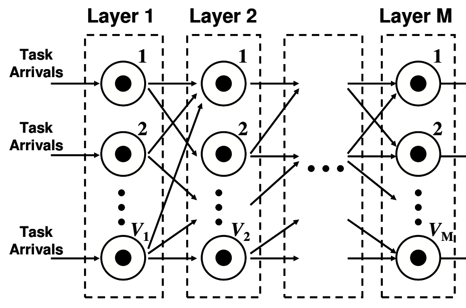

Fig. 3. A multi-layer system (each node includes queues for incoming tasks which are not shown)

so that the time when a task becomes available to that node is simply the departure time of the previous stage. In the multi-layer system we consider (which is not an arbitrary tree structure), each node may have more than one predecessor so that the time when a task becomes available for processing at some node (often referred to as its "release time") is a function of the departure times of this task at its predecessors. As mentioned in the introduction, we consider this function to be the maximum value of the departure times of the task at predecessor nodes. Let $s_{i,l,n}$ and $x_{i,l,n}$ denote task i's service time and departure time at node (l,n) respectively and define $S_{l,n}=[s_{i,l,n}]_{\forall i}$, $S_l=[S_{l,n}]_{\forall n}$, $S=[S_l]_{\forall l}$, $X_{l,n}=[x_{i,l,n}]_{\forall i}$, $X_l=[X_{l,n}]_{\forall n}$, $X=[X_l]_{\forall l}$, $A_n=[a_{i,n}]_{\forall i}$, $A=[A_n]_{\forall n}$, $D_n = [d_{i,n}]_{\forall i}$ and $D=[D_n]_{\forall n}$, where the notation $[\cdot]_{\forall i}$ is shorthand for a vector or matrix whose entries correspond to all possible values of i. In what follows, inequalities and max operators involving vectors or matrices should be understood to apply componentwise. For notational consistency, let $X_0=A$ and $P_{1,n}=\{n\}$ for $n=1,...,V_1$. Then, the dynamics of the multi-layer system can be written as

$$x_{i,l,n} = \max\left(x_{i-1,l,n}, \max_{h\in P_{l,n}}(x_{i,l-1,h})\right) + s_{i,l,n}$$

To simplify notation, we define

$$\hat{X}_{i,l,n} = \left[x_{i-1,l,n},\ [x_{i,l-1,h}]_{\forall h\in P_{l,n}}\right] \qquad (5)$$

Assuming $s_{i,l,n}(u_{i,l,n})$ for all i,l,n are known monotonically decreasing functions of $u_{i,l,n}$, we will concentrate on controlling directly $s_{i,l,n}$ ($u_{i,l,n}$ can then be recovered) for all i,l,n. We can formulate the multi-layer problem as:

$$\begin{aligned}&\min_S\left\{J(S) = \sum_{l=1}^M \sum_{n=1}^{V_l} \sum_{i=1}^N \theta_{i,l,n}(s_{i,l,n})\right\}\\ &\text{s.t.}\quad x_{i,l,n} = \max\left(\hat{X}_{i,l,n}\right) + s_{i,l,n},\ \forall i,l,n;\\ &\qquad X_M \le D,\ S\ge 0;\quad x_{0,l,n} = -\infty,\ \forall l,n.\end{aligned} \qquad (6)$$

The functions $\theta_{i,l,n}(\cdot)$ are assumed to be continuously differentiable, strictly convex and monotonically decreasing, which is consistent with most applications of interest. The problem in (6) reflects the trade-off between guaranteeing the dependability requirements expressed through $x_M \le D$ and the costs $\theta_{i,l,n}(s_{i,l,n})$ of operating nodes at rates resulting in processing times given by $s_{i,l,n}$. Intuitively, we seek the lowest possible processing rates (hence, low cost) that can guarantee the real-time constraints for all tasks.

3. OPTIMALITY PROPERTIES

3.1 Virtual Deadline Property

The first structural property we identify is one leading to a partial decoupling of the $\sum_{l=1}^M V_l$ nodes by introducing a "virtual" deadline for tasks at all nodes in layers $l < M$ and show that we can replace (6) by a set of much simpler problems with a weaker form of coupling among nodes.

We transform (6) into an equivalent problem below by setting the control variables in S to be the entries of X and incorporating the dynamics into the objective function. In what follows, we will omit the subscripts i,l,n from the function $\theta_{i,l,n}(\cdot)$ only to simplify the unavoidable notational burden necessitated by indexing tasks, layers, and nodes. It will be seen that this causes no loss of generality, as all subsequent results do not depend on any differences among cost functions associated with tasks or nodes, as long as all such functions remain convex and monotonic. The transformed problem (6) becomes:

$$\begin{aligned}&\min_X\left\{J(X) = \sum_{l=1}^M \sum_{n=1}^{V_l} \sum_{i=1}^N \theta\left(x_{i,l,n} - \max(\hat{X}_{i,l,n})\right)\right\}\\ &\text{s.t.}\quad X_M \le D;\quad x_{0,l,n} = -\infty,\ \forall l,n;\\ &\qquad x_{i,l,n} - \max(\hat{X}_{i,l,n}) \ge 0,\ \forall i,l,n.\end{aligned} \qquad (7)$$

Its optimal solution will henceforth be denoted by X^*.

We can see that all nodes are strongly coupled because of the end-to-end real time constraints. Now, imagine that there exist virtual deadlines for all tasks at each node (l,n) for $l < M$, denoted by $\Delta_{l,n} = [\delta_{i,l,n}]_{\forall i}$, and that every node can independently optimize its control to meet these virtual deadlines. Then, the multi-layer problem (7) would be reduced to $\sum_{l=1}^M V_l$ single-node problems of the form studied in Mao et al. [June 2007], where the *release time vector* at node (l,n), denoted by $R_{l,n} = [r_{i,l,n}]_{\forall i}$, can be written as $R_{l,n} = \max_{h\in P_{l,n}}\left(X_{l-1,h}\right)$. Define

$$L\left(X_{l,n}|R_{l,n}\right) = \sum_{i=1}^N \theta\left(x_{i,l,n} - \max(x_{i-1,l,n}, r_{i,l,n})\right) \quad (8)$$

Let $\Delta_{M,n} = D_n$, $n = 1,...,V_M$ for notational consistency and formulate a single-node problem for each (l,n):

$$\min_{X_{l,n}\in\Phi(R_{l,n},\Delta_{l,n})} L\left(X_{l,n}|R_{l,n}\right) \qquad (9)$$

where the feasible space $\Phi(R_{l,n},\Delta_{l,n})$ is defined as

$$\begin{aligned}\Phi(R_{l,n},\Delta_{l,n}) = \{&X_{l,n}|X_{l,n}\le\Delta_{l,n},\dots\\ &x_{i,l,n} - \max(x_{i-1,l,n}, r_{i,l,n}) \ge 0, \forall i.\}\end{aligned} \qquad (10)$$

Thus, in (9) we fix the vector $R_{l,n}$ with all release times at node (l,n) and control $X_{l,n}$. Since these single-node problems can be efficiently solved by the CTDA, solving $\sum_{l=1}^M V_l$ separate single-node problems is much easier than solving the multi-layer problem (7). If we can obtain the optimal solution of (7) by solving these single-node problems above, then the complexity of solving (7) will be greatly reduced. We show that this may be possible through Theorem 1. (All proofs in this paper are omitted; full versions can be found in Mao and Cassandras [2007].)

Theorem 1. Let X^* denote the optimal solution of Problem (7) and $X_0^* = A$. If

$$R_{l,n}^* = \max_{h\in P_{l,n}}\left(X_{l-1,h}^*\right),\ \forall i,l,n \qquad (11)$$

$$\Delta_{l,n}^* = X_{l,n}^*\mathbf{1}_{[l<M]} + D_l\mathbf{1}_{[l=M]},\ \forall l,n \qquad (12)$$

then $X_{l,n}^* = \arg\min_{X_{l,n}\in\Phi(R_{l,n}^*,\Delta_{l,n}^*)} L(X_{l,n}|R_{l,n}^*), \forall l,n.$

Based on Theorem 1, it is possible that the whole multi-layer system reaches optimality when each node reaches its own optimality by setting virtual deadlines to $\Delta_{l,n} = X_{l,n}^*$. However, for arbitrary $\Delta_{l,n}$, the optimality for each node does not correspond to the optimality of the whole multi-layer system. Thus, this decomposition applies only at the optimal point X^*, making the theorem of little apparent use. However, we will see that combining this result with an additional property discussed in the next section leads to an efficient iterative process which still reduces the solution of the original problem to solving $\sum_{l=1}^{M} V_l$ partially coupled single-node problems.

3.2 Q-Chain Property

In this section, we will revisit what we refer to as the *Q-Chain Property*, introduced in Mao and Cassandras [2006], which can decompose the multi-layer problem into a sequence of partially coupled problems, referred to as *Q-Problems*. This property leads to the main result of this section, Theorem 2.

To fully understand the Q-Chain property, we begin with the simple single-node problem, a special case of the multi-layer system with $M = 1$ and $V_1 = 1$, where $X \equiv [x_i]_{i=1,...,N}$ and x_i is the departure time of task i:

$$\begin{aligned} &\min\nolimits_X J(X) = \sum\nolimits_{i=1}^{N} \theta\big(x_i - \max(a_i, x_{i-1})\big) \\ &\text{s.t. } \; x_i - \max(a_i, x_{i-1}) \geq 0, \;\; i = 1, ..., N; \quad X \leq D \end{aligned} \tag{13}$$

Introduce the Q-Problem for this simple scalar case:

$$\begin{aligned} \min\nolimits_{x_i} \big\{ & Q(x_i|x_{i-1}, x_{i+1}) \equiv \theta\,(x_i - \max(a_i, x_{i-1})) \\ & + \theta\,(x_{i+1} - \max(a_{i+1}, x_i)) \,\big\} \\ \text{s.t. } \; & x_i - \max(a_i, x_{i-1}) \geq 0, \\ & x_{i+1} - \max(a_{i+1}, x_i) \geq 0, \;\; x_i \leq d_i. \end{aligned}$$

Observe that in the Q-Problem above x_i is controllable while x_{i-1} and x_{i+1} are treated as fixed. There is a total of N such Q-Problems. Each one is a small piece of the single-node problem (13) and is only coupled to its two neighboring Q-Problems, $Q(x_{i-1}|x_{i-2}, x_i)$ and $Q(x_{i+1}|x_i, x_{i+2})$. We refer to all these Q-Problems collectively for $i = 1, \ldots, N$ as the "*Q-Chain*".

Let $\mathcal{D}_Y J(X)$ denote the directional derivative of $J(X)$ at X along a feasible direction $Y = [y_i]_{i=1,...,N}$ and let $\mathcal{D}_{y_i} Q(x_i|x_{i-1}, x_{i+1})$ denote the directional derivative of $Q(x_i|x_{i-1}, x_{i+1})$ at x_i along a feasible direction y_i. It can be easily verified that

$$\mathcal{D}_Y J(X) = \sum\nolimits_{i=1}^{N} \mathcal{D}_{y_i} Q(x_i|x_{i-1}, x_{i+1}) \tag{14}$$

Optimality for a convex programming problem (see Theorem 4.3.2 in Giorgi et al. [2004]) such as (13) means that for any feasible direction Y at X^*:

$$\mathcal{D}_Y J(X^*) \geq 0$$

Suppose we can obtain solutions to all Q-Problems and define a vector $X^* = [x_i^*]_{\forall i}$ such that

$$\mathcal{D}_{y_i} Q(x_i^*|x_{i-1}^*, x_{i+1}^*) \geq 0, \quad \text{for all } \; i = 1, ..., N$$

Then, combining this condition with (14), we have

$$\mathcal{D}_Y J(X^*) = \sum\nolimits_{i=1}^{N} \mathcal{D}_{y_i} Q(x_i^*|x_{i-1}^*, x_{i+1}^*) \geq 0$$

which implies that X^* is also the optimal solution of the single-node problem (13). Each Q-Problem above is a scalar problem which is much easier than solving (13). This relationship provides an opportunity to obtain the optimal solution of a large-scale problem by solving a set of much simpler Q-Problems.

Based on this analysis, we see that the key to establishing the equivalence between the optimality of Q-Problems and (13) is the summation form in (14) satisfied by the directional derivative $\mathcal{D}_Y J(X)$. If we can similarly express the directional derivative of the multi-layer problem (7) as the summation of the directional derivatives of cost functions for some properly defined Q-Problems, then we can establish a similar equivalence of the multi-layer problem and a set of simpler Q-Problems. A direct extension of the Q-Problem defined above to the multi-stage system of Fig. 2 (a special case of our multi-layer problem) is to vary $[x_{i,j}]_{\forall j}$ by fixing $[x_{i-1,j}]_{\forall j}$ and $[x_{i+1,j}]_{\forall j}$ where j is the stage index. Unfortunately, this can be easily shown to violate a relationship similar to (14). This failure is due to the presence of the function $\max(x_{i-1,j}, x_{i,j-1})$ which introduces a coupling between $x_{i-1,j}$ and $x_{i,j-1}$ for any i, j. To satisfy a summation form as in (14), we have to include both $x_{i-1,j}$ and $x_{i,j-1}$ as controllable variables when defining the Q-Problem (see Mao and Cassandras [2006]). With this motivation in mind, in the multi-layer case we define $\tilde{X}_i = [x_{i-l,l,n}]_{\forall l,n}$ and, recalling $\hat{X}$ in (5),

$$\begin{aligned} Q\big(\tilde{X}_i \big|\, \tilde{X}_{i-1}, \tilde{X}_{i+1}\big) = \sum_{l=1}^{M} \sum_{n=1}^{V_l} \Big(& \theta\big(x_{i-l,l,n} - \max(\hat{X}_{i-l,l,n})\big) \\ & + \theta\big(x_{i-l+1,l,n} - \max(\hat{X}_{i-l+1,l,n})\big)\Big) \end{aligned}$$

We then formulate the following Q-Problem for $i = 2, ..., M + N$:

$$\min\nolimits_{\tilde{X}_i \in \Psi(\tilde{X}_{i-1}, \tilde{X}_{i+1})} Q\big(\tilde{X}_i \big| \tilde{X}_{i-1}, \tilde{X}_{i+1}\big) \tag{15}$$

where the feasible set $\Psi(\tilde{X}_{i-1}, \tilde{X}_{i+1})$ is defined as

$$\begin{aligned} \Psi(\tilde{X}_{i-1}, \tilde{X}_{i+1}) = \{ & x_{i-M,M,n} \leq d_{i-M,n}, \; \forall n; \ldots \\ & x_{i-l+1,l,n} - \max(\hat{X}_{i-l+1,l,n}) \geq 0, \; \ldots \\ & x_{i-l,l,n} - \max(\hat{X}_{i-l,l,n}) \geq 0, \; \forall l, n. \} \end{aligned}$$

Note that there exist some $x_{i,l,n}$, $d_{i,n}$ and $a_{i,n}$ such that $i < 1$ or $i > N$ in the definition of the Q-Problem (15), which are "dummy variables" because $x_{i,l,n}$, $d_{i,n}$ and $a_{i,n}$ are only defined for $i = 1, ..., N$. In order to eliminate the influence of these dummy variables, we set $d_{i,n} = \min(a_{1,1}, ..., a_{1,V_1})$ for $i < 1$ and let $x_{i,l,n}$ be arbitrary constants smaller than $\min(a_{1,1}, ..., a_{1,V_1})$ for all i such that $i < 1$; that is, we force all "dummy" tasks before task 1 to leave before the smallest arrival time of task 1 so as to decouple them from $1, ..., N$. Similarly, we set $a_{i,n}$ and $x_{i,l,n}$ to be arbitrary constants larger than $\max(d_{N,1}, ..., d_{N,V_M})$ for all $i > N$, that is, we force all tasks after N to arrive after the largest deadline of task N so as to decouple them from tasks $1, ..., N$.

In what follows, Lemma 1 shows that the definition of the Q-Problem in (15) satisfies a summation form condition similar to (14) established for single-node systems. Let $Y = [y_{i,l,n}]_{\forall i,l,n}$ and $\tilde{Y}_i = [y_{i-l,l,n}]_{\forall l,n}$ such that $y_{i,l,n} = 0$ for any $i < 1$ or $i > N$ or $l < 1$.

Lemma 1.

$$\mathcal{D}_Y J(X) = \sum\nolimits_{i=2}^{M+M} \mathcal{D}_{\tilde{Y}_i} Q(\tilde{X}_i|\tilde{X}_{i-1}, \tilde{X}_{i+1}).$$

Lemma 1 can only guarantee local optimality for the multi-layer problem (7). To establish global optimality, we need to ensure the convexity of (7) and (15), as shown below.

Lemma 2. The multi-layer problem (7) is strictly convex in X and the Q-Problem (15) is strictly convex in $\tilde{X}_i$.

Using Lemmas 1 and 2, we can finally obtain the following necessary and sufficient condition for global optimality.

Theorem 2. Let $X^* = [x^*_{i,l,n}]_{\forall i,l,n}$ and $\tilde{X}^*_i = [x^*_{i-l,l,n}]_{\forall l,n}$. X^* is the unique global optimum of the multi-layer problem (7) if and only if

$$\tilde{X}^*_i = \arg\min_{\tilde{X}_i} Q_i\big(\tilde{X}_i | \tilde{X}^*_{i-1}, \tilde{X}^*_{i+1}\big), \text{for } i = 2, ..., N+M.$$

Theorem 2 provides a way to determine the optimality of the multi-layer problem (7) by solving a set of $\sum_{l=1}^{M} V_l$-dimensional convex optimization problems. The final remaining question is how to exploit this property in order to efficiently determine X^*.

4. CONVERGENCE ANALYSIS OF SINGLE-NODE SOLUTION SEQUENCES

As shown in the previous section, we can partially decompose the original multi-layer problem into single-node problems or Q-Problems. Although the optimal solution X^* of the multi-layer problem still cannot be directly obtained by solving these two types of problems, together they can be utilized to solve the multi-layer problem through a sequence of single-node problem solutions. We will describe next how to construct such a sequence and prove that it monotonically converges to X^*.

Consider a sequence of single-node problems of the form (9) with solutions defined by

$$X^k_{l,n} = \arg\min_{X_{l,n} \in \Phi(R^k_{l,n}, \Delta^k_{l,n})} L(X_{l,n} | R^k_{l,n}), \quad \forall l, n \quad (16)$$

This gives rise to a sequence $\{X^k\}$, $k = 1, 2, \ldots$ with $X^k = [X^k_{l,n}]_{\forall l,n}$ and $X^k_{l,n}$ dependent on the virtual deadline vector $\Delta^k_{l,n}$. Let us initialize this vector so that

$$\Delta^1_M = D, \;\; \Delta^1_{l,n} = \min_{h \in B_{l,n}} \Delta^1_{l+1,h}, \; \forall\, l < M.$$

Define $\tilde{\Delta}^{k+1}_i = [\delta^{k+1}_{i-l,l,n}]_{\forall l,n}$ to be the solution of the Q-Problem and let $\tilde{X}^k_i = [x^k_{i-l,l,n}]_{\forall l,n}$, so that for all i

$$\tilde{\Delta}^{k+1}_i = \arg\min_{\tilde{X}_i \in \Psi(\tilde{X}^k_{i-1}, \tilde{X}^k_{i+1})} Q(\tilde{X}_i | \tilde{X}^k_{i-1}, \tilde{X}^k_{i+1}). \quad (17)$$

We construct $\Delta^{k+1} = [\Delta^{k+1}_{l,n}]_{\forall l,n}$, where $\Delta^{k+1}_{l,n} = [\delta^{k+1}_{i,l,n}]_{\forall i}$ is obtained from $[\delta^{k+1}_{i-l,l,n}]_{\forall l,n}$ for any given i, l, n. This process is illustrated in Fig. 4 where we see that X^k is the input to a collection of Q-Problems and Δ^{k+1} consists of the solutions of these problems, which are then input to single-node problems with virtual deadlines given by Δ^{k+1}.

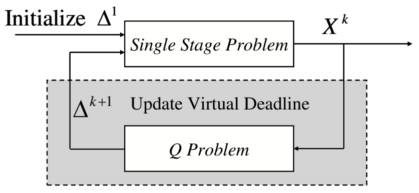

Fig. 4. Generating the sequence of X^k for $k = 1, 2, \ldots$

In the following, we show that $X^k \to X^*$ as $l \to \infty$. We begin with some auxiliary results, i.e., Lemma 3 and Lemma 4, which establish monotonicity properties satisfied by the solutions of the single-node virtual deadline problems and the solutions of the Q-Problems respectively.

Lemma 3. Let

$$X^1_{l,n} = \arg\min_{X_{l,n} \in \Phi(R^1_{l,n}, \Delta^1_{l,n})} L(X_{l,n} | R^1_{l,n}) \quad (18)$$

$$X^2_{l,n} = \arg\min_{X_{l,n} \in \Phi(R^2_{l,n}, \Delta^2_{l,n})} L(X_{l,n} | R^2_{l,n}). \quad (19)$$

If $R^1_{l,n} \le R^2_{l,n}$ and $\Delta^1_{l,n} \le \Delta^2_{l,n}$, then $X^1_{l,n} \le X^2_{l,n}$.

Lemma 4. Let

$$\tilde{X}^1_i = \arg\min_{\tilde{X}_i \in \Psi(\tilde{X}^1_{i-1}, \tilde{X}^1_{i+1})} Q(\tilde{X}_i | \tilde{X}^1_{i-1}, \tilde{X}^1_{i+1}) \quad (20)$$

$$\tilde{X}^2_i = \arg\min_{\tilde{X}_i \in \Psi(\tilde{X}^2_{i-1}, \tilde{X}^2_{i+1})} Q(\tilde{X}_i | \tilde{X}^2_{i-1}, \tilde{X}^2_{i+1}) \quad (21)$$

If, for any $i = 2, \ldots, N+M$, $\tilde{X}^1_{i-1} \le \tilde{X}^2_{i-1}$ and $\tilde{X}^1_{i+1} \le \tilde{X}^2_{i+1}$, then $\tilde{X}^1_i \le \tilde{X}^2_i$.

Before getting to the main convergence result regarding the sequence $\{X^k\}$, we establish one more monotonicity property which applies to the sequence $\{\Delta^k\}$, $k = 1, 2, \ldots$

Lemma 5. The sequence $\{\Delta^k\}$, $k = 1, 2, \ldots$, is monotonically nonincreasing.

Theorem 3. The sequence $\{X^k\}$, $k = 1, 2, \ldots$, is monotonically nonincreasing and $\lim_{k\to\infty} X^k = X^*$.

5. MULTI-LAYER VIRTUAL DEADLINE ALGORITHM

From above, the Multi-Layer Virtual Deadline Algorithm (MLVDA) in Table 1 is a direct implementation of the sequence construction in (16)-(17) also illustrated in Fig. 4. The MLVDA provides a computationally efficient way to obtain X^* by exploiting the fact that each single-node problem in Step 2 can be very efficiently solved with the CTDA in Mao et al. [June 2007] (or its generalized version GCTDA Miao and Cassandras [2006]), while solving each Q-Problem in Step3, an $\sum_{l=1}^{M} V_l$-dimensional convex optimization problem, is a relatively simple task. Theorem 3 guarantees that the MLVDA will converge to the global optimum for our original problem.

Table 1. MLVDA

Step 1:	$k = 1$, $\Delta^1_M = D$, $\Delta^1_{l,n} = \min_{h \in B_{l,n}} \Delta^1_{l+1,h}$ for $l < M$;
Step 2:	$X^k_{l,n} = \arg\min_{X_{l,n} \in \Phi(R^k_{l,n}, \Delta^k_{l,n})} L(X_{l,n} \mid R^k_{l,n}), \forall l, n$; where $R^k_{l,n} = \max_{h \in P_{l,n}} (X^k_{l-1,h}), \forall l, n$;
Step 3:	$\tilde{\Delta}^{k+1}_i = \arg\min_{\tilde{X}_i \in \Psi(\tilde{X}^k_{i-1}, \tilde{X}^k_{i+1})} Q(\tilde{X}_i \mid \tilde{X}^k_{i-1}, \tilde{X}^k_{i+1})$, $\forall i$;
Step 4:	if $\Vert X^{k+1} - X^k \Vert > \epsilon$, then $k = k + 1$, goto Step 2;
Step 5:	$X^* = X^k$.

6. NUMERICAL RESULTS

We apply the MLVDA to an assembly system in Fig. 5 (another simple example can also be found in Mao and Cassandras [2007]) and test the complexity of the MLVDA in terms of CPU time compared to CVX, an efficient convex programming solver (Grant et al. [2006]). In these tests, the MLVDA was programmed using Matlab 7.0 on an Intel Pentium4 3.06GHz, 1.0 GB RAM machine. We tested cases where N varied from 400 to 4000 in increments of 400. We randomly generated 10 samples for each N. For each case, we recorded the elapsed CPU time, finally averaging them to obtain the corresponding performance.

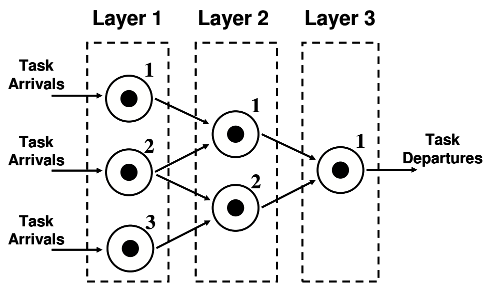

Fig. 5. A three-layer system

Figure 6 shows the average CPU time (in seconds) as a function of the number of tasks N, where solid line and dash line represent the results of CVX and MLVDA respectively. We observe that the MLVDA complexity scales with N while CVX does not possess this property because MLVDA only needs to solve $\sum_{l=1}^{M} V_l$-dimensional ($\sum_{l=1}^{M} V_l = 6$ in this example) convex optimization problem no matter what N is and the CTDA utilized in MLVDA also scales with N.

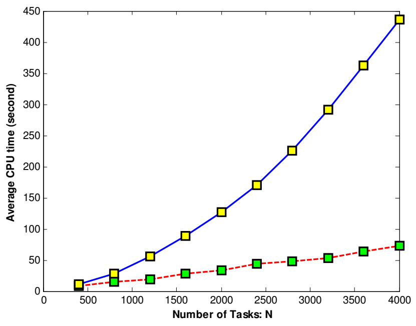

Fig. 6. MLVDA vs. CVX

7. CONCLUSIONS

We have considered DES consisting of multiple resources grouped in sequential layers so as to process tasks with dependability requirements in the form of end-to-end real-time constraints. The processing times of tasks at different resources are controlled so as to satisfy these constraints while also minimizing a given cost function. We have extended earlier work for the case where each layer contains a single node and derived structural properties of the solution of the constrained optimization problem of interest. Based on these properties, we have constructed a sequence of solutions of simpler problems and proved that this sequence converges to the global optimum of the original problem. This has led to an efficient scalable Multi-Layer Virtual Deadline Algorithm (MLVDA). This approach can further be extended to a network environment with a general tree structure, since we can always convert it to a multi-layer system by introducing appropriate "virtual nodes". Future work is targeting problems where the arrival times of tasks are not known in advance, in which case one must proceed by repeatedly solving the problem as new arrival information is obtained, by estimating future arrivals, or by relying on stochastic optimization techniques making use of distributional information regarding the arrival process.

REFERENCES

H. Aydin, R. Melhem, D. Mossé, and P. Mejia-Alvarez. Power-aware scheduling for periodic real-time tasks. *IEEE Trans. on Computers*, 53(5):584 – 600, May 2004.

C.G. Cassandras, Q. Liu, K. Gokbayrak, and D.L. Pepyne. Optimal control of a two-stage hybrid manufacturing system model. In *Proc. of the 38th IEEE Conf. on Decision and Control*, pages 450–455, 1999.

Y.C. Cho, C.G. Cassandras, and D.L. Pepyne. Forward decomposition algorithms for optimal control of a class of hybrid systems. *Intl. J. of Robust and Nonlinear Control*, 11(5):497–513, 2001.

A. E. Gamal, C. Nair, B. Prabhakar, Elif Uysal-Biyikoglu, and S. Zahedi. Energy-efficient scheduling of packet transmissions over wireless networks. In *Proc. of IEEE INFOCOM*, volume 3, 23-27, pages 1773–1782, 2002.

C. Giorgi, A. Guerraggio, and J. Thierfelder. *Mathematics of Optimization: Smooth and Nonsmooth Case.* Elserier, 2004.

K. Gokbayrak and C.G. Cassandras. Constrained optimal control for multistage hybrid manufacturing system models. In *Proc. of 8th IEEE Mediterranean Conf. on New Directions in Control and Automation*, 2000.

M. Grant, S. Boyd, and Y. Ye. Disciplined convex programming. *Chapter in Global Optimization: From Theory to Implementation, L. Liberti and N. Maculan (eds.), Springer*, pages 155–210, 2006.

W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocols for wireless microsensor networks. In *Proceedings of the 33rd Annual Hawaii International Conference on Systems Science*, page 10, Hawaii, USA, 2002.

K. Jeffay, D.F. Stanat, and C.U. Martel. On non-preemptive scheduling of periodic and sporadic tasks. In *Proc. of the IEEE Real-Time Systems Symposium*, pages 129–139, 1991.

J. Mao and C.G. Cassandras. Optimal control of multi-stage discrete event systems with real-time constraints. In *Proc. of 45rd IEEE Conf. Decision and Control*, pages 1057–1062, 2006. (subm. to IEEE Trans. on AC, 2007).

J. Mao and C.G. Cassandras. Optimal control of multi-layer discrete event systems with real-time constraints. *Technical Report, Boston University*, 2007. See also ftp://dacta.bu.edu:2491/TechReport/2007VDAmL.pdf.

J. Mao, C.G. Cassandras, and Q.C. Zhao. Optimal dynamic voltage scaling in power-limited systems with real-time constraints. *IEEE Trans. on Mobile Computing*, 6(6):678–688, June 2007.

L. Miao and C. G. Cassandras. Optimal transmission scheduling for energy-efficient wireless networks. In *Proc. of INFOCOM*, 2006.

L. Miao and C. G. Cassandras. Optimality of static control policies in some discrete event systems. *IEEE Trans. on AC*, 50(9):1427 – 1431, Sep. 2005.

D.L. Pepyne and C.G. Cassandras. Optimal control of hybrid systems in manufacturing. In *Proc. of the IEEE*, volume 88, pages 1108–1123, 2000.

F. Yao, A. Demers, and S. Shenker. A scheduling model for reduced cpu energy. In *Proc. of the 36th Annual Symposium on Foundations of Computer Science*, pages 374–382. IEEE Computer Society, 1995.