Automatica 89 (2018) 392–397

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Brief paper

Priority-free conditionally-preemptive scheduling of modular sporadic real-time systems ☆

Xi Wang [a], Zhiwu Li [b,c,*], W.M. Wonham [d]

[a] School of Electro-Mechanical Engineering, Xidian University, Xi'an 710071, China
[b] Institute of Systems Engineering, Macau University of Science and Technology, Taipa 999078, Macau
[c] Key Laboratory of Electronic Equipment Structure Design, Ministry of Eduction, Xi'an 710071, China
[d] Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S 3G4, Canada

ARTICLE INFO

Article history:
Received 19 July 2016
Received in revised form 20 July 2017
Accepted 2 November 2017
Available online 30 January 2018

Keywords:
Real-time system
Scheduling
Discrete-event system
Supervisory control
Multiple-period
Conditional-preemption

ABSTRACT

For a real-time system (RTS) processing both sporadic and (multiple-period) periodic tasks, this study presents a novel modular modeling framework to describe the parameters of tasks, conforming to the pertinent concepts and techniques of discrete-event systems (DES). A task is represented by an automaton synchronized by the modular models corresponding to its parameters. As a consequence, a DES model depicting the RTS is synchronized by the DES representing these tasks. Based on supervisory control theory, priority-free conditionally-preemptive (PFCP) real-time scheduling is solved by finding all the safe execution sequences. Finally, the PFCP scheduling is illustrated by real-world examples.

1. Introduction

Supervisory control theory (SCT) (Cury, De Queiroz, Bouzon, & Teixeira, 2015; Ramadge & Wonham, 1987; Wonham & Cai, 2017) aims to discover general principles common to a wide range of application domains such as manufacturing systems, traffic systems, database management systems, communication protocols, and logistic (service) systems. SCT has been implemented for real-time scheduling (Alareqi, Gorges, & Liu, 2015; Fontanelli, Greco, & Palopoli, 2013; Veenman & Scherer, 2014) of real-time systems (RTS), in which real-time tasks are modeled following two different approaches: timed discrete-event systems (TDES) (Chen & Wonham, 2002; Janarthanan, Gohari, & Saffar, 2006; Park & Cho, 2008; Wang, Li, & Wonham, 2016) and (untimed) discrete-event systems (DES) (Wang, Li, & Wonham, 2017). For the TDES-based real-time scheduling, SCT can provide optimal schedulers satisfying the preemptive or non-preemptive scheduling policies defined in specifications. Wang et al. (2017) show that both preemptive and non-preemptive scheduling policies are conservative, i.e., some scheduling specifications may not be consistent with the preemptive or non-preemptive scheduling policies. Based on SCT, a novel real-time scheduling principle, namely priority-free conditionally-preemptive (PFCP) real-time scheduling, is developed. Accordingly, some classic real-time scheduling policies such as fixed-priority (FP) scheduling (Liu & Layland, 1973), preemption threshold scheduling (PTS) (Wang et al., 2015; Wang & Saksena, 1999), and deferred preemption scheduling (DPS) (Baruah, 2005) are considered as consequences of the developed specifications.

SCT-based real-time scheduling (Chen & Wonham, 2002; Janarthanan et al., 2006; Park & Cho, 2008; Wang et al., 2016, 2017) is a newly-identified research topic. In this study, a formal and unified modular real-time task framework is developed, which can be utilized to model both multiple-period and sporadic tasks that are not constrained by periods. Based on this modeling framework, each RTS can be built via a three-step approach: (1) the task parameters are represented by modular DES models; (2) the task behavior is constrained by the modular DES models corresponding to its parameters; and (3) the RTS model is constrained by the DES real-time task models corresponding to the running tasks. According to the PFCP principle, all the safe execution sequences in an RTS processing both multiple-period periodic and sporadic tasks can be found. This model is applied to a real-world manufacturing example.

The rest of this paper is organized as follows. The modular real-time task model is described in Section 2. By applying the

☆ The material in this paper was not presented at any conference. This paper was recommended for publication in revised form by Associate Editor Christoforos Hadjicostis under the direction of Editor Christos G. Cassandras.

* Corresponding author at: Institute of Systems Engineering, Macau University of Science and Technology, Taipa 999078, Macau.

E-mail addresses: wangxi@xidian.edu.cn (X. Wang), zhwli@xidian.edu.cn (Z. Li), wonham@ece.utoronto.ca (W.M. Wonham).

https://doi.org/10.1016/j.automatica.2017.12.010
0005-1098/

supervisory control of DES to real-world systems, Section 3 reports applications of real-time scheduling with priority-free conditional-preemption. Further relevant issues are discussed in Section 4. Finally, Section 5 concludes this paper.

2. Modular sporadic RTS model

This study deals with sporadic RTS processing sporadic and/or multiple-period tasks. Sporadic RTS and the corresponding modular DES models are defined below.

2.1. Basic concepts

A DES plant is a *generator* $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m)$ where Q is the finite *state set*; Σ is the finite *event set* (*alphabet*); $\delta : Q \times \Sigma \to Q$ is the *partial state transition function*; q_0 is the *initial state*; and $Q_m \subseteq Q$ is the subset of *marked states*. In accordance with Wonham and Cai (2017), Σ^+ is the set of all the possible strings of symbols in Σ. After adjoining the *empty string* ϵ, the set of strings over the alphabet Σ is written as Σ^*, i.e., $\Sigma^* = \Sigma^+ \cup \{\epsilon\}$. Function δ can be extended to $\delta : Q \times \Sigma^* \to Q$. We write $\delta(q, s)!$ to mean that $\delta(q, s)$ is *defined*, where state $q \in Q$ and string $s \in \Sigma^*$. The *length* $|s|$ of a string $s \in \Sigma^*$ is defined as $|\epsilon| = 0$; $|s| = k$, if $s = \sigma_1\sigma_2\cdots\sigma_k \in \Sigma^+$. If $L \subseteq \Sigma^*$, the *prefix closure* of language L is denoted by $\overline{L}$ consisting of all prefixes of strings of L. The *closed behavior* of generator $\mathbf{G}$ is represented by *formal language* $L(\mathbf{G}) := \{s \in \Sigma^* | \delta(q_0, s)!\}$ and the corresponding *marked behavior* is $L_m(\mathbf{G}) := \{s \in L(\mathbf{G}) | \delta(q_0, s) \in Q_m\} \subseteq L(\mathbf{G})$. *Synchronous product* (Wonham & Cai, 2017) is a standard operation to combine a finite set of DES into a single and more complex one. Given n DES, where $L_i \subseteq \Sigma_i^*$ with $\Sigma = \bigcup_{i \in \mathbf{n}} \Sigma_i$, and $\mathbf{n} := \{1, 2, \ldots, n\}$, the *natural projection* $P_i : \Sigma^* \to \Sigma_i^*$ is defined by (1) $P_i(\epsilon) = \epsilon$, (2) $P_i(\sigma) = \begin{cases} \epsilon, & \text{if } \sigma \notin \Sigma_i \\ \sigma, & \text{if } \sigma \in \Sigma_i \end{cases}$, and (3) $P_i(s\sigma) = P_i(s)P_i(\sigma)$, $s \in \Sigma^*, \sigma \in \Sigma$. The *inverse image function* of P_i is $P_i^{-1} : Pwr(\Sigma_i^*) \to Pwr(\Sigma^*)$. For $H \subseteq \Sigma_i^*, P_i^{-1}(H) := \{s \in \Sigma^* | P_i(s) \in H\}$. The *synchronous product* of $L_1, L_2, \ldots, L_n$ is denoted by $L_1 \| L_2 \| \cdots \| L_n$ with $L_1 \| L_2 \| \cdots \| L_n := P_1^{-1}L_1 \cap P_2^{-1}L_2 \cap \cdots \cap P_n^{-1}L_n$.

2.2. Sporadic RTS model

In a sporadic RTS, a sporadic task has an irregular arrival time and an either *soft* or *hard* deadline; a periodic task has a regular arrival time and a hard deadline. The period of a periodic task could be: equal to the corresponding deadline (Liu & Layland, 1973), greater than the deadline (Nassor & Bres, 1991), or multiple (Wang et al., 2016). In case the minimum period of a multiple-period periodic task is equal to its maximum period, it is a traditional periodic task.

Suppose that a uniprocessor sporadic RTS $\mathbb{S}$ processes a set of independent real-time tasks discussed above, i.e., $\mathbb{S} = \{\tau_1, \tau_2, \ldots, \tau_n\}$. For $i \in \mathbf{n} = \{1, 2, \ldots, n\}$, if $\tau_i \in \mathbb{S}$ is periodic, it is specified as a four-tuple $\tau_i = (R_i, C_i, D_i, \mathbf{T}_i)$ with a *release time* R_i, a *worst-case execution time* (*WCET*) C_i, a *hard deadline* D_i, and a *multiple-period* $\mathbf{T}_i$, where R_i, C_i, and D_i are integer multiples of the processor time unit. A multiple-period (Wang et al., 2016) is a period set containing several possible periods (positive integers): the minimum (resp., maximum) period is represented by $T_{i_{min}}$ (resp., $T_{i_{max}}$). Thus, we have $\mathbf{T}_i = [T_{i_{min}}, T_{i_{max}}]$. Only a period T in $\mathbf{T}_i$ of task τ_i is selected in each scheduling period. Normally, D_i is ignored in case $D_i = T_{i_{max}}$. Since release time and periods are not assigned to sporadic tasks, a sporadic task is specified by a pair $\tau_i = (C_i, D_i)$.

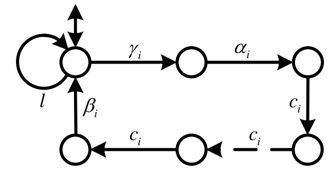

Fig. 1. WCET model.

2.3. Modular DES models

In accordance with Wang et al. (2017), the *alphabet* (set of *event labels*) Σ_i describing the processor's behavior to execute task τ_i is:

- γ_i: task τ_i is *released*;
- α_i: the execution of τ_i is *started*;
- β_i: the execution of τ_i is *completed*;
- c_i ($i \in \mathbf{n}$): τ_i starts to be *processed* in the processor for one *processor time unit*; and
- l: *empty action*, i.e., no task is being processed in the following processor time unit.

The global event set of an RTS $\mathbb{S}$ is denoted by $\Sigma = \Sigma_1 \cup \Sigma_2 \cup \cdots \cup \Sigma_n$. As a consequence, the WCET, release time, period, and deadline of a task running in $\mathbb{S}$ are described by languages over Σ. The alphabet Σ is partitioned into *controllable* events and *uncontrollable* events. Formally, $\Sigma = \Sigma_c \dot{\cup} \Sigma_u$, with $\Sigma_c = \{\alpha_i, c_i | i \in \mathbf{n}\}$ as the controllable event subset and $\Sigma_u = \{\beta_i, \gamma_i, l | i \in \mathbf{n}\}$ as the uncontrollable event subset. Moreover, Σ_i is also partitioned into the *operation event set* $\Sigma_o = \{\gamma_i, \alpha_i, \beta_i | i \in \mathbf{n}\}$ and the *execution event set* $\Sigma_e = \{c_i, l | i \in \mathbf{n}\}$.

2.3.1. WCET model

A sporadic (resp., periodic) task τ_i describes an infinite stream of jobs arriving at irregular (resp., regular) time intervals. Suppose that n sporadic/periodic tasks are running in an RTS. For each task $\tau_i, i \in \mathbf{n}$, its WCET C_i is represented by a DES generator $\mathbf{G}_i^C$. Suppose that $s \in \Sigma^*$ and accordingly s^* represents $\epsilon + s + s^2 + \cdots$. The *marked language* $L_m(\mathbf{G}_i^C)$ over Σ describes the execution sequences of task τ_i, i.e.,

$$L_m(\mathbf{G}_i^C) = (uvw)^* \quad (1)$$

with

- $u, w \in L_l = l^* = \epsilon + l + l^2 + \cdots$; and
- $v \in L_a = (\gamma_i\alpha_i(c_i)^{C_i}\beta_i)^*$.

Language L_a (resp., L_l) represents τ_i arriving (resp., never arriving), in which $(c_i)^{C_i}$ means the execution of task τ_i. The WCET model is depicted in Fig. 1.

2.3.2. Release time and period model

Generally, for any periodic task τ_i, we have $C_i \le T_{i_{min}}$. Let T be an arbitrary period, i.e., $T_{i_{min}} \le T \le T_{i_{max}}$. In a period, after the release of τ_i, C_i time units are utilized to process τ_i. According to Wang et al. (2017), the other $T - C_i$ time units are occupied by other tasks τ_j in $\mathbb{S}$ ($\tau_i \neq \tau_j$) or alternatively left idle. A release time and period model for τ_i is presented, which describes the system behavior before its first release and in each period T. The marked language $L_m(\mathbf{G}_i^T)$ is over Σ, i.e.,

$$L_m(\mathbf{G}_i^T) = \overline{L_r} + u(\gamma_i v)^* \quad (2)$$

with

- $u \in L_r = \{s | s \in (\Sigma_e - \{c_i\})^* \;\&\; |s| = R_i\}$; and