PERGAMON

Control Engineering Practice 8 (2000) 689–695

www.elsevier.com/locate/conengprac

A new algorithm to prove the schedulability of real-time systems

Shuhua Wang[a,*,1], Georg Färber[b]

[a]1CM6, ROHDE and SCHWARZ Gmbh and Co. KG, Postfach 80 14 69, D-81614 München, Germany

[b]Institute for Real-Time Computer Systems, Technische Universitat München, Postfach 80 14 69, D-81614 München, Germany

Received 7 October 1999; accepted 20 December 1999

Abstract

In hard real-time multiprocessor systems, tasks not only have timing constraints but also often have precedence constraints caused by communication among themselves. In this paper a new algorithm to prove the schedulability of real-time systems is proposed, in which both precedence constraints and communication costs are considered and represented by offsets and modified deadlines. To obtain a tight upper bound for the worst-case response time, the concepts of *local critical instant* and *local worst-case response time* are introduced. The proposed algorithm is compared with other algorithms using test cases. The comparison shows that the proposed algorithm has improved the performances to these compared algorithms. © *2000 Elsevier Science Ltd. All rights reserved.*

Keywords: Algorithm; Real-time system; Deadline; Offset; Constraint

1. Introduction

A real-time system must be both functionally and temporally correct. This is especially true for hard real-time systems, where a key requirement is whether hard real-time conditions are fulfilled or not, i.e., whether each task is guaranteed to meet its deadline. The proof is performed by using schedulability analysis.

Fixed priority preemptive scheduling methods are efficient ways of constructing and analyzing schedules for hard real-time systems. Among them, deadline monotonic scheduling is more suitable for use in parallel environments with communicating tasks, since precedence constraints caused by communication could be taken into consideration by modifying the original deadlines of the tasks.

The majority of the schedulability analyses performed to date have assumed a *critical instant*, a concept introduced by Liu and Layland (Liu & Layland, 1973). They stated that for a task set in which all tasks are independent there is a critical instant, i.e., when all tasks are simultaneously released. If the schedulability analysis is carried out for the critical instant and all tasks are assumed to execute in their worst-case execution times, then the test is both sufficient and necessary. However, in multiprocessor systems, tasks often have precedence constraints caused by communication among them. This means that all tasks cannot be simultaneously released. Hence the schedulability analysis based on the critical instant becomes pessimistic.

There are several approaches in which precedence constraints are considered (Altenbernd, 1995; Tindell, 1994; Bate & Burns, 1997; Wang & Färber, 1998). An exact analysis was presented in Audsley, Tindell and Burns (1993b). Since the exact analysis is computationally intractable, a sufficient but not necessary analysis was developed (Tindell, 1994). Recently, a non-preemptive fixed priority schedulability analysis with offset was proposed (Bate & Burns, 1997). The limitation of these analyses is that they are based on uniprocessor systems.

For multiprocessor systems, an analysis using minimal–maximal offset intervals was presented (Altenbernd, 1995). In that paper, the computation of the worst-case response time is divided into two parts: transaction response time and interference of other transactions. However, the communication cost between tasks residing on different processors is not considered. Communication costs influence both the predecessor task and the successor task and make their deadlines harder to be met.

[1] Present address. Connollystrasse 11, App. H15, D-80809 München, Germany.

* Corresponding author. Fax: + 49-89-41293443.

E-mail address: Shuhua.Wang@rsd.rsd.de (S. Wang).

0967-0661/00/$ - see front matter
PII: S0967-0661(00)00011-3

An improved analysis (Wang & Färber, 1998) was proposed by the authors of this paper. In that paper the communication cost is considered and the calculation of transaction response time was more accurate. However, the calculation of interference by the other transactions was not changed and is based on a less accurate algorithm. Thus the analysis is also pessimistic in some cases.

In this paper, the worst-case response time is analyzed with another method. The new analysis algorithm is based on a *busy period analysis* (Lehoczky, Sha & Ding, 1989; Tindell, Burns & Wellings, 1994). Simulation results show that it is more accurate than the analysis algorithm used most often in the literature.

The rest of the paper is organized as follows: Section 2 briefly describes the basic assumptions that will be used in the proposed analysis algorithm. Then the new algorithm is presented in Section 3. Section 4 gives evaluation results in which the proposed schedulability analysis is compared with other approaches using test cases. Finally, Section 5 summarizes the result of this paper.

2. Basic assumptions

A real-time application consists of a set $\Gamma = \{\tau_i: 1 \leqslant i \leqslant n\}$ of tasks. A task refers to a single thread of control in the system. Task τ_i refers to the arrival of the task or the release of the task for execution in the system. In general tasks could be associated with one of the following categories:

- *Periodic task*: which is released at regular intervals of time. Each periodic task gives rise to an infinite sequence of invocations.
- *Aperiodic task*: which could be released at any instant of time. Normally, it has only a soft deadline and corresponds to an event or external stimuli of the system.
- *Sporadic task*: which is similar to an aperiodic task but with a hard deadline and the additional constraint on the minimum time interval between two successive invocations of the same task.

In this paper, periodic tasks and sporadic tasks are considered. Each task $\tau_i \in \Gamma$ has a worst-case execution time c_i, a relative deadline D_i and a minimum time interval T_i between two consecutive invocations.

Because tasks communicate with each other, precedence constraints arise from communicating tasks. These constraints are represented by a partial order $\prec$ over Γ. If task τ_i cannot begin its execution until task τ_j has completed its execution, it is represented as $\tau_j \prec \tau_i$. Tasks τ_j and τ_i are called predecessor and successor tasks, respectively.

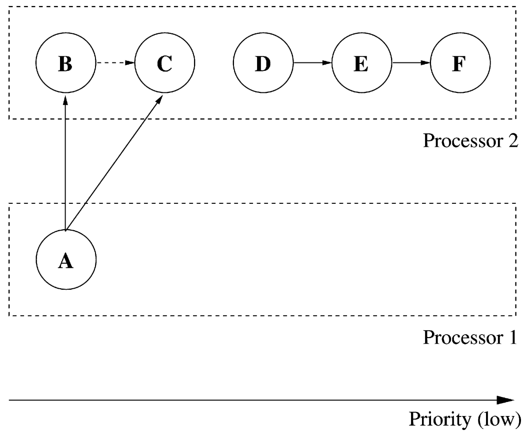

Fig. 1. An example of a task graph.

The task set and the intertask precedence constraints are represented by a directed acyclic graph that is also called task graph $TG = (N, A)$. N is a set of nodes representing the tasks in the set Γ. A is a set of directed arcs representing the precedence constraints between the tasks in Γ, that is, if $\tau_j \prec \tau_i$. Then $(\tau_j, \tau_i) \in A$. Additionally, the task graph also contains information about task allocation. An example of task graphs is shown in Fig. 1.

In addition to task parameters and task graph, the general assumptions which will be used in the new analysis are as follows:

- Tasks are statically allocated on a number of homogeneous processors.
- Communication between two tasks residing on the same processor is done via accessing shared memory and its cost can be negligible.
- Communication between two tasks residing on different processors is done via a shared bus system with a token-passing protocol.
- Scheduling is done locally for each processor by deadline monotonic scheduling policy. Preemption by higher priority tasks is allowed.
- For all tasks, $D_i \leqslant T_i$.

To predict operating system overheads Burns (1993) proposed a method in which two parameters are added to the task set. For clarity, it is assumed in this paper that various overheads such as the context switching are included in the task worst-case execution time c_i.

3. The new schedulability analysis algorithm

In hard real-time multiprocessor systems, tasks often have precedence constraints caused by communication among them. This means that all tasks cannot be simultaneously released. Hence the schedulability analysis based on critical instant becomes pessimistic.