Available online at www.sciencedirect.com

ScienceDirect

Journal of Applied Logic 5 (2007) 667–680

JOURNAL OF APPLIED LOGIC

www.elsevier.com/locate/jal

A polynomial time algorithm for Zero-Clairvoyant scheduling

K. Subramani [1]

LDCSEE, West Virginia University, Morgantown, WV, USA

Received 17 November 2005; received in revised form 19 June 2006; accepted 7 August 2006

Available online 21 December 2006

Abstract

In the design of real-time systems, it is often the case that certain process parameters, such as the execution time of a job are not known precisely. The challenge in real-time system design then, is to develop techniques that efficiently meet the requirements of impreciseness, while simultaneously guaranteeing safety. In a traditional scheduling model, such as the one discussed in [M. Pinedo, Scheduling: Theory, Algorithms, and Systems, Prentice-Hall, Englewood Cliffs, 1995. [23]]; [P. Brucker, Scheduling Algorithms, second ed., Springer, 1998. [3]], the tendency is to either overlook the effects of impreciseness or to simplify the issue of impreciseness by assuming *worst-case* values. This assumption is unrealistic and at the same time, may cause certain timing constraints to be violated at run-time. Further, depending on the nature of the constraints involved, it is not immediately apparent, what the worst-case value for a given parameter is. Whereas, in traditional scheduling, constraints among jobs are no more complex than those that can be represented by a precedence graph, in case of real-time scheduling, complicated constraints such as relative timing constraints are commonplace. Additionally, the purpose of scheduling is to achieve a schedule that optimizes some performance metric, whereas in real-time scheduling the goal is to ensure that the imposed constraints are met at run-time. In this paper, we study the problem of scheduling a set of ordered, non-preemptive jobs under non-constant execution times. Typical applications for variable execution time scheduling include process scheduling in Real-Time Operating Systems such as Maruti, compiler scheduling, database transaction scheduling and automated machine control. An important feature of application areas such as robotics is the interaction between execution times of various processes. We explicitly model this interaction through the representation of execution time vectors as points in convex sets. This modeling vastly extends previous models of execution times as either single points or range-bound intervals. Our algorithms do not assume any knowledge of the distributions of execution times, i.e. they are *Zero-Clairvoyant*. We present both sequential and parallel algorithms for determining the existence of a Zero-Clairvoyant schedule. To the best of our knowledge, our techniques are the first of their kind.

Keywords: Zero-Clarirvoyant; Real-time scheduling; Polytope; Dispatching; Fault-tolerance

1. Introduction

Scheduling strategies for real-time systems confront two principal issues that are not addressed by traditional scheduling models viz., parameter variability and the existence of complex timing constraints among constituent jobs. In particular, execution times of jobs within a job-set are almost never known with certitude. Impreciseness in problem data is of both theoretical and practical significance. From an empirical perspective, system designers have used *worst-*

E-mail address: ksmani@csee.wvu.edu.

[1] This research has been supported in part by the Air-Force Office of Scientific Research under grant FA9550-06-1-0050.

1570-8683/$ – see front matter
doi:10.1016/j.jal.2006.08.001

case values in order to address non-determinism of execution time values [23]. However, the assumption that every job will have an execution time equal to the maximum value in its allowable range is unrealistic and at the same time, may cause constraint violation at run-time. From the theoretical perspective, it is important to develop models and algorithms that explicitly account for uncertainty.

In this paper, we study the problem of scheduling a set of ordered, non-preemptive jobs with non-constant execution times, with the goal of obtaining a single, rational, start time vector, such that the constraints on the jobs are satisfied, irrespective of the actual execution times of the jobs at run time. We explicitly model execution time non-determinism through convex sets. To the best of our knowledge, our work represents the first effort in studying this generalization of execution time domains. Our algorithm is Zero-Clairvoyant in that it makes no assumptions about the distribution of execution times; we present both sequential and parallel algorithms for determining the existence of such a Zero-Clairvoyant schedule. Zero-Clairvoyant schedules are also called *Static schedules*, since the schedule in every single window is the same (with appropriate offsets). We shall be using the terms Static and Zero-Clairvoyant interchangeably, for the rest of this paper.

We are concerned with the following problems:

(a) Determining the Zero-Clairvoyant schedulability of a job set in a periodic real-time system (defined in Section 2),
(b) Determining the dispatch vector of the job set in a scheduling window.

The rest of this paper is organized as follows: In Section 2, we detail the Zero-Clairvoyant scheduling problem and pose the Zero-Clairvoyant schedulability query. The succeeding section, viz., Section 3, motivates the necessity for Zero-Clairvoyant scheduling, while Section 4 describes related approaches to this problem. Section 5 commences the process of answering the Zero-Clairvoyant schedulability query posed in Section 2 through the application of the convex programming techniques, discussed in [8]. The algorithm we present is very general, in that it is applicable as long as the execution time vectors belong to a convex set and the constraints on the system are linear. A straightforward parallelization of the algorithm is provided, subsequent to the complexity analysis. Section 6 specializes the algorithm in Section 5 to a number of interesting restrictions. A detailed implementational profile is provided in Section 7. We conclude in Section 8 by tabulating the results discussed in this paper and outlining problems for future research. In order to make this exposition self-contained, we have included a discussion on *Convex Programming* and *convex minimization* algorithms in Appendix A.

2. Statement of problem

2.1. Job model

Assume an infinite time-axis divided into windows of length L, starting at time $t = 0$. These windows are called *periods* or *scheduling windows*. There is a set of non-preemptive, ordered jobs, $\mathcal{J} = \{J_1, J_2, \ldots, J_n\}$ that execute in each scheduling window.

2.2. Constraint model

The constraints on the jobs are described by system (1):

$$\mathbf{A} \cdot [\vec{\mathbf{s}}\, \vec{\mathbf{e}}]^{\mathrm{T}} \leqslant \vec{\mathbf{b}}, \quad \vec{\mathbf{e}} \in \mathbf{E}, \tag{1}$$

where,

- $\mathbf{A}$ is an $m \times 2.n$ rational matrix, $\vec{\mathbf{b}}$ is a rational m-vector; $(\mathbf{A}, \vec{\mathbf{b}})$ is called the constraint matrix;
- $\mathbf{E}$ is an arbitrary convex set;
- $\vec{\mathbf{s}} = [s_1, s_2, \ldots, s_n]$ is the start time vector of the jobs, and
- $\vec{\mathbf{e}} = [e_1, e_2, \ldots, e_n] \in \mathbf{E}$ is the execution time vector of the jobs.

In this work, we consider generalized linear constraints among jobs i.e. those that can be expressed in the form: $\sum_{i=1}^{n} a_i.s_i + b_i.e_i \leqslant k$, for arbitrary rationals a_i, b_i, k.

The convex set $\mathbf{E}$ serves to model the following situations: