ARTICLE IN PRESS

Discrete Applied Mathematics ■ (■■■■) ■■■–■■■

Contents lists available at ScienceDirect

Discrete Applied Mathematics

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

A polynomial algorithm for the homogeneously non-idling scheduling problem of unit-time independent jobs on identical parallel machines

Philippe Chrétienne [a,*], Alain Quilliot [b]

[a] Sorbonne Universités, UPMC Univ Paris 06, UMR 7606, LIP6, F-75005, Paris, France
[b] Université Blaise Pascal, UMR 6158, LIMOS, F-63173, Aubière, France

ARTICLE INFO

Article history:
Received 20 September 2016
Received in revised form 7 November 2017
Accepted 1 February 2018
Available online xxxx

Keywords:
Parallel-machine scheduling
Directed graph
Algorithmic complexity

ABSTRACT

In this paper, we consider the homogeneous m-machine scheduling problem where unit-time jobs have to be scheduled within their time windows so that, for any subset of machines, the set of the time units at which at least one machine is busy, is an interval. For this problem, a time assignment of the jobs satisfying the time windows constraints is a schedule if it has a pyramidal profile and is a k-schedule if this profile property is satisfied up to level k only. We develop a generic algorithm to solve the problem and show it is correct using a proof that mainly relies on a necessary and sufficient condition for a schedule to exist proved in a previous paper. We finally show that the generic algorithm is polynomial. We conclude by giving the directions of ongoing works and by bringing open questions related to different variants of the basic non-idling m-machine scheduling problem.

1. Introduction

Most scheduling problems assume that no cost is incurred when a machine waits between the completion of a job and the start of the next job. Moreover, it is well-known that such waiting delays are often necessary to get optimality. However, in some applications such as those described in [6], the cost of making a running machine stop and restart later is so high that a non-idling constraint is put on the machines, so that only schedules without any intermediate delays are required. Problems concerning power management policies may also yield similar scheduling problems [3] where for instance each idling period has a cost and the total cost has to be minimized [1].

As a matter of fact, non-idling management is usually a cost issue, rather than a hard constraint. While in other papers involving non-idling requirements, authors express this cost issue as a constraint related to machines or resources independently from each other, we focus here on situations which impose this constraint to any group of machines, all considered as being identical: when identical machines are strongly interconnected and involve a common source of additional resources (energy, human resources), then the cost which derives from stopping or starting a given machine may be high (in comparison with other costs) and depend in a decreasing way on the current number of active machines. So, minimizing those costs leads to concentrating task executions into active periods with high parallelism rates, which means to designing schedules for which the number of busy machines has a pyramidal shape over time, a property that will be at the core of the following sections.

* Corresponding author.
E-mail address: philippe.chretienne@lip6.fr (P. Chrétienne).

https://doi.org/10.1016/j.dam.2018.02.011

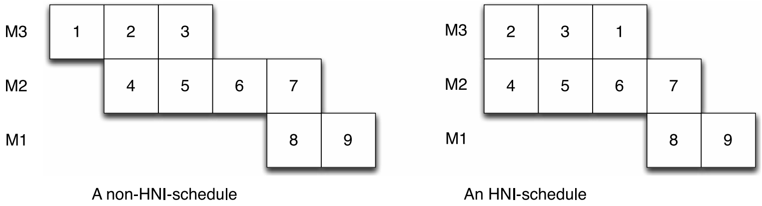

Fig. 1. HNI and non-HNI schedules.

The complexity of single machine scheduling problems involving non-idling constraints was studied in [2], and, in [4], exact methods were designed in order to solve the basic one-machine non-idling problem. Also, in [5], approximation algorithms have been developed for the non-idling single-machine scheduling problem with release and delivery times.

More recently (see [7]), we have been performing a theoretical analysis of the basic homogeneous m-machine non-idling problem where weakly dependent unit-time jobs have to be scheduled within their time windows so that the non-idling constraint must be satisfied not only for each machine but for every subset of machines. Our main contribution there was a min/max necessary and sufficient condition for this problem to admit a feasible solution but left opened the complexity question and did not lead to an algorithm solving the problem.

This paper gives an answer to that question. In Section 2, we define the homogeneously non-idling scheduling problem. Section 3 defines the m-matchings of an instance, the notions of T-block, k-hole and propagation path attached to an m-matching, specifies what makes an m-matching be a k-schedule and gives some basic properties of these objects. Section 4 describes a generic algorithm solving the problem. Section 5 is devoted to the correctness proof of the generic algorithm. In Section 6 we show that the generic algorithm is polynomial. In the conclusion, we present some problem extensions that have been polynomially solved and give some future research directions.

2. Problem definition

2.1. Preliminary notations

We consider the discrete time space $\mathbb{N}$ each element of which is called a time-unit. A subset Θ of $\mathbb{N}$ is an *interval* if it is made of a finite number of consecutive time-units. The earliest (respectively latest) time-unit of an interval Θ is denoted by $a(\Theta)$ (respectively $b(\Theta)$).

Interval Θ_2 is said to dominate interval Θ_1 (what is denoted by $\Theta_1 \propto \Theta_2$) if $b(\Theta_1)+1 < a(\Theta_2)$. If $\Theta_1 \propto \Theta_2$, we denote by $Mid(\Theta_1, \Theta_2)$ the (non-empty) interval $[b(\Theta_1)+1, a(\Theta_2)-1]$. By convention, two intervals Θ_1 and Θ_2 are said to be *connected* if $\Theta_1 \cup \Theta_2$ is an interval.

2.2. The homogeneous non-idling scheduling problem

An instance $I = (J, F, m)$ of the homogeneous non-idling scheduling problem is defined as follows. $J = \{J_1, \ldots, J_n\}$ is a set of n unit-time independent jobs. $M = \{M_1, \ldots, M_m\}$ is a set of m identical machines. Each job J_i must be executed by a machine within a given time-window $F(i) = \{r_i, \ldots, d_i\}$ which is an interval.

A schedule of the instance (J, F, m) must satisfy the resource constraint ((RE) in short), the time-window constraint ((TW) in short), and the following "homogeneous non-idling" constraint ((HNI) in short): for any subset $M' \subseteq M$, the time-units at which *at least one machine* in M' is busy make an interval.

The constraint (HNI) is illustrated in Fig. 1. While each machine satisfies the non-idling constraint, the time diagram on the left does not satisfy the constraint (HNI) since the times units when M_1 or M_3 is busy do not make an interval. On the contrary, the time diagram on the right satisfies the constraint (HNI).

It is easy to see that every schedule may be transformed into an equivalent "histogram" schedule where, if k machines are busy at a given time unit, these machines are $M_1, \ldots, M_k$. So, in the remainder of the paper we will only consider histogram schedules.

We denote by r_{min} (respectively d_{max}) the smallest r_i (respectively largest d_i) and by $\mathcal{H}$ the interval $\{r_{min}, \ldots, d_{max}\}$.

Let Π_0 be the problem to decide whether a given instance (J, F, m) has at least one schedule. If the answer is yes, the instance is said to be *feasible*.

3. *m*-matchings

Let $I = (J, F, m)$ be an instance of Π_0. A time function $T : J \mapsto \mathbb{N}$ that satisfies the constraints (RE) and (TW) will be called a *m-matching* of I. If T is an m-matching of I, we denote by $s(T)$ (respectively $e(T)$) the smallest (respectively largest) time-unit such that at least one job is processed. The interval $\{s(T), \ldots, e(T)\}$ is denoted by $L(T)$. If $t \in L(T)$, we denote by