Computers & Industrial Engineering 59 (2010) 272–279

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Exact and heuristic algorithms for parallel-machine scheduling with DeJong's learning effect ☆

Dariusz Okołowski, Stanisław Gawiejnowicz *

Adam Mickiewicz University, Faculty of Mathematics and Computer Science, Umultowska 87, 61-614 Poznań, Poland

ARTICLE INFO

Article history:
Received 22 September 2008
Received in revised form 19 March 2010
Accepted 20 April 2010
Available online 24 April 2010

Keywords:
Parallel-machine scheduling
Learning effect
Branch-and-bound algorithms
Heuristic scheduling algorithms

ABSTRACT

We consider a parallel-machine scheduling problem with a learning effect and the makespan objective. The impact of the learning effect on job processing times is modelled by the general DeJong's learning curve. For this $\mathcal{NP}$-hard problem we propose two exact algorithms: a sequential branch-and-bound algorithm and a parallel branch-and-bound algorithm. We also present the results of experimental evaluation of these algorithms on a computational cluster. Finally, we use the exact algorithms to estimate the performance of two greedy heuristic scheduling algorithms for the problem.

1. Introduction

Scheduling problems concerning multi-machine production environments are encountered in many modern manufacturing processes (Leung, 2004; Pinedo, 2008). Since the classical scheduling theory (Conway, Maxwell, & Miller, 1967) turned out to be too rigid for some of these environments, the theory started to evolve in early 1990s. This led to a rise of new scheduling models such as, e.g., scheduling with controllable job processing times (Shabtay & Steiner, 2007), scheduling jobs with time-dependent processing times (Gawiejnowicz, 2008) and multiprocessor task scheduling (Drozdowski, 2009). An important group of such new scheduling models constitute scheduling models with the so-called *learning effect* (Biskup, 2008) that we consider in this paper.

The impact of the learning effect on production issues was first discussed in 1936 by Wright (Biskup, 2008), who observed that learning may decrease the processing times of production tasks in the aircraft industry. The observation was later confirmed by many empirical studies saying that the knowledge of a learning curve during the planning process may result in cost savings in manufacturing, industrial production and software engineering (Badiru, 1992; Globerson & Seidmann, 1988; Raccoon, 1995).

A general learning effect is modelled in scheduling theory by assumption that the processing time of a job is a function of the job position in a schedule. In literature there are known different models of the learning effect that lead to distinct forms of the function (see, e.g., Bachman & Janiak, 2004; Biskup, 2008; Dondeti & Mohanty, 1998; Gawiejnowicz, 1996, for details). Thus, since the learning effect allows to take into account human factors in scheduling, the problems similar to the mentioned above belong to intensively studied topics in scheduling research (Lodree, Geiger, & Jiang, 2009).

Throughout the paper, we will consider the following scheduling problem with a learning effect. There is given a set of jobs $J_1, J_2, \ldots, J_n$ which have to be processed on machines $M_1, M_2, \ldots, M_m$. All jobs are available for processing at time 0 and job preemption is not allowed. The processing time of job J_j is described by *DeJong's learning curve*, $p_{j,r} = p_j(M + (1 - M)r^a)$, where p_j is the initial job processing time, $a \leqslant 0$ is the learning index, M is the incompressibility factor, r is the current position of a job in a given schedule and $1 \leqslant j, r \leqslant n$. To the best of our knowledge, no scheduling problems with this form of a learning effect have been considered earlier.

Function $p_{j,r}$, introduced by DeJong (Badiru, 1992), mirrors the impact of a learning effect on job processing times and is a generalization of both the *log-linear learning curve*, $p_{j,r} = p_j r^a$, and the case of fixed job processing times, $p_{j,r} = p_j$. (The first case is obtained for $M = 0$, while the second one for $M = 1$.) The main advantage of DeJong's model follows from the fact that parameter M represents the part of job processing time that is limited by some conditions and cannot be shortened. Different values of M are recommended in literature. For example, DeJong suggests $M = 0.25$ for labour-intensive jobs and $M = 0.5$ for machine-intensive jobs (Raccoon, 1995). Throughout the paper, we assume that $M \in [0, 1]$.

The criterion of schedule optimality in our problem is to minimize the makespan, $C_{\max} := \max\{C_j : 1 \leqslant j \leqslant n\}$, where C_j is the completion time of job J_j. Extending the three-field notation (Graham,

☆ This manuscript was processed by Area Editor Maged M. Dessouky.

* Corresponding author. Tel.: +48 61 829 5334; fax: +48 61 829 5315.
E-mail address: stgawiej@amu.edu.pl (S. Gawiejnowicz).

0360-8352/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cie.2010.04.008

Lawler, Lenstra, & Rinnooy Kan, 1979), we will denote the problem as $Pm|p_{j,r} = p_j(M + (1-M)r^a)|C_{\max}$.

The contribution of the paper is threefold. First, we prove some basic properties of the problem. Next, on the basis of the properties, we propose for this problem two exact algorithms: a sequential branch-and-bound algorithm and a parallel branch-and-bound algorithm. To the best of our knowledge, no branch-and-bound algorithms for parallel-machine scheduling problems with a learning effect have been proposed earlier. We also present the results of experiments conducted on a computational cluster in order to evaluate the quality of schedules generated by our exact algorithms. Finally, we discuss the application of two greedy heuristic scheduling algorithms for the problem, comparing schedules generated by the heuristics with schedules generated by the branch-and-bound algorithms.

The remainder of the paper is organized as follows. In Section 2, we present a brief review of recent research on multi-machine scheduling with a learning effect. In Section 3, we prove basic properties of the considered problem. In Sections 4 and 5, we describe the sequential and the parallel branch-and-bound algorithm for the problem, respectively. In Section 6 we discuss two greedy heuristic scheduling algorithms for the problem. In Section 7, we present the results of computational experiments conducted on a computational cluster for both the exact algorithms and the greedy scheduling algorithms. We complete the paper by Section 8 with conclusions and remarks about future research.

2. Previous research

In this section, we briefly review the most recent literature concerning multi-machine scheduling problems with a learning effect and branch-and-bound algorithms proposed for problems of this type.

2.1. Multi-machine scheduling with a learning effect

Below we describe the main results concerning scheduling with a learning effect on parallel and dedicated machines. Since some of them are discussed in Biskup (2008), we mention only these that have been published after the review. We begin with parallel-machine scheduling problems.

Mosheiov (2008) has shown that the problem of minimizing the total absolute deviation of job completion times, $\sum_i\sum_j|C_i - C_j|$, subject to a general learning effect can be solved in a polynomial time.

Toksarı and Güner (2008) considered parallel-machine earliness/tardiness scheduling problem with simultaneous effects of learning and linear deterioration, sequence dependent setups and a common due-date for all jobs, and gave a mixed non-linear integer programming formulation of the problem.

Eren (2009a, 2009b) and Toksarı and Güner (2009a, 2009b) proposed a mathematical programming formulation of a parallel-machine scheduling problem with a learning effect of setup times and removal times, and the objective to minimize the weighted sum of total completion time and total tardiness, $\alpha\sum C_j + \beta\sum T_j$, where $T_j := \max\{C_j - d_j\}$, $\alpha > 0$ and $\beta > 0$.

All the papers concern job processing times with the log-linear learning effect, $p_{j,r} = p_j r^a$ for $1 \leqslant j \leqslant n$. Now, we pass to dedicated-machine scheduling problems with a learning effect.

Cheng, Wu, and Lee (2008a, 2008b) proposed polynomial-time optimal solutions for some special cases of multi-machine flow shop problems with the $C_{\max}$ and $\sum C_j$ criteria, in which the actual processing time of each job is a function of the total normal processing times of the jobs already processed and of the position of the job in a schedule.

Eren and Güner (2008) considered a two-machine flow shop problem with the log-linear learning effect and with the objective to minimize the weighted sum of total completion time, $\sum w_jC_j$, and the $C_{\max}$. In order to solve this problem, the authors proposed an integer programming formulation.

Wang (2008) has shown by a counter-example that some of results presented by Cheng, Sun, and Yu (2007) for a multi-machine flow shop with log-linear learning effect and the $C_{\max}$ criterion are not correct.

Xu, Sun, and Gong (2008) analyzed the worst-case behaviour of single-machine optimal job sequences applied to a multi-machine flow shop scheduling with a general learning effect and with the following three criteria: the $\sum w_jC_j$, the discounted total weighted completion time, $\sum w_j(1 - e^{-rC_j})$, and the sum of the quadratic job completion times, $\sum C_j^2$.

Lee and Wu (2009a, 2009b), Lee, Lai, and Wu (2010) and Yin, Xu, Sun, and Li (2009) have shown polynomial-time solvability of some multi-machine flow shop problems with the $C_{\max}$, the maximum lateness, $L_{\max} := \max\{C_j - d_j : 1 \leqslant j \leqslant n\}$, $\sum C_j$ and the $\sum w_jC_j$ criteria and with a learning effect that depends not only on the position of a job in a schedule but also on the processing times of the jobs already processed. Zhang and Yan (2010) obtained similar results for some flow shop scheduling problems with another form of a learning effect and with the $C_{\max}$, $\sum C_j$ and $L_{\max}$ criteria. Kuo and Yang (2010) have shown by a counter-example that some of the latter results are incorrect.

2.2. Branch-and-bound algorithms

In this section, we briefly review branch-and-bound algorithms proposed for multi-machine scheduling problems with a learning effect. All algorithms from this group concern two-machine flow shop problems, since no branch-and-bound algorithms for other multi-machine scheduling problems with a learning effect have been proposed.

Lee and Wu (2004) proposed a branch-and-bound algorithm for the $\sum C_j$ criterion. The algorithm was coded in Fortran 77, run on a Pentium 4 PC and tested on 300 instances with $n \leqslant 35$ jobs.

Chen, Wu, and Lee (2006) proposed a branch-and-bound for criterion $\lambda\sum C_j + (1-\lambda)T_{\max}$, where $\lambda \in (0,1)$, and tested it on 7200 instances with $n \leqslant 15$ jobs. Wu, Lee, and Wang (2007) proposed a branch-and-bound algorithm for the $T_{\max}$ criterion. The algorithm was tested on 6000 instances with $n \leqslant 14$ jobs.

Wu et al. (2007) and Wu and Lee (2009) proposed a branch-and-bound algorithm for the $\sum C_j$ criterion. These algorithms have been tested on instances with $n \leqslant 16$ jobs and two, three or five machines.

All the algorithms were coded in Fortran 90 and run on a Pentium 4 PC.

3. Problem properties

In this section, we prove some properties of the considered problem, which will be used in subsequent sections of the paper.

Problem $Pm|p_{j,r} = p_j(M + (1-M)r^a)|C_{\max}$ is a generalization of the ordinary $\mathcal{NP}$-hard problem $Pm||C_{\max}$.

Property 1. *Problem $Pm|p_{j,r} = p_j(M + (1-M)r^a)|C_{max}$ is ordinary $\mathcal{NP}$-hard.*

A general version of the problem, $P|p_{j,r} = p_j(M + (1-M)r^a)|C_{\max}$, is a generalization of the strong $\mathcal{NP}$-hard problem $P||C_{\max}$.

Property 2. *Problem $P|p_{j,r} = p_j(M + (1-M)r^a)|C_{max}$ is strongly $\mathcal{NP}$-hard.*

Since the processing time of a particular job J_j depends only on its number in sequence and its initial processing time, we obtain two next properties.