Computers & Industrial Engineering 59 (2010) 537–543

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

# Minimizing the makespan in the non-preemptive job-shop scheduling with limited machine availability

Yazid Mati

*Al-Qassim University, College of Business & Economics, Almelaida, P.O. Box 6633, Saudi Arabia*

ARTICLE INFO

*Article history:*
Received 8 November 2009
Received in revised form 17 June 2010
Accepted 19 June 2010
Available online 1 July 2010

*Keywords:*
Job-shop scheduling
Availability constraints
Makespan
Disjunctive graph
Taboo thresholding

ABSTRACT

This paper addresses the makespan minimization in a job-shop environment where the machines are not available during the whole planning horizon. The disjunctive graph model is used to represent the schedules and the concept of blocks is generalized to include the unavailability periods of machines. To solve the problem, we develop a taboo thresholding heuristic that uses a new block-based neighborhood function. Some sufficient conditions to eliminate the evaluation of non-improving moves are proposed. Experiments performed on existing problem instances of the literature show the efficiency of the proposed heuristic.



## 1. Introduction

Current trends in scheduling literature are toward integrating practical constraints in traditional scheduling models. One of these constraints that attracted researchers during the last decade is the unavailability of machines (Ma, Chu, & Zuo, 2010). In practice, machines are not available during the whole planning horizon because of breakdowns that may happen. Moreover, machines can be voluntarily stopped to perform some maintenance tasks such as washing or control operations.

The machine availability constraints encountered in production systems can either be fixed or non-fixed. In the fixed type, the starting times of unavailability periods (or holes) are known in advance while they are flexible in the second type and must be determined by the production scheduling procedure. The non-fixed type is generally used to implement preventive maintenances. Unavailability periods can also be classified depending on whether they allow or not the preemption of operations, i.e. resumable, semi-resumable or non-resumable (Lee, 1997, 1999), non-preemptive (Aggoune & Portmann, 2006), crossable or non-crossable (Mauguière, Billaut, & Bouquard, 2005).

We investigate in this paper the fixed and non-preemptive unavailability periods in a job-shop environment with $n$ jobs and $m$ machines in order to minimize the makespan. Each job $J_i$ consists of a linear sequence of $m$ operations. Each machine $M_k$ can perform at most one operation at a time and each operation $x$ of job $J_i$ needs one machine during $p_x$ time-units. Several unavailability periods $H_{k,j} = [l_{k,j}, u_{k,j}]$ may occur on each machine $M_k$, $k = 1,\ldots,m$; $j = 1,\ldots,s_k$, where $s_k$ is the number of unavailability periods of machine $M_k$. The starting and finishing times of these periods are known in advance and fixed. Operations are strictly non-preemptive, which means that their execution can be interrupted neither by unavailability periods, nor by other operations.

This paper proposes a simple and efficient local search heuristic for the job-shop problem with availability constraints, based on known methods solving the classical job-shop problem. The remainder of the paper is organized as follows. Section 2 summarizes the previous studies on shop scheduling problems with machine availability. Section 3 first recalls the disjunctive graph model for representing the schedules and then presents some properties of this graph. The main contribution of the paper is presented in Section 4 in which a taboo thresholding procedure is proposed to solve the problem. Comparisons of the proposed heuristic with existing approaches are shown in Section 5. Section 6 concludes the paper and gives some research directions.

## 2. Review of previous researches

This section gives a brief overview of the previous studies on shop scheduling problems with unavailability periods. We first summarize the studies on flow-shop and hybrid flow-shop then we present works dealing with job-shop and flexible job-shop.

The majority of the studies on flow-shop environment deal with the two-machine case. These studies are originated by the work of

*E-mail address:* matie@qu.edu.sa

0360-8352/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cie.2010.06.010

Lee (1997) who proves that the two-machine flow-shop scheduling problem with resumable unavailability periods is NP-hard when a single unavailability period is imposed on only one machine. He also proposes heuristics with worst-case error bounds. Since this work several heuristics with improved bounds have been proposed in the literature (see for instance Breit (2004)). For the semi-resumable case, a dynamic programming algorithm as well as heuristics with worst-case error bounds are proposed in Lee (1999). Another dynamic programming algorithm is proposed in Allaoui, Artiba, Elmaghraby, and Riane (2006) to deal with the non-resumable case when there is a single unavailability period on the first machine. The two-machine flow-shop problem has been also investigated in the no-wait context in Cheng and Liu (2003) and Espinouse et al. (2001).

The flow-shop problem with multiple machines is studied in Aggoune (2004) to minimize the makespan. An insertion heuristic that builds active schedules is proposed. In Aggoune and Portmann (2006) the authors first propose a polynomial algorithm to solve the special case of two jobs. Based on this algorithm they propose a greedy heuristic to solve the problem with more than two jobs. The greedy heuristic schedules the jobs two per two according to an input job sequence. Computational experiments proved that the greedy heuristic outperforms the insertion heuristic proposed in Aggoune (2004).

We now recall the literature on the hybrid flow-shop. An approach coupling simulation and optimization is proposed in Allaoui and Artiba (2004) to optimize various regular criteria in the general case with $m$ stages. The special case of two stages with one machine on the first stage and $m$ machines on the second stage is investigated in Allaoui and Artiba (2006) for the non-resumable case. A branch-and-bound algorithm is proposed and the performance of three heuristics is investigated. The resumable case in a two-stage problem is studied in Xie and Wang (2005) where there is only one unavailability period on the single machine of the first stage and no unavailability periods on the $m$ machines of the second stage.

Surprisingly, the literature on the job-shop problem with limited machine availability is not abundant. A polynomial algorithm for the particular case with two jobs is proposed in Aggoune (2002) for the non-preemptive case. The polynomial algorithm is used to develop a heuristic for solving the general case with multiple jobs. In Mauguière et al. (2005), branch-and-bound algorithms are proposed to solve the job-shop problem with various types of unavailability periods. Two mathematical models that include all possible cases of unavailability periods are proposed in Azem, Aggoune, and Dauzère-Pérès (2007) for the job-shop problem. The flexible job-shop problem is studied in Gao, Gen, and Sun (2006) Zribi, El Kamel, and Borne (2008). In the former paper, the non-fixed unavailability periods is tackled using genetic algorithms. The later paper considers the fixed case and proposes a hierarchical approach that first solves the assignment problem and then the sequencing problem.

In conclusion, there is an increasing interest in integrating unavailability periods in scheduling problems. However, the majority of existing studies concentrate on special cases with only one unavailability period, and do not investigate general problems such as job-shop, hybrid flow-shop and flexible job-shop. This paper aims at solving efficiently the job-shop problem with non-preemptive unavailability periods.

## 3. Problem representation and properties

The disjunctive graph model, introduced by Roy and Sussmann (1969) for the classical job-shop problem, is used to represent feasible schedules of the studied problem without explicitly integrating the unavailability periods of machines. These periods are taken into account during the computation of the longest paths. The set of nodes of the graph is associated to operations of jobs plus a source node that represents the beginning of each job, and a sink node which corresponds to the end of the schedule. Constraints of the scheduling problem are modeled by a set of arcs including conjunctive and disjunctive arcs. A conjunctive arc connects two consecutive operations of a job and represents the precedence constraints. A disjunctive arc connects two operations performed on the same machine and implies that these operations cannot be executed concurrently. The source node is connected via a conjunctive arc to the first operation of each job while the last operation of each job is connected via a conjunctive arc to the sink node. The length of an arc from the source node to the first operation of each job is equal to 0 (jobs are ready from time 0), and the length of the remaining arcs is equal to the processing time of the operation from which they start.

Fig. 1a gives the disjunctive graph for the following problem instance with two jobs and three machines: $J_1 = M_3(7)M_2(2)M_1(3)$ and $J_2 = M_1(2)M_3(5)M_2(4)$.

A selection is obtained by fixing a direction to each disjunctive arc, i.e. replacing each disjunctive arc by a conjunctive arc. Each selection corresponds to a schedule of the job-shop problem with limited machine availability. A selection is feasible if the induced conjunctive graph is circuit-free. The head $h_x$ of a node $x$ is equal to the length of a longest path from the source node to $x$. Consequently, the value of the makespan corresponds to the length of a longest path from the source node to the sink node.

Fig. 1b gives a feasible selection in which the sequence on machines are the following: $M_1$:$J_2 \rightarrow J_1$, $M_2$:$J_1 \rightarrow J_2$ and $M_3$:$J_1 \rightarrow J_2$.

### 3.1. Longest path calculation

Since the unavailability periods of machines are not explicitly integrated in the disjunctive graph, the Bellman's labeling algorithm needs to be adapted to integrate these periods. Let us consider a node $x$ representing a given operation that is performed on machine $M_k$ and denote by $h_x$ its head, by $pr_x$ the node preceding $x$ on the routing and by $ps_x$ the node preceding $x$ on machine $M_k$. The head $h_x$ of $x$ is computed in $O(K)$, where $K$ is the number of unavailability periods of $M_k$, using the pseudo-code given in Algorithm 1. The algorithm first calculates the earliest starting time $h_x$ using the node $pr_x$ and the node $ps_x$ ($h_{ps_x} = -\infty$ if $ps_x$ does not exist). Then, the algorithm searches, by scanning the unavailability periods $H_{k,j} = [l_{k,j}, u_{k,j}]$ of $M_k$ in increasing order of $l_{k,j}$ the first time satisfying the inequality $h_x + p_x \leqslant l_{k,j}$, which means that operation $x$ can be started and finished without preemption.

**Algorithm 1.** Head of a node $x$

$h_x = \max\{h_{pr_x} + p_{pr_x}, h_{ps_x} + p_{ps_x}\}$
**for** each unavailability $H_{k,j} = [l_{k,j}, u_{k,j}]$ of $M_k$ **do**
  **if** $h_x < l_{k,j}$ **then**
    **if** $h_x + p_x \leqslant l_{k,j}$ **then** break
    **else** $h_x \leftarrow u_{k,j}$
  **else**
    **if** $h_x \leqslant u_{k,j}$ **then** $h_x \leftarrow u_{k,j}$
**end for**

### 3.2. An extended concept of blocks

Let us consider a longest path from the source node to the sink node. This longest path can be decomposed in two types of blocks. The first type, that we call *classical block*, corresponds to a maximal succession of operations without idle time on the same machine