Computers & Operations Research 60 (2015) 111–120

Contents lists available at ScienceDirect

Computers & Operations Research

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

A Discrete Inter-Species Cuckoo Search for flowshop scheduling problems

Preetam Dasgupta [a], Swagatam Das [b,*]

[a] Electronics and Telecommunication Engineering Jadavpur University, Kolkata, India
[b] Electronics and Communication Sciences Indian Statistical Institute, Kolkata, India

ARTICLE INFO

Available online 17 February 2015

Keywords:
Hybrid Flow-shop scheduling (HFS)
Heuristics
Discrete Inter-Species Cuckoo Search (DISCS)
Makespan
Mean flow time
Permutation Flow-shop Scheduling Problems (PESP)

ABSTRACT

This paper presents a discrete version of the Inter-Species Cuckoo Search (ISCS) algorithm and illustrates its use for solving two significant types of the flow-shop scheduling problems. These are Hybrid Flow-shop Scheduling (HFS) and Permutation Flow-shop Sequencing Problems (PFSP). Hybrid flowshop scheduling problems are a generalization of flowshops having parallel machines in some stages and these problems are known to be NP-hard. A heuristic rule called the Smallest Position Value (SPV) is used to enable the continuous inter-species cuckoo search to be applied to most types of sequencing problems. Makespan and mean flow time are the objective functions considered and computational experiments are carried out to compare the proposed Discrete Inter-Species Cuckoo Search (DISCS) with other state-of-the-art meta-heuristic algorithms. Experimental results confirm the superiority of DISCS with respect to many other existing metaheuristic search algorithms.

1. Introduction

Scheduling is considered to be a method of allocating resources to perform a collection of tasks which may or may not be directly interdependent [1]. Hybrid Flow-shop Scheduling (HFS) problem is an important scheduling as many real industries resemble the hybrid flow-shop situations and many researchers concentrated on these problems which were first proposed by Arthanari and Ramamurthy [2]. The Permutation Flow-shop Sequencing Problem (PFSP) determines the order of processing jobs over machines in order to optimize certain performance measures, mostly makespan and mean flow time, when all the jobs have the same machine sequence. These types of scheduling problems were proposed in [3] and since then they have attracted a lot of research attentions.

The minimization of makespan and mean flow time leads to the optimization of total production run, better utilization of resources, and minimization of work-in-process (WIP) inventory [4]. As for the computational complexity involved, both the makespan minimization and flow time minimization turned out to be NP-complete problems [5,6]. Hybrid Flow-shop Scheduling problem is considered to be more complex than other flow-shop scheduling problems as HFS is a combination of flow-shop and parallel machine environment and HFS problems were proved to be of the type of NP-hard [7,8]. Thus, these problems cannot be solved to absolute limit by the exact algorithms and efforts have been devoted to find high-quality solutions by heuristic and meta-heuristic optimization methods instead finding an optimal solution by deterministic methods.

Heuristics for the makespan minimization problem have been proposed by Palmer [9], Campbell et al. [10], Dannenbring [11], Nawaz et al. [12], Taillard [13], Framinan et al. [14] and Framinan and Leisten [15]. Modern metaheuristics have been presented for the flow shop scheduling with makespan minimization such as Simulated Annealing (SA) [16], tabu search [17,18], Genetic Algorithms (GAs) [19,20], Ant Colony Optimization (ACO) [21,22], Particle Swarm Optimizer (PSO) [23], and local search methods [24,25]. In order to test the performance of heuristics, a120 benchmark instances as presented by Taillard in [26] are mostly used in all these modern heuristic algorithms.

In case of flow time minimization, several heuristic optimization methods have been proposed over the past few decades, see for example [27–40]. A comprehensive review and computational evaluation of 22 existing heuristics in this context have been recently published by Pan and Ruiz [41].

Rajendran and Chaudhuri [6] proposed a well-known heuristics to minimize total flow time for flow-shop problems appearing in multi-stage parallel processor. Different heuristics were developed by the researchers with various cost functions [42,43]. HFS problems were solved to a large extent by Tang and Wang [45] where they applied the tabu search algorithm. Genetic Algorithm (GA) is a widely used metaheuristics algorithm for solving the HFS problem [46,47]. ACO [48], PSO algorithm [49] were applied to solve the HFS problem to a large degree. Babayan and He [44] put forward an agent based

* Corresponding author.
E-mail addresses: dasguptapreetam@gmail.com (P. Dasgupta), swagatam.das@isical.ac.in (S. Das).

http://dx.doi.org/10.1016/j.cor.2015.01.005
0305-0548/

scheduling incorporated with game theory for minimizing the makespan to solve the n job 3 stage flexible flowshop scheduling problems. Jungwattanakit [50] undertook a comparison of three distinct meta-heuristic algorithms. These were GA, tabu search, and SA. A combination of these was used to minimize the convex sum of makespan and the number of tardy jobs for flexible flowshop problems with parallel machines which were mostly unrelated. Vazquez-Rodriguez and Ruiz [51] provided another comprehensive review on the HFS kind of problems.

Cuckoo search via Lévy flights [54] is a meta-heuristic algorithm based on the parasitic breeding behavior of certain species of cuckoo and the presence of Lévy flights in such reproduction strategy. Recently Cuckoo Search (CS) has attracted the attention of a vast majority of researchers from different application fields and domain; see for example [55–57]. Till now CS did not have vast number of applications in discrete or combinatorial optimization problems, but a very significant work has been published by Quaarab et al. in [57] where the basic CS is used to solve the traveling salesman problem. CS has also found a recent and significant application to the NP hard annual crop-planning problem in [58]. Recently a local search based CS has been used to solve the permutation flowshop scheduling problems.

This paper presents the discrete version of a novel variant of the CS algorithm referred as the Inter-Species Cuckoo Search (ISCS) [59]. The discrete version of ISCS is developed to minimize the makespan and mean flow time in both Hybrid Flow-shop Scheduling (HFS) and Permutation Flow-Shop Sequencing Problems (PFSP).

The rest of the paper is organized as follows. The flow-shop scheduling problems have been formally presented in Section 2. The basic cuckoo search algorithm is outlined in Section 3. Section 4 presents and discusses the inter-species cuckoo search. The proposed Discrete Inter-Species Cuckoo Search (DISCS) algorithm will is discussed in Section 5. The experimental results are provided and discussed in Section 6. Finally, Section 7 concludes the paper.

2. Problem formulation

In this paper the proposed DISCS algorithm is applied in two types of flow-shop scheduling problems. These are

- Hybrid Flow-shop Scheduling (HFS).
- Permutation Flow-shop Scheduling Problem (PFSP).

The basics of these two problems are explained as,

2.1. Permutation Flow-shop Scheduling Problem (PFSP)

In PFSP, given the processing times p_{jk} for job j and machine k, and a job permutation $\pi = \{\pi_1, \pi_2, ..., \pi_n\}$ where n jobs (j=1, 2, …, n) will be sequenced through m machines (k=1, 2, …, m), then the problem is to the find the best permutation of jobs to be valid for each machine. For $n/m/P/C_{\max}$ problem, $C(\pi_j, m)$ signifies the completion time of job π_j on machine m. Given the job permutation $\pi = \{\pi_1, \pi_2, ..., \pi_n\}$, the calculation of finish time for the n-job m-machine problem is given as follows:

$$C(\pi_1, 1) = p_{\pi_1,1}, \tag{1}$$

$$C(\pi_j, 1) = C(\pi_{j-1}, 1) + p_{\pi_j,1}, \quad j = 2, \ldots, n, \tag{2}$$

$$C(\pi_1, k) = C(\pi_1, k-1) + p_{\pi_1,k}, \quad k = 2, m, \tag{3}$$

$$C(\pi_j, k) = \max\{C(\pi_{j-1}, k), C(\pi_j, k-1) + p_{\pi_j,k}\}, \tag{4}$$

for j=2, …, n; k=2, …,m.

Then the makespan can be defined as

$$C_{\max}(\pi) = C(\pi_n, m). \tag{5}$$

Thus, the PFSP with the makespan criterion is to find a permutation π^* in the set of all permutations Π such that

$$C_{\max}(\pi^*) \le C(\pi_n, m) \quad \forall \pi \in \Pi. \tag{6}$$

Let $F(\pi_j)$ denote the flow-time of job π_j and is equivalent to the completion time of job π_j on machine m since ready times are zero. We can calculate the total flow-time $TFT(\pi)$ of a permutation π by adding all the flow times or completion times of the jobs. Then total flow-time can be defined as

$$TFT(\pi) = \sum_{j=1}^{n} F(\pi_j) = \sum_{j=1}^{n} C(\pi_j, m). \tag{7}$$

Therefore, the PFSP with total flow-time criterion boils down to the determination of a permutation π^* in the set of all permutations Π such that

$$TFT(\pi^*) \le TFT(\pi) \quad \forall \pi \in \Pi. \tag{8}$$

2.2. Hybrid Flow-shop Scheduling (HFS)

Hybrid flow-shop scheduling consists of a series of production stages and each stage has multiple machines operating in parallel. Some stages may have only one machine, but it is mandatory that at least one stage have more than one machines. Each job is processed by one machine in each stage and these machines may be identical, uniform or unrelated. The layout is shown in Fig. 1.

The HFS problems are NP-hard problems and for these problems the objective function to be minimized is the weighted sum of makespan and mean flow time. Makespan may be defined as the time consumed by the most recently finished job to leave the system and the average time spent in the system is known as mean flow-time.

The mathematical model of the problem is shown below.

$$Z = Min(w_1 C_{\max} + w_2 f) \tag{9}$$

where

$$C_{\max} \ge C_{js}, \quad \text{for all } s = 1, 2, \ldots, M \text{ and } j = 1, 2, \ldots, n, \tag{10}$$

$$C_{js} = S_{js} + P_{sj}, \tag{11}$$

$$\sum_{i=1}^{m_s} Y_{jis} = 1, \quad \text{for all } s = 1, 2, \ldots, M \text{ and } j = 1, 2, \ldots, n, \tag{12}$$

$$C_{js} \le S_{j(s+1)}, \quad \text{for all } s = 1, 2, \ldots, M-1 \tag{13}$$

$$S_{hs} \ge C_{js} - KW_{hjs}, \quad \text{for all job pairs}(h, j), \tag{14}$$

$$S_{js} \ge C_{hs} - K - 1, \quad \text{for all job pairs}(h, j), \tag{15}$$

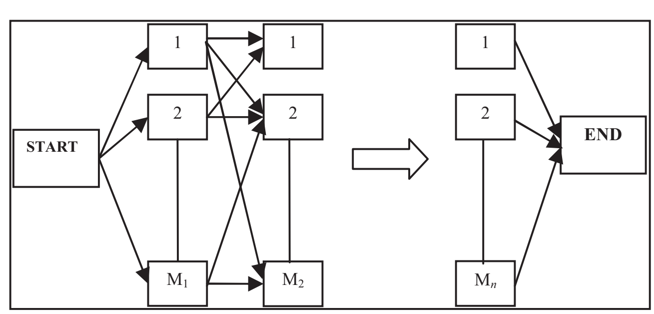

Fig. 1. Layout of an M-stage Hybrid Flow-shop Scheduling instance (figure adopted from [52]).