Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 167 (2020) 1120–1127

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Computational Intelligence and Data Science (ICCIDS 2019)

Modelling of heuristic distribution algorithm to optimize flexible production scheduling in Indian industry

Guduru Ramakrishna Reddy[a,*], Harpreet Singh[b], Aurelijus Domeika[a], Nallapaneni Manoj Kumar[c], Ma Quanjin[d,e]

[a] *Institute of Mechatronics, Faculty of Mechanical Engineering & Design, Kaunas University of Technology, K. Donelaicio g. 73, Lithuania*
[b] *I.K. Gujral Punjab Technical University, Jalandhar-144603, Punjab, India*
[c] *School of Energy and Environment, City University of Hong Kong, Kowloon, Hong Kong*
[d] *Faculty of Mechanical and Manufacturing Engineering, Universiti Malaysia Pahang, 26600 Pekan, Pahang, Malaysia*
[e] *School of Mechanical Engineering, Ningxia University, 750021 Yinchuan, China*

Abstract

Multi-objective scheduling with the NP-dependent relay preparation time becomes difficult because the complexity of the optimization increases within a reasonable time. Research methods have become a more important option to solve the difficult problems of NP because there are more powerful solutions and a great potential to require biology in a reasonable time. In the present work, Two Heuristic Algorithms are modelled and the best algorithm among those two Heuristics is selected after few comparisons 3M to 5M, this can optimize the scheduling processes up to 10x10 jobs i.e. 10 machines and 10 jobs. In context of Heuristic optimization, the results clearly show the variation in times (decrease) of all-time dependents i.e. 46% decrease, when the increase in machines and jobs are considered, therefore, it implicates the error of 0.468 as the make-span decreased by 221 minutes. The proposed model gives a large edge in minimization of make-span i.e., 40-50% decrease in the production times, and it can produce even more when the number of sources and jobs are more. Therefore, the optimized error of 0.456 than the mathematical data and hence, this model is validated.

Keywords: Scheduling; heuristics; ant colony optimization; genetic algorithm; branch and bound.

* Corresponding author. Tel.: +91-9550546205
E-mail address: rama.guduru@ktu.edu

1877-0509 © 2020 The Authors. Published by Elsevier B.V.

10.1016/j.procs.2020.03.414

1. Introduction

It is generally accepted that finding optimization of difficult problems with NP is not a viable option, since it takes a long time for the calculation to estimate these solutions. Many researchers have suggested lower algorithms to schedule a workshop to improve the many performances of Johnson's pioneering work (1954). In fact, a good initial solution can be obtained through the course of events within a reasonable time frame. Heuristic algorithms can be broadly categorized in the rules of transmission, inference, construction and optimization. [1]. Despatching rules are the most classical and well-known methods to build a schedule. These rules often used in practice but mostly used for obtaining initial sequence in some improvement heuristics and metaheuristics. Constructive heuristics build a schedule from the scrape by making a series of passes through the list of unscheduled jobs where at each pass one or more jobs are chosen and added to the schedule [2] [3]. Contrary to constructive heuristics, improvement heuristics start from an accessible solution and apply some improvement procedure [4].

In the present research, it is firstly selected the two heuristic algorithms based on previous work one is a search algorithm and the other is normal dependent algorithm [6] [7]. The goal of the present exploration is to minimize the make-span of a job shop scheduling issue by utilizing a heuristic calculation. For this, we have chosen a case with 3 machines and 4 jobs. Where the 4 jobs must be handled by the 3 machines in a way such that it expends least time for fruition of all the jobs. The 3 machines and 4 jobs are assumed as:

Table 1. Model Problem of the Job Shop Scheduling

Jobs/Machines	Machine1	Machine 2	Machine 3
Job 1	J_{11}	J_{12}	J_{13}
Job 2	J_{21}	J_{22}	J_{23}
Job 3	J_{31}	J_{32}	J_{33}
Job 4	J_{41}	J_{42}	J_{43}

The Mathematical model of the problem is as follows:

J_{MN} – Job M processing in Machine N.

M= {1,2,3} and N= {1,2,3,4}

Let Make-span – Z

Objective Function for Minimize (Make-span [Z])

The aim of the present research is to Model a Heuristic Algorithm for a scheduling problem to optimize the Make-span. Later, the results of various algorithms are compared.

2. Problem formulation

In fact, a good initial solution can be obtained through the course of events within a reasonable time frame. Heuristic algorithms can be broadly classified in the rules of transmission, inference, construction and optimization [5]. The rules of transmission are the best known and known methods for constructing a schedule.

2.1 Assumptions

- All the jobs and machines are accessible at time Zero.
- Preventive action is not permitted.
- An operation started on the machine must be finished before the start of another operation.
- Machines are accessible all through the period.
- All preparing time on the machine are known, deterministic, limited and free of succession of the jobs to be handled.

- Each machine is ceaselessly accessible for task, without critical division of the scale into movements or days and without thought of provisional inaccessibility, for example, breakdown or support.
- The first machine is thought to be prepared whichever and whatever job is to be handled on it first.
- Machines might be unmoving.
- Each job is prepared through each of the m machines once and just once. Besides, a job does not get to be accessible to the following machine until and unless preparing on the present machine is finished i.e. part of job or job cancelation is not permitted.
- In-process stock is permitted. In the event that the following machine on the arrangement required by a job is not accessible, the job can hold up and joins the line at that machine.

2.2 Considerations

Each resource has multiple processes that must be processed for a specific period. These functions must be programmed in a way to minimize them. The works are subject to the following restrictions:

- No-Wait.
- No two jobs must be processed by the same machine.
- No two machines should perform same job at given time.

3. Modified Heuristic Methods

In the present research we modified two heuristic methods for minimizing the make-span in job shop scheduling problem. The algorithms are Ant Colony Optimization Algorithm and Branch and Bound Algorithm.

3.1 Ant Colony Method of Optimization

This method was developed by Dorigo and his colleagues in the early 1990s. The optimization process of ant colony can be explained by presenting the problem of improvement as a multilayer chart. The number of design variables and the number of nodes in a given layer is equal to the number of individual values allowed for the corresponding design variable. Therefore, each node is associated with a special value allowed for the design variable. The ACO process can be explained as follows. Let the colony consist of an ant. Ants start at the main node and pass through different layers from the first layer to the end or the last layer and end at the destination node in each cycle or repetition. The gradual process is described below: -

i. Initially, the number of ants is to be selected and the number of paths available are to be assumed. Let, Number of Ants is ‘N’ and Number of paths be ‘Z’
ii. The Ant Behaviour is to be determined by the probability function. If an ant ‘k’ is assumed to be used pheromone of ‘ τ_{ij} ’. The ant can select any of the available paths initially randomly. This can be calculated by using the probability function given below: -

$$p_{ij}^{k} = \begin{cases} \tau_{ij}^{\alpha} \Big/ \sum_{j \in N_i^k} & \text{if } j \in N_i^k \\ 0 & \text{if } j \notin N_i^k \end{cases} \tag{1}$$

   α- Need of Pheromone
   N_i^k - Neighbour nodes of ant ‘k’ when it is in path
   τ_{ij} - Pheromone Rate

iii. Once the ants start the journey, they reach the destination and returns back to the home or nest. In the return journey the ants will release some amount of pheromone in the paths. For this purpose, we have to update the pheromone amount. This can be done by the following equation

$$\tau_{ij}^{(updated)} \leftarrow \tau_{ij}^{old} + \Delta\tau^{k} \tag{2}$$

iv. While the journey the ants will move from one node to another by leaving pheromone on their path. The pheromone released by the ants will be evaporated by time. The present case for calculating the effect of pheromone in selecting the particular route by ant we have to calculate the rate of evaporation of pheromone

$$\tau_{ij} \neg (1\text{-}p)\,\tau_{ij};\ "(i,j)\ \hat{I}A \tag{3}$$

v. When all the ants reach the nest or home there will be the change in the pheromone through the path. The number of ants increases through the particular path the pheromone in that path will be more, So the pheromone rate should be updated

$$\tau_{ij} = (1\text{-}p)\,\tau_{ij} + \sum_{k=1}^{N} V\,\tau_{ij}^{k} \tag{4}$$

vi. Now to analyse the best path we have to find the worst and best paths from the calculated values. The formula to be used for finding the best paths is

$$V\,\tau_{ij}^{k} = \begin{cases} \frac{ç\, f_{best}}{f_{worst}}; & \text{if } (i,j)\ \hat{I} \text{ best tour} \\ 0; & \text{Otherwise} \end{cases} \tag{5}$$

vii. After the first iteration the ants will repeat the same by selecting various paths. The process will be terminated if it reaches the maximum iteration or better solution is not found.

3.2 Branch and Bound

In present work we modified the existing method for scheduling purpose. The main part of modification is for branching and bounding at same time for each job or operation. This mainly involves in following steps, In the primary stage, we have to analyse the data available such as number of machines and the jobs available. In this, we assumed 3 machines and 4 jobs. Let, the Machines – M_1, M_2, M_3 and Jobs - J_1, J_2, J_3, J_4. The branch and bound technique optimize the problem by branching and bounding at each step by creating nodes at each branch. The nodes will be created when the problem has various sub objectives. In this we assume nodes as 'N'. Some assumptions for this model are

q- Completion time of last job on each machine
σ- Partial permutation
b- Lower bound on individual machine

Table 2. Various Jobs Associated with Machines for Input

Jobs/Machines	Machine1	Machine 2	Machine 3
Job 1	J_{11}	J_{12}	J_{13}
Job 2	J_{21}	J_{22}	J_{23}
Job 3	J_{31}	J_{32}	J_{33}
Job 4	J_{41}	J_{42}	J_{43}

Now from the above problem the nodes can be created as, B- Final lower bound for σ

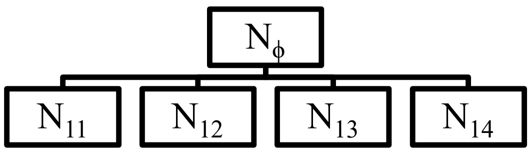

Fig.1 presentation of Initial Nodes of B&B Technique

Now, we have to calculate for nodes. First for node 1 i.e. N_{11}. Initially, $\sigma = 1$, $\sigma^1 = \{0,0,0\}$, Calculate the completion times for first node q_1, q_2, q_3. This can be done by using formula $q_1 = (J_{11}+ J_{12}+J_{13})$, $q_2 = (J_{21}+ J_{22}+J_{23})$ and $q_3 = (J_{31}+ J_{32}+J_{33})$. After finding the completion time the main step is to find the lower bound to minimize the make-span. The lower bound can be find for individual machine by the expressions formulated. These expressions will change with the increase in number of machines as these are dependent on each other. The equations are given below:

$$b_1 = q_1 + \sum_{i \in \sigma^1} c_{i1} + \underset{i \in \sigma^1}{Min} \{ c_{i2} + c_{i3} \} \tag{6}$$

In the same way we have to calculate the lower bound values for remaining machines

$$b_2 = q_2 + \sum_{i \in \sigma^1} c_{i2} + \underset{i \in \sigma^1}{Min} \{ c_3 \} \tag{7}$$

and

$$b_3 q_3 + \sum_{i \in \sigma^1} c_{i3} \tag{8}$$

If the no of jobs and machines are more the number of lower bounds for individual machines increases accordingly. It has the lower bound values for all the individual machines ready. Now we have to evaluate the final lower bound for the present node at σ. The final lower bound is the maximum of the all individual lower bounds. This is given as

$$B = Max(b_1, b_2, b_3) \tag{9}$$

Where, B - Final lower bound, the same procedure is to be continued for all the nodes for finding the completion time for all machines and the lower bounds of all machines and jobs. After finding all the final lower bounds for all σ. It is to find the lower bound for each branch then consider the node with minimum B value of all nodes for further branching. There again we have to start the branching from minimum B value node. Final stage when further branching is not possible the algorithm has to be concluded by finding the schedule and time to perform the jobs. We will consider the sequence which gives the minimum time for completion of the job. Let us consider an example of branching from the above diagram for better understanding.

4. Results and Discussion

4.1 Branch and bound technique as system implementation

Branch and Bound methods have been developed to find precise solutions to work scheduling problems. The most popular conclusion is based on the practice of priority rules. Others are more sophisticated. Among them, there is a method for Adams and others. She was very successful. A good conclusion is also important in terms of related branches and methods. The course of events has not yet been developed with guaranteed performance. For most of the considerations, there are cases in which this conclusion leads. In the present work, it is considered a model problem for formulation and development of the branch and bound technique. The problem considered is of 3 machines and 4 jobs. We applied this primarily to show the strength of the proposed model and variation of results among the other models. The Problem statement is shown in Table 4.

Table 3. Input Data for Formulating Analytical Output

Jobs/Machines	Machine1	Machine 2	Machine 3
Job 1	10	10	11
Job 2	11	11	14
Job 3	12	13	11
Job 4	14	9	16

For the above given objective, we have solved it analytically to select the best algorithm for continuing our research. The algorithm used branch and bound technique and shows the following results. The lower bound initially is set to be zero (B=0) and root node is N_ϕ. The nodes can be branched as, the initial node is branched into four sub nodes. This will be further branched depending on the completion calculated at this stage.

a. In this stage, the process has to set the permutation occurrence and the initial completion times. It is to consider them as zero initially. The branching of present node is shown in Figure 2. Now, the initial assumptions are

Permutation occurrence(σ) = 1 and σ^1= {2, 3, 4}

Completion times (q_1, q_2, q_3) = (0, 0, 0)

b. Now, it has to calculate the Completion Times of individual machines w.r.t jobs at various nodes

At Node N_1, (q_1, q_2, q_3) = (10, 20, 31)

At Node N_2, (q_1, q_2, q_3) = (11, 22, 36)

At Node N_3, (q_1, q_2, q_3) = (12, 25, 36)

At Node N_4, (q_1, q_2, q_3) = (14, 23, 39)

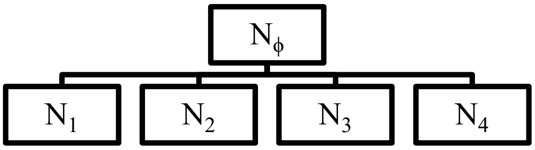

Fig.2 Nodes at Initial Stage of B&B Technique

c. After finding the completion times for all machines at all nodes now to find the lower bound values at each node. This value can be calculated by using the formulas mentioned above

At Node N_1, (b_1, b_2, b_3) = (71, 64, 72)

At Node N_2, (b_1, b_2, b_3) = (68, 65, 74)

At Node N_3, (b_1, b_2, b_3) = (68, 66, 77)

At Node N_4, (b_1, b_2, b_3) = (68, 68, 75)

The final lower bound for the nodes calculated will be, B = {B_1, B_2, B_3, B_4} = {72, 74, 77, 75}

d. It is to select the branching node from above values, as the value B_1 having minimum value and it will be the further branching node. Now it should be branched. The branching takes place according to the previous branching data. Now, the same procedure repeats for finding Completion times and lower bounds for each machine. From the calculated individual lower bound values, to determine the final lower bound values. Where, the values are B = {B_{12}, B_{13}, B_{14}} = {73, 76, 74}. These are the final lower bound values associated with the second node branching, which can be further branched at node B_{12} as it is the minimum of all the nodes determined. Now the final node, which is branched at B_{12} as depicted in Figure 3. In this final branching after finding completion time and lower bound values the final values of lower bound are

B = {B_{123}, B_{124}} = {73, 73}

e. Finally, the results have to be analysed, for that considered a table for better understanding the variation in results as shown in Table 4. Now the minimum values are **72** and **73**. From this we can find the best schedule which is associated with less make span is

$1 \rightarrow 2 \rightarrow 3 \rightarrow 4$ and $1 \rightarrow 2 \rightarrow 4 \rightarrow 3$

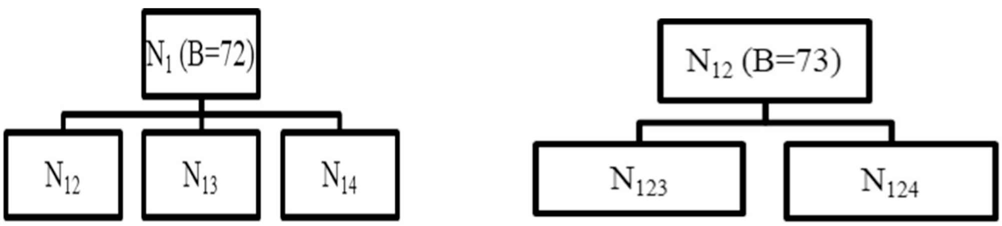

Fig.3 Nodes of B&B Technique at Second and Third Stages Branching

f. Finally, the best schedule and the minimum time associated with it is calculated. Hence it is clear that the algorithm is the good alternative for the other optimization techniques.

Table 4. Results for Branch and Bound Problem

Permutation Occurrence (σ)	Completion Times (q_1, q_2, q_3)	Lower Bounds (b_1, b_2, b_3)	Final Lower Bounds B
1	(10, 20, 31)	(71, 64, 72)	**72**
2	(11, 22, 36)	(68, 65, 74)	74
3	(12, 25, 36)	(68, 66, 77)	77
4	(14, 23, 39)	(68, 68, 75)	75
12	(21, 32, 46)	(71, 65, 73)	**73**
13	(22, 35, 46)	(72, 69, 76)	76
14	(24, 33, 49)	(71, 68, 74)	74
123	(33, 46, 57)	(72, 71, 73)	**73**
124	(35, 44, 62)	(71, 68, 73)	**73**

4.2 Comparison of Results

After consideration of data and evaluation of results in Figure 4, it is clear that the proposed model is dominating over the others in all cases. The results in blue colour are the actual data from industry, In red is ant colony optimization and green colour is the proposed BB Algorithm.

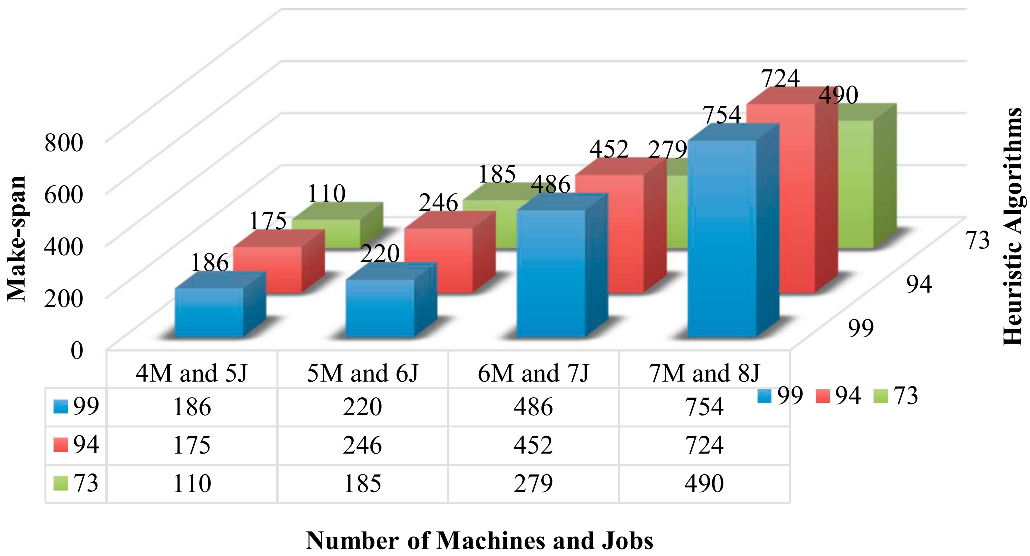

Fig.4 Comparison of Results in Bar Graph of Heuristics

Moreover, the proposed model is minimizing the Make-span in each and every case up to 50%, whereas the other considered algorithms are consuming more time i.e. difference of 754 and 490. This work clearly shows the importance of the proposed algorithm in scheduling purposes. Hence, the developed model is validated in the case of considered previous research work and other heuristic methods as well. In the next chapter, it is discussed about the implementation of the proposed algorithm and comparison of results with production industry.

5 Conclusion and Discussions

Heuristics are playing key role in the current production scheduling sector. In the present research, a new heuristic model is proposed depending on search algorithms for production scheduling to make the present scheduling process efficient. The proposed algorithm can be used for solving scheduling problems up to 10 machines. Otherwise, there are some cases to be considered as mentioned. Some of the major discovering conclusions from the confront work experienced. The make-span has been minimized using branch and bound algorithm and the optimized results show that the variation in the completion times. Based on the objective function, the unknown variables of each machine are evaluated. Multi-objective function is suggested for scheduling the tasks in sequence of scheduling as an efficient and including all measures, as multiple (many) decisions are often required in this dynamic and competitive surrounding. To carry on with conflicting condition, it is also compromising as optimum successiveness can be found by selecting the assignment depending on number of sources and customers. The actual data (processing time and make-span) that obtained from piston manufacturing industry is optimized using Heuristics. For machine 1, the processing times for two batches are 11 minutes after optimization and as a result, the time is reduced to extent by 36.8%. The population size and the length of the string are dynamic, each consists of number of machines and jobs. Performance of proposed algorithm shows most beneficial results as it changes with the increase in number of machines particularly for huge job size in scheduling problems. Proposed branch and bound algorithm prove to be an effective approach for optimizing multi-objective sequence and Scheduling problem for any job and machine size as it does not depend on random numbers as the other heuristics. By using heuristics, the processing time and the total make-span have been optimized. More products (jobs) and more machines processing time can be minimized i.e. up to 10 machines and 10 jobs, it can be easily optimized for all the considered data.

6 Future Scope

The future work may be minimization of total transportation time, delay time and also cycle time by using Hybrid branch and bound and the results can be compared with other heuristics algorithms. An application can also be developed for fast and accurate scheduling using the above modified method.

References

[1] J. H. Duan, M. Zhang, G. Y. Qiao and J. Q. Li. (2011) "A Genetic Algorithm for Permutation Flowshop Scheduling with Total Flowtime Criterion." *Chinese Control and Decision Conference (CCDC)*, Mianyang, China.

[2] M. Amiri, L. Olfat and M. Ghorabaee. (2014) "Simultaneous Minimization of Total Tardiness and Waiting Time Variance on a Single Machine by Genetic Algorithms." *International Journal of Advanced Manufacturing Technology,* **72(4)**: 439-446.

[3] N. Nouri and T. Ladhari. (2015) "An ABC Algorithm Minimizing Regular Objectives for Blocking Permutation Flow Shop Scheduling Problem," *2nd International Conference on Automation,Control, Engineering and Computer Science (ACECS-2015)*.

[4] S. H. Zegordi and M. Yavari. (2018) "A branch and bound algorithm for solving large-scale single-machine scheduling problems with non-identical release dates" *European Journal of Industrial Engineering,* **12(1)**: 24-42.

[5] R. Cordone, P. Hosteins and G. Righini. (2018) "A Branch-and-Bound Algorithm for the Prize-Collecting Single-Machine Scheduling Problem with Deadlines and Total Tardiness Minimization," *INFORMS Journal on Computing.* **30(1)**: 28-36.

[6] K. Abhary, Z. Kovacic, S. Lundberg, R. Narayanan and S. Spuzic. (2015) "The Application of a Hybrid Algorithm to Roll Pass Design," *International Journal of Advanced Manufacturing Technology.* **79(1)**: 1063-1070.

[7] A. Costa, F. Cappadonna and S. Fichera. (2014) "A Novel Genetic Algorithm for the Hybrid Flow Shop Scheduling with Parallel Batching and Eligibility Constraints," *International Journal of Advanced Manufacturing Technology.* **75(5)**: 833-847.