International Journal of Research In Science & Engineering
Volume: 2 Issue: 3
e-ISSN: 2394-8299
p-ISSN: 2394-8280

A simple advanced evolutionary algorithm for job shop scheduling

Kalyani D. Pawar[1], Kanchan A. Deshmane[2]

[1] *CSE, SVERI's COE Pandharpur,* ***kalyanipawar12@gmail.com***
[2]*Assi. Prof., CSE, SVERI's COE Pandharpur,* ***K_Deshmane@rediffmail.com***

ABSTRACT

Job shop scheduling is the one of the major conceptual tasks in today's fast processing demanding era. Many times this job scheduling is done based on some limited amount of resources therefore many assumptions need to make while achieving this. The major assumption for this is every job need to be run in one after another manner in the respective patterns. Only one job can be done in one machine at any given instances. These kind of scheduling techniques are been very helpful in managing the CPU performance. So it is really a challenging job to identify the pattern of the job that can be performing in the lowest minimum span. Many methods are been existing to solve this thing but all are struggling with the increasing complexity as the number of job grows.

Keywords*: Job Shop Scheduling , Evolutionary Algorithm, IGA etc.*

---

1. INTRODUCTION

Job Shop Scheduling Problem (JSSP) is one of the most difficult combinatorial optimization problems, which is used in complex equipment manufacturing system to validate the performance of heuristic algorithms. The research on JSSP not only promotes the development of relative algorithms in the field of artificial intelligence, but also provides means of solutions and applications for complex JSSP. JSSP can be thought of as the allocation of resources over a specified time to perform a predetermined collection of tasks. Job shop scheduling has received a large amount of attention, because it has the potential to dramatically decrease costs and increase throughput, thereby, increasing the profits in automation industries. Job-shop is a system that processes n number of tasks on m number of machines. The total ordering defines a set of precedence constraints, meaning that no activity can begin execution until the activity that immediately precedes it in the complete ordering has finished execution. Each of the m activities in a single job requires exclusive use of the resources defined in the problem. No activities that require the same resource can overlap in their execution and once an activity is started it must be executed for its entire duration. Usually, these orders differ in terms of processing requirements, materials needed, processing time, processing sequence and setup times. Job-shop problems are widely known as a NP-hard (NonPolynomial Deterministic) problem and commonly defined as a set of jobs whose operations are to be processed in an uninterrupted manner on a given machine for a specified length of time.

GA is a local search algorithm that follows the evolution paradigm. Starting from an initial population, the algorithm applies genetic operators in order to produce offsprings, which are presumably more fit than their ancestors. Every individual in the population represents a solution at the end of every generation. The major strength of GA when compared with other local search algorithms lies in the fact that in a GA framework more strategies can be adopted together to find individuals to add to the mating pool, both in the initial population phase and in the dynamic generation phase. Such an adaptation allows the search space to be explored at every algorithm step. The Ant Colony System (ACS) algorithm is a distributed algorithm which is extensively used to solve NP-hard combinatorial optimization problems. Its original model is based on the foraging behavior of real ants who find an approximately shortest way to the food by detecting the density of pheromone deposited on the route. In real ants, the term pheromone denotes the chemical substance deposited by ants as they move, but in artificial ants it acts like an attraction for the other ants to follow. In general, ants create pheromone paths from their nests to the available

food sources and the shortest path is the one with highest pheromone concentration between the source and the destination.

2. LITERATURE REVIEW

2.1 DYNAMIC SCHEDULING TECHNIQUES

Dynamic scheduling has been solved using the following techniques, heuristics, meta-heuristics, knowledge-based systems, fuzzy logic, neural networks, hybrid techniques, and multi-agent systems.

2.2 HEURISTICS

Heuristics in this context are problem specific schedule repair methods, which do not guarantee to find an optimal schedule, but have the ability to find reasonably good solutions in a short time.

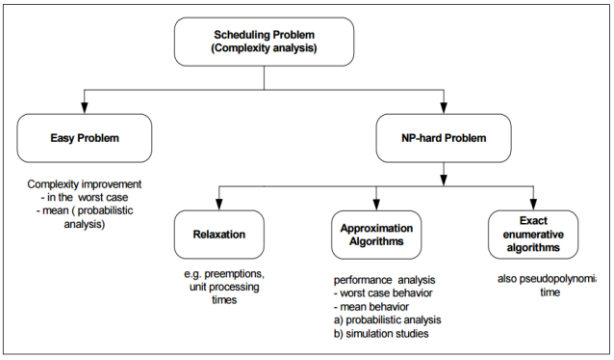

Fig 2.1 An Analysis of a scheduling problem – schematic view

The study could be concluded that the performance of dispatching rules is being influenced by routing of jobs and shop floor configurations. Hiroshi used shift bottleneck procedure to solve job shop scheduling problem. The objective of problem is to minimize total holding cost. The specific constraint is added to the problem. The added constraint is no tardy job constraint. The experiment show that shift bottleneck procedure can reduce computation time. Anthony presented Memmetic algorithm for job shop with time lag. The time lag means minimal and maximal between start times of operations.

3. PROPOSED SYSTEM

Here we describe our “AnEfficient Framework for multiple job shop scheduling using interactive genetic algorithm “ in the above figure with the following steps

e-ISSN: 2394-8299
p-ISSN: 2394-8280

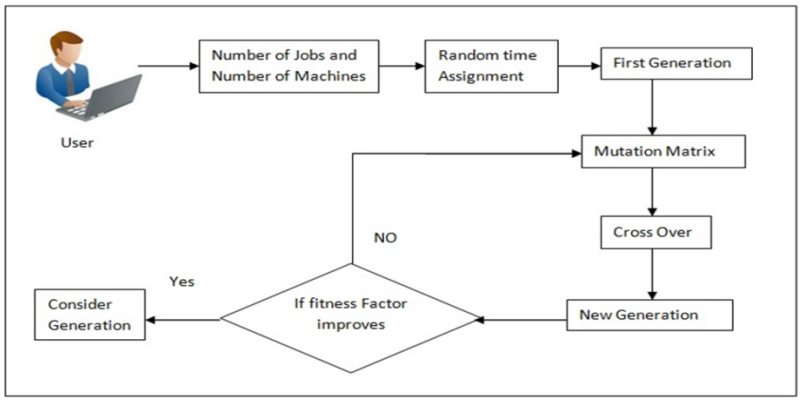

Fig 3.1 System Overview

3.1 Random Time Assignment

In this step our system takes input as number of jobs assigned to the number of machines to identify the optimized job combination which takes the least span of time by the last machine in the list.

3.2 Mutation matrix creation

In this step first mutation matrix is been created for the given input of number of jobs for the number of machines by assigning some random time in seconds as shown in the below example.

Number of Jobs : 4

Number of Machines : 3

Random Assigned JOB Schedule

	M1	M2	M3
J1	8	5	12
J2	4	14	8
J3	7	11	14
J4	10	2	10

3.3 Fitness Factor Identification

Here in this step total time that is been taking to complete this task for each machine is been calculating.

e-ISSN: 2394-8299
p-ISSN: 2394-8280

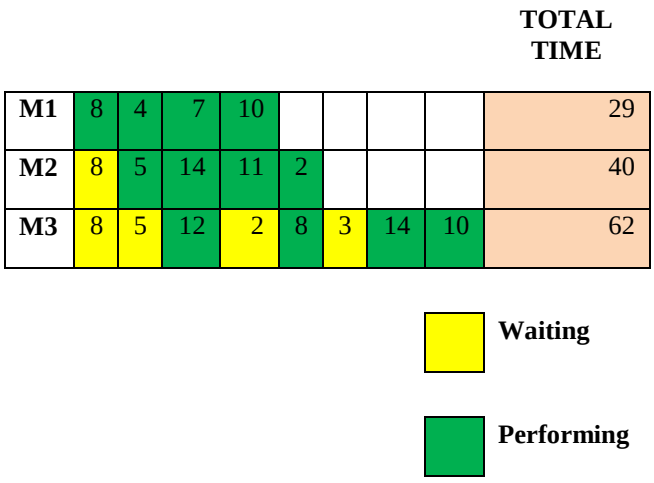

Fig 3.2 Fitness Factor Identification

3.4 Cross Over

Here in this step various other combinations created to calculate the fitness factor based on the time taken by each machines,combinations

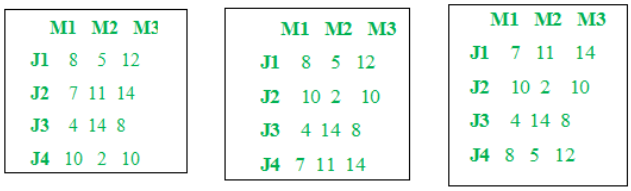

Fig 3.3 Example of Cross Over

3.5 Decision making

Here in this step for every cross over fitness function is calculated based on the probability of the last machines timing.

4. CONCLUSION

Proposed method is efficiently allocates the random time for multiple jobs assigned to the multiple machines to create job matrix. Proposed method's technique of identification of fitness factor before the last job reaching to the last machine eventually saves the time of the generation. Based on the last best time the generation crossover are allowed to carry on further to identify the best possible least performance time combination.

The major contradictory to the proposed system is that this system is developed in real time development technology like java and it adversely works efficiently for all the possible combination of jobs based on the hardware combination of the machine.

5. FUTURE SCOPE

The proposed system can be enhanced to identify the right combination of the jobs by the waiting time analysis by the different machines. This process can be carried out by in depth analysis of the job switchover method among different machines to enhance the technique of job shop scheduling.

REFERENCES

[1] Moscato, Pablo. "On genetic crossover operators for relative order preservation." *C3P Report* 778 (1989).

[2] Sousa, P., & Ramos, C. (1999). A distributed architecture and negotiation protocol for scheduling in manufacturing systems. Computers in Industry, 38(2), 103–113.

[3] Wu, S. D., Storer, R. H., & Chang, P. C. (1991). A rescheduling procedure for manufacturing systems under random disruptions. In Proceedings joint USA/German conference on new directions for operations research in manufacturing (pp. 292–306).

[4] Youssef, H., Sait, S. M., & Adiche, H. (2001). Evolutionary algorithms simulated annealing and tabu search: a comparative study. Engineering Applications of Artificial Intelligence, 14(2), 167–181.

[5] Sabuncuoglu, I. (1998). A study of scheduling rules of flexible manufacturing systems: a simulation approach. International Journal of Operational Research, 36(2), 527–546.

[6] Applegate D., Cook W., 1991, A Computational Study of the Job-Shop Scheduling Problem, ORSA Journal on Computing, Vol. 3, pp. 149–156.