A Survey on Different Techniques used for Solving Job Shop Scheduling Problem

Deepa Chaudhary, Deepali Singh, Nancy Rawat, Prerit Agarwal, Piyush Kumar

Abstract: This paper aims to have a comparison of a few of those techniques which are already suggested by different researchers in order to find optimal solution for the problem of job shop scheduling. Many approaches such as different crossover operators, variation in mutation operators and constrained problem statement etc. have been applied in order to achieve this aim. Different techniques discussed here are critical block (CB) neighbourhood and disjunctive graph (DG) distance in crossover, fusion of crossover and local search, penalty function, penalty function with delay constraint and random keys for sequencing.

Keywords: **comparison, genetic algorithm (GA), job shop scheduling problem (JSSP), optimization, survey.**

I. INTRODUCTION

Job shop scheduling problem (JSSP) is not a very new problem being faced by industries. Since 1960's researchers have been trying to unravel the mystery of JSSP and still are putting their efforts into it. The reason behind this problem being in research for so long is that one cannot be sure of finding schedule for *m* number of jobs occurring on *n* number of machines (with a processing time of their own) while making an optimal use of resources and time. That is why this has been considered as an N-P Complete problem where, we know that solution exists but one cannot assure the solution within polynomial time.

During the study, it has been found out that JSSP taken into consideration can be described as [1] [2] [3] [4] [5] [11] a problem existing on shop floor with *m* number of tasks on *n* number of machines.

- Here jobs are considered as activities that take place and machines are considered as resources that are to be used optimally.
- Technique constraint is applied on JSSP which says that an operation must be processed only after all its precedent operations are finished.
- Also, resource constraint is applied on JSSP which says that each job must be processed on each machine exactly once and one at a time.
- Time taken by first job processing on first machine till finishing last job processing on last machine is known as make-span of schedule. Researchers aim to minimise this make-span.
- Jobs must not be interrupted in between.

Deepa Chaudhary, is with Department of Computer Science and Engineering, Inderprastha Engineering College, Ghaziabad, Uttar Pradesh, noordeepa.chaudhary@gmail.com, Deepali Singh, c/o Mahendra Singh, H. No. 765, Sector- 15, Vasundhara, Ghaziabad, U.P. 201012, deepali0310@gmail.com and Nancy Rawat, c/o Prem Singh, 43/1, Uttranchal Enclave, Kamal Pur, Burari, Delhi 84, rawat.nancy09@gmail.com.

JSSP is such a complex combinatorial problem that instead of finding an exact solution, researchers look for an optimal solution in reasonable time by use of heuristics techniques. Different approaches have been proposed for such a scheduling problem like branch and bound [6], dynamic programming [7], simulated annealing [8], priority rules, Tabu search [9], genetic algorithm [10].

Among many suggested techniques, genetic algorithms (GA) have been taken frequently into consideration for solving this problem.

Organization of rest of the paper is done in the following manner : section 2 gives a very brief overview of Genetic Algorithm, in section 3 different techniques used for solving a JSSP namely, critical block (CB) neighbourhood and disjunctive graph (DG) distance in crossover of GA, the penalty method for constraints in JSSP, fusion of crossover and local search, penalty function with delay constraints, random keys for sequencing and optimization respectively, have been discussed, section 4 presents a tabular comparison of these algorithms with a conclusion in section 5.

.

II. GENETIC ALGORITHM

Genetic algorithm (GA) was proposed by John Holland [10]. It follows the basic principle of Darwin's theory of evolution. This principle states that only the strongest of all individuals survives over generations. GA repeatedly uses information present in solution population to generate new solutions with better performances.

GA involves few basic steps [11]:-

1. Population initialisation
2. Fitness evaluation
3. Selection
4. Crossovers
5. Mutation
6. Termination criterion

III. DIFFERENT JSSP SOLVING TECHNIQUES USED IN DIFFERENT RESEARCHES

A. CB Neighbourhood and DG Distance in Crossover of GA [1]

A concept of critical block (CB) neighbourhood and disjunctive graph (DG) distance is used during the process of crossover in genetic algorithm.

For crossover based on CB neighbourhood and DG distance [1] :-

- Considering two parents $m1$ and $m2$.
- Set $z = a$, which generates CB neighbourhood for $z, N(z)$.
- do

  for each y_i in $N(z)$, calculate distance with respect to $m2$ to produce $D(y_i, m2)$.

  sort $D(y_i, m2)$ in ascending order.

  starting from first in sorted $D(y_i, m2)$ where y_i with probability=1

  if fitness value is less than current fitness value i.e., $V(y_i) \leq V(z)$

  else accepted if probability=0.5.
- Starting from $m1$, modify z step by step towards $m2$.
- After sometime, z loses $m1$'s characteristic and inherit $m2$'s characteristic.
- Choose child depending on less DG distance between child and both its parents.

Here, in crossover and mutation, researchers used critical block neighborhood and the distance measured helped them to evaluate the schedules. Result has shown that the implementation of critical block neighbourhood and the distance measure can lead us to the same result obtained by other methods..

B. *The Penalty Method for Constraints in JSSP* [2]

Here, another new method of employing penalty on the solution, if the solution violates any constraint, was analyzed during the study. The penalty function can be explained through given equation :

$$P(\vec{x}) = v.k^{\alpha}.(\frac{n_{infeasible}}{n})^{\beta} \sum_{i=1}^{n_c} w_i < g_i(\vec{x}) >^2$$

where,

v is a constant that controls the proportion between $F(\vec{x})$ and $P(\vec{x})$,

k is the iteration number, $n_{infeasible}$ is the number of solutions,

n is the number of solutions,

$< g_i(\vec{x}) >$ function denotes degree of violating $i-$th constraint

w_i is the factor denoting the importance of constraint

and α and β are constants

The objective is to minimize the make-span, i.e., the final completion time of all the jobs. For a permutation, we can get a complete scheduling by decoding process.

C. *Fusion of Crossover and Local Search* [3]

Multistep Crossover Fusion (MSXF) is a new crossover operator in which local search functionality is built-in. A local neighbourhood search algorithm is used for base algorithm of MSXF.

Algorithm of MSXF [3]:

- We have two parents $n1$ and $n2$.
- Now we set $x = n1 = z$.

  do

  for each y_i that belongs to $P(x)$, we calculate $d(y_i, n2)$.

  - sort these y_i in ascending order of $d(y_i, n2)$.

  do

  - randomly choose y_i from $P(x)$ but giving more preference to the smaller value of *i*.
  - calculate $V(y_i)$ if y_i has not yet been visited.
  - if $V(y_i) \leq V(x)$ then accept y_i with probability=1, else accept y_i with probability=$P_c(y_i)$.
  - now, the index of y is changed to n from i and induces of $y_k^{(k} \in \{i+1, i+2 \ldots n\}$ from k to $k-1$

  until y_i is accepted.

  - set $x = y_i$.
  - set $z = x \; if \; V(x) < V(z)$

  until some termination criteria is satisfied.
- next generation uses z.

D. *Penalty Functions with Delay Constraints* [4]

This method makes use of a penalty function as described in [2] with an addition of delay constraint. It has been analyzed that the problem considered was for 4 machines and 6 jobs where each job has 2 numbers of identical pieces. The delays of any kind, during the manufacturing of jobs at manufacturing site, are also considered during the algorithm generation. Hence, the inputs are processing time and delay time. Researchers have taken the processing time from a manufacturing unit in Northern India.

Assumptions made were [4] :

- The load/unload station capacity is unlimited.
- Each machine completely manufactures the job assigned to it.
- The jobs are atomic.

The inputs once set cannot be changed during the generation of the particular schedule.

E. *Random Keys for Sequencing and Optimization* [5]

Random keys can be defined as a method of solution representation using which many feasible offspring can be produced for various problems of optimization and sequencing. These random keys are used to represent a solution with random numbers. For decoding the solution random values are used as sort keys. Offspring feasibility problem can be eliminated by using chromosomal encoding.

Structure of random keys can be defined as [5]:

- Chromosomes are formed by generating random number for every modeling issue.
- A derived solution can be reached from these sorted random keys and prioritizing the keys from the order deduced.

- Instead of derived solutions, crossover operator is applied on random keys.

Random keys are considered so important because as they form feasible offspring solution after crossover.

Considering, an n jobs and m machine problem, generate an integer (1…m) randomly for each job and a uniform deviate (0, 1) is added to it. Here, machines are assigned according to the integer part of the random key and sequence is sorted according to fractional part. Assuming that jobs are processed at their earliest possible time, a schedule can be constructed.

IV. A COMPARATIVE STUDY

A tabular comparison between all the algorithms stated in the previous section (DIFFERENT JSSP SOLVING TECHNIQUES USED IN DIFFERENT RESEARCHES) is depicted in the table (TABLE 1).

TABLE 1 COMPARISON OF VARIOUS JSSP SOLVING GENETIC ALGORITHMS

GA	**Technique Employed**	**Chromosome Representation**	**Initial Population**	**Crossover Operation**	**Mutation Operation**	**Termination Criteria**
CB Neighbourhood & DG distances in crossover of GA[1]	Critical block neighbourhood & disjunctive graph distances	Job sequence matrix	Randomly generated	CB neighbourhood & DG distances	If DG distance between parents is less than predefined value	Population size, number of iterations for crossover and mutation.
Penalty function for constraints in JSSP[2]	Penalty function	3-D vector <j,m,d>	Giffler & Thompson algorithm	Repeated appending to parent 2 after drawing from parent1	Inverse mutation, Interchange mutation & Insert mutation	Best solution obtained or total number of generations set.
Fusion of crossover and local search[3]	Multi step crossover fusion	A set of nodes with 0 as (start) & * as (end) nodes	Equal number of left and right active schedules	Local search	Multistep Mutation Fusion when distance between parents is too small	Number of iterations.
Delay constraint with penalty function[4]	Delay constraint		Randomly generated	One point crossover	Transposition with 0.5 probability	Number of iterations
Random keys for sequencing[5] and optimization	Random keys for solution representation	Mapping of chromosomes in literal space to random numbers.	Randomly generated	Parameterized crossover	Immigration	Number of iterations.

V. CONCLUSION

By the comparisons done so far, we have realized that there is no specific algorithm for the computation of an optimal solution of JSSP. It is deduced after the comparisons that if we try to solve the problem with some constraints then a near optimal solution can be achieved.

REFERENCES

[1] Mahanim Omar, Adam Baharum, Yahya Abu Hasan, “A job-shop scheduling problem (JSSP) using genetic algorithm (GA),” Proc. 2nd IMT-GT Regional Conference on Mathematics, Statistics and Applications, University Sains Malaysia, Penang, June 2006.

[2] Liang Sun, Xiaochun Cheng, Yanchun Liang, “Solving job shop scheduling problem using genetic algorithm with penalty function,” International Journal of Intelligent Information Processing, vol. 1, no. 2, December 2010.

[3] Takeshi Yamada and Ryohei Nakano, “A fusion of crossover and local search,” IEEE International Conference on Industrial Technology, (ICIT ‘96) Shanghai, China, pp. 426-430, December1996.

[4] Swati Singh, Shruti Kapoor, Shelly Chikara, Mr. Vijai Singh, “Improved solution to job shop scheduling problem with delay constraints using genetic algorithm,” International Journal of Computer Applications(0975-8887), vol. 45, no.-13, May 2012.

[5] James C. Bean, “Genetic algorithms and random keys for sequencing and optimization,” OSRA Journal on Computing, vol. 6, no. 2, Spring 1994.

[6] P. Brucker, B. Jurisch, B. Sievers, “A branch and bound algorithm for job shop scheduling problem,” Discrete Applied Math, vol. 49, pp. 105-127, 1994.

[7] Lorigeon, “A dynamic programming algorithm for scheduling jobs in a two-machine open shop with an availability constraint,” Journal of the Operational Research Society, vol. 53, no. 11, pp. 1239-1246, 2002.

[8] J. Gao, M. Gen, L.Y. Sun, “A hybrid of genetic algorithm and bottleneck shifting for multiobjective flexible job shop scheduling problems,” Computers and Industrial Engineering, vol. 53, no. 1, pp. 149-162, 2007.

[9] C.Y. Zhang, P.G. Li, Y.Q. Rao, “A very fast TS/SA algorithm for the job shop scheduling problem,” Computers & Operations Research, vol. 35, pp. 282-294, 2008.

[10] J.H. Holland, Adaptation In Natural And Artificial Systems, Ann Arbor, MI: University of Michigan Press, 1975.

[11] Ryohei Nakano and Takeshi Yamada, Conventional Genetic Algorithm for Job Shop Scheduling, University of California, San Deigo, Morgan Kaufmann Publishers.

Deepa Chaudhary is working as an Associate professor in Department of Computer Science and Engineering in Inderprastha Engineering College, Ghaziabad, Uttar Pradesh. She completed her M.Tech from Kurukshetra Univercity, Kurukshetra and now pursuing Ph.D. Her areas of interest are Artificial Intelligence, Data Mining, Web Mining and Machine Learning. She has a number of publications in International Journals and in International Conferences. She is an ACM member.

Deepali Singh is pursuing graduation in bachelor of technology in computer science stream (2009 - 2013) from Inderprastha Engineering College, Ghaziabad.

Nancy Rawat is pursuing graduation in bachelor of technology in computer science stream (2009 - 2013) from Inderprastha Engineering College, Ghaziabad.

Prerit Agarwal is pursuing graduation in bachelor of technology in computer science stream (2009 – 2013) from Inderprastha Engineering College, Ghaziabad.

Piyush Kumar is pursuing graduation in bachelor of technology in computer science stream (2009 – 2013) from Inderprastha Engineering College, Ghaziabad.