A SURVEY ON PARALLEL OPTIMIZATION ALGORITHM FOR JOB SHOP SCHEDULING PROBLEM

Mr.S.V.Kamble[1], Mr.V.C. Sakhare[2], Mrs.Jakiya Alase[3]
[1]M.Tech.(CSE), [2]M.E Electronics Engg., [3]M.E E&TC
Department of Information Technology,DKTE'S Textile & Engineering Institute, Ichalkaranji, MS, India

Abstract: In this paper we focus with parallel optimization algorithm that are mostly used for solving job shop scheduling problem (JSSP) using particle swarm optimization, Genetic algorithm and simulated annealing. Particle swarm optimization (PSO) algorithm is one of the most popular swarm intelligence-based algorithm, which is enriched with robustness, simplicity and global search, PSO inherits a natural parallelism, and receptiveness of fast processing capabilities. Genetic Algorithm (GA) is integrated with the parallel version of Simulated Annealing Algorithm (SA) is applied to the scheduling problem for better result. Parallelization in this algorithm is based on the application of multiple independent searches increasing the exploration in the search space. To enhance the performance for solving job shop scheduling problem parallel techniques are useful.
Keywords: particle swarm optimization (PSO), Genetic Algorithm (GA), Simulated Annealing (SA), job shop scheduling problem (JSSP)

I. INTRODUCTION

An optimization algorithm is a method which is executed iteratively by comparing various solutions till an optimum or a satisfactory solution is found. Scheduling is a techniques that allocates resources to perform a set of tasks over a period of time. Many real scheduling problems in the manufacturing industries are quite complex and very difficult to be solved by conventional optimization techniques [2]. The job shop scheduling problem (JSSP) is considered the most challenging one which has been the subject of many research studies for consumer demand and development in the production system during the recent decades [1]. The complexity of these problems has a direct dependence on constraints and shop environments upon which these problems are defined. The job-shop scheduling problem (JSP) is one of the hardest combinatorial optimization problems in the field of scheduling [2]. Job Shop Scheduling (JSSP) is one of the most complex scheduling problems related to manufacturing industries. However, FJSSP is also an NP-hard problem as the number of jobs increases; hence parallel techniques is mostly applicable for better result with good performance. The problem is described simply as follows: given n jobs to be processed on m machines. Each job consists of a predetermined sequence of task operations, each of which requires processing without interruption for a given period of time on a given machine. The rest of the paper is organized as follows: Literature review is presented in section II, Problem description and formulation is presented in section III, section IV describes the parallelization techniques of referred algorithm section V describes discussion and section VI concludes giving scope for future research.

II. LITERATURE REVIEW

Many researches have been done on Job shop scheduling and several methods so far used that include different approach, that have been developed to solve JSSP or Flexible job shop Scheduling (FJSSP). Hierarchical approaches consist of decomposing the original problem in order to reduce its complexity. This type of approach is natural for FJSSP since the routing and the scheduling sub-problem can be separated. S.V.Kamble, A.J.Umbarkar [1], developed hybrid multi-objective (PSO) for FJSSP with minimize five objective with Pareto archive

Ariyasingha and Fernando [4], developed ant colony algorithm .Using Ant Colony Optimization (ACO) algorithm, this research shows how well the algorithm solves the problem in multi-objectives. There are four objectives in this research; make span, mean flow time, mean tardiness and mean machine idle time.

Brandimarte [5], was the first to apply the decomposition approach into the FJSSP. He solved the routing sub-problem by using some existing dispatching rules and then focused on the scheduling sub-problem; it was solved by using a tabu search heuristic. Pauli [6] applied hierarchical approach.
Bozejko [7], have proposed the parallel simulated annealing for the job shop scheduling. But the same sequential algorithm is implemented more than one machine in a parallel order.
Thamilselvan and Balasubramanie [8,9], have used the various crossover strategies for genetic algorithm for JSSP and integration of Genetic algorithm with Tabu Search for the JSSP.

Xia and Wu [10], developed an easily implemented approach for the multi-objective flexible JSSP based on the combination of PSO and SA. A particle swarm optimization (PSO) is to use assign each operation to appropriate machine and simulated annealing (SA) to sequence of operation on the machine. PSO provides an initial solution for SA during the hybrid search process.

J. Gao [11],developed a hybrid genetic algorithm (GA) for the FJSSP. The GA uses two vectors to represent solutions. Advanced crossover and mutation operators are used to adapt to the special chromosome structure and the characteristics of the problem. Individuals of GA are first improved by a variable neighborhood descent (VND) which involves two local search procedures: local search of moving one operation and local search of moving two operation

Thamilselvan Rakkiannan [12], proposed parallel hybridization genetic algorithm with parallel simulated annealing for jssp, he analyze performance of individual algorithm to parallel and he found better result in parallel

Zhang, Shao[14], developed a particle swarm optimization (PSO) algorithm and a tabu search (TS) algorithm are combined to solve the multi-objective FJSSP with several conflicting and incommensurable objectives.

Saidi-Mehrabad, Fattahi [15], proposed TS algorithm for FJSSP, in this technique the initial feasible solution of jobs and operations sequences is first generated and then the algorithm searches for the best choice of the machine's alternative for this job and operation sequence.

III. PROBLEM FORMULATION

Problem Description:

In flexible job-shop problem, a set of n jobs is considered to process on a set of m machines. Each job, denoted by Ji ($1 \leq I \leq n$) has a predefine sequence nj of operations. Each operation Oij is processed by any one machine from a set *Mij*, which is the subset of machines that can perform *Oi*j. The processing time of *Oij* on machine k $(1 \leq k \leq m)$ is fixed and is denoted by *Pijk*. The FJSSP is designed to assign all the operations of the jobs to available machines to identify their starting and completion time and is aimed to obtain an optimal schedule with some objectives.

Assumptions:

- All job released at time 0.
- All machines available at time 0.
- Only one job can be carried on one machine at one time.
- Once an operation starts, it cannot be terminated before it finishes.
- Order constrains only exist in operation on the same jobs

Objective:

The objectives are to find an assignment and a schedule to minimize following objective simultaneously.

- Makespan, i.e., the maximal completion time of machines or jobs.
- Machine workload, i.e., the maximum processing time spent on any machine; this objective is used to keep the balance of workload among different machines and may help to avoid too much work scheduling on a certain machine.
- The total workload of machines, which is defined as total processing time over all machines.
- Total idle time, which is defined idle time of machines.
- Total Tardiness, which is defined as lateness of

TABLE I. A PROBLEM EXAMPLE

	O1	O2	O3	O4	Total Time	Due Time (D)
J1	4	5	2	3	14	15
J2	5	6	6	4	21	22
J3	3	4	5	6	18	22
J4	6	5	4	4	19	22

TABLE II. A SOLUTION EXAMPLE

	O1	O2	O3	O4		D
J1	M2	M3	M1	M4		15
J2	M1	M4	M3	M2		22
J3	M3	M1	M2	M4		22
J4	M4	M2	M1	M3		22

IV. PARALLELIZATION TECHNIQUES FOR OPTIMIZATION ALGORITHM

1. Parallel GA-PSO for JSSP

Authors has implemented CUDA programming, the computing works are divided into many smaller works that will be handled by the smallest processor unit, called thread. A thread manages its own memory and process concurrently with all other threads some threads make one group of threads called block. There could be many blocks with the same specific number of threads in a GPU. Each block shares a memory for communicating among the threads in that block. All blocks are eventually grouped to become a grid; which is used to perform a computation [3]

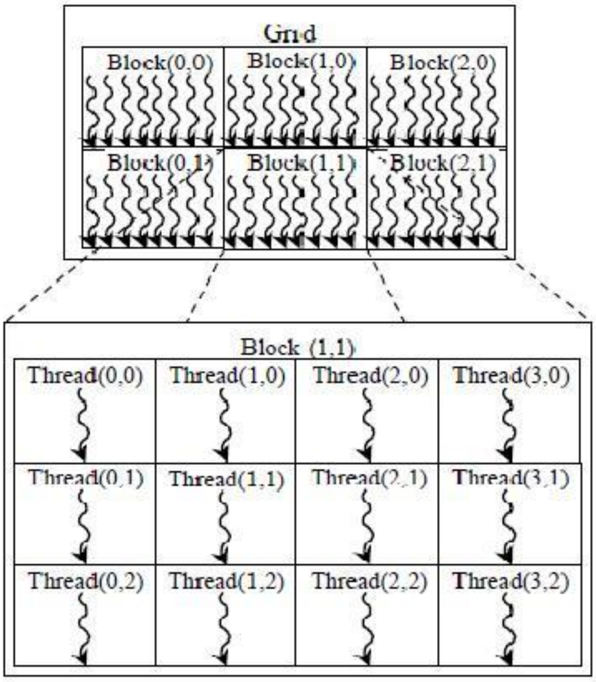

Fig.1 CUDA Processor Structure [13]

A particle or a solution in GA-PSO is actually independent to each other. Therefore it can be run independently. This gives us an opportunity to utilize the parallel programming by treating the particles as the independent agents that perform simultaneously. Instead of sequencing the execution of the particle of GA-PSO, parallelized GA-PSO executes them in parallel. The particle is treated in parallel programming. Firstly, the creation of every single particle takes place in the host. Secondly, they are copied into the device so that they can be handled in parallel manner. Thirdly, the parallel execution happens in the device. Finally, the computation result of every particle in the device is copied back to the host [3] .

1. *Set I, the maximum iteration.*
2. *Create N particles randomly. Goto parallel mode:*

Begin

– ***Label X**: Evaluate the fitness of each particle; which is the maximum of summation of delay and time taken by each operation of each job. The delay is there because of the constraint that prevents the machines to overlap each other among the jobs.*

– *Calculate the personal best of each particle.*

– *Mark the global best particle.* – *Calculate the distance of each particle to the best one.*

– *Calculate the Fitness difference of each particle to the best one.*

– *By using the possibility based on location distance and fitness difference, move the particle using mutation and crossover. Apply Crossover between selected particle's personal best and the global best one.*

– *If the iteration number ≤ I then goto **X** or else terminate computation.* – *Copy back the result to the host. End*

3. *Present the result to the user.*

Fig.2 GA-PSO Parallel algorithm [3]

2. Parallel PSO for JSSP
Authors has developed parallel PSO application, that they have followed the coarse-grained parallelization process; it is based on the master/slave paradigm. Each client evaluates in the same time its own population on different calculators, i.e. each client executes the sequential PSO algorithm. A master program (at the server level) is responsible for launching the parameters (number of particles and number of iterations) that are necessary for their functioning (slaves). At each iteration, each slave sends back its best particle and the corresponding Cmax to the server, which, in turn, pays attention to the results of each client. The server then chooses the best one from the received particles and sends it back to all the slaves to inform them of the best particle in the group (Gbest); each slave updates its Gbest if necessary [17]

General PSO Algorithm

Begin

Initialization of the parameters.

Initialization of the swarm (position and velocity).

Evaluation.

Repeat

Determine the best position P

Determine the best position of all or part of the swarm Update the velocities and the positions

Evaluation

Local search (optional)

Until (the stopping criterion is satisfied)

End

Parallel PSO for Master Computer

Begin

Given a network of K+1 computer.

Communicate to all the client computers the following:

1 Number of particles per population.

2 Global number of iterations.

3 Gbest=∅, Cmax*= ∅.*

4 Start the process of execution.

Repeat

1 At the reception of a new $Gbest_i$ value from client i. 2/ if (the received $Gbest_i$ is better than Gbest) then*

Gbest= $Gbest_i$*

Cmax=$Cmax_i$;*

Dispatch the new Gbest* to all the client machines; Until (the stopping condition is satisfied) Re

turn the best soluti on End.

PSO Algorithm for Slave Computer

Begin

When receiving

1 The number of particles per population

2 The global number of iteratio ns

3 Gbest.*

Do

1 Initialization of the swarm.

2 Initialization position and velocity.

3Evaluation.

4 Send Gbest to the server.

Repeat

Determine the best position Pbest. Determine the best position of the swarm Gbest Send Gbest to the server.

Update the velocities and the positions

Ealuation
Local
search
(optional)

Update Gbest from the server.

Until (the topping
criterion is satisfied)
End.

Fig.3 Parallel PSO algorithm [17]

3. Hybrid GA with Parallel SA for JSSP

Parallel optimization with SA generates a better solution with faster convergence. Initially, n number of client machines processes the SA algorithm with different initial Schedule. After the fixed number of iterations, the client machines are exchange the results with the server machine to get the best schedule [16]. Author has focus the main objective function of the proposed algorithm is to minimize the makespan. The procedure SA_Parallel () generate an Initial schedule S[i] and then the algorithm is parallelly running on different machines. Let N be the total

Number of iterations in each SA algorithm temperature, S[j] is the neighbourhood of S[i], Bc is best known solution cost, Bs is the best known schedule of JSSP. The generated S[j] is the input to the scheduling algorithm and then the algorithm compute the cost of S[j] as CS[j]. The cost of the new schedule is compared with the cost of the initial schedule to process the algorithm [16].

Procedure: SA_Parallel () [16]
Input:
temperature T, starting temperature Ts,
ending temperature Te, number of iteration
N. Begin
generate initial schedule S[i].
compute the cost CS[i] of initial schedule S[i].
n=1, T=Ts.
while T<Te
while n<N
select neighbourhood S[j] of S[i].
compute the cost CS[j] of the new schedule S[j].
compute = Cs[j]-Cs[i] .
if d£0 then
S[i]=S[j].
CS[i]=CS[j].
else
generate a random variable R~(0, 1).
if exp (-d/T)>R
S[i]=S[j].
CS[i]=CS[j].
end if
end if
n=n+1.
end while
*T= T*0.995.*
end while
if CS[i]<Bc
Bc=CS[i].
Bs=S[i].
end if
End

V. DISCUSSION

Reference paper	*Techniques*	*Scheduling Problem*	*Conclusion*
16	*GA –SA Parallelization*	*JSSP*	*Proposed genetic algorithm with simulated annealing is quite Successful to achieve better solution than the individual genetic or simulated annealing algorithm.*
3	*GA-PSO Parallelization*	*JSSP*	*GA-PSO while implementing parallel programming to gain significant increasingly speed, with increase size of problem*
17	*Parallel PSO*	*JSSP*	*performance of the parallel PSO algorithm is better than that obtained with the sequential PSO algorithm because it allows exploring more solutions than the former.*

VII. CONCLUSION

The empirical results show that the algorithm with parallel techniques obtain satisfactory result than individual algorithm.to increase the speed and performance is needed when the size of problem may be increase, for this parallelization is essential for better result. This problem needs to be addressed in the future. Also more than two meta-heuristic algorithms may be interpreted to improve the solution space.

ACKNOWLEDGMENT
The authors are thankful to the anonymous reviewers as well as the editors for their valuable comments and suggestions which have led to improve the quality of the paper. Our special thanks to Soniya Lalwani, Thamilselvan Rakkiannan ,Abdelhakim AitZai their work on Job shop scheduling using parallel optimization algorithm published in various journals.

REFERENCES

[1] S.V.Kamble , A.J.Umbarkar "Hybrid Multi-Objective Particle Swarm Optimization for Flexible Job Shop Scheduling Problem" I.J. Intelligent Systems and Applications, 2015, 04, 54-61 Published Online March 2015 in MECS,DOI: 10.5815/ijisa.2015.04.08
[2] Soniya Lalwani, Harish Sharma "A Survey on Parallel Particle Swarm Optimization Algorithms",Arabian Journal for Science and Engineering. https://doi.org/10.1007/s13369-018-03713-6
[3] Paulus Mudjihartono,Rachsuda Jiamthapthaksin "Parallelized GA-PSO Algorithm for Solving Job Shop Scheduling Problem ",2016 2nd International Conference on Science in Information Technology (ICSITech)
[4] I.D.I.D. Ariyasingha and T.G.I. Fernando, "A performance study for the multi-objective Ant Colony Optimization Algorithms on the Job Shop Scheduling Problem", International Journal of Computer Applications(0975 - 8887) vol 132 (14), December 2015.
[5] P. Brandimarte, "Routing and scheduling in a flexible job shop by tabu Search,", Annals of Oper. Res., 1993, Vol. 41, pp. 157-183.
[6] J Paulli, "A hierarchical approach for the FMS scheduling problem" , Eur J. Oper. Res., 1995, Vol. 86(1), pp. 32–42.
[7] Bozejko, W., J. Pempera and C. Smuntnicki, 2009. "Parallel simulated annealing for the job shop scheduling problem", Comput. Sci., 5544: 631-640. DOI: 10.1007/978-3-642-01970-8_62
[8] Thamilselvan, R. and P. Balasubramanie, 2011. "Analysis of various alternate crossover strategies for genetic algorithm to solve job shop scheduling problems", Eur. J. Sci. Res., 64: 538-554.
[9] Thamilselvan, R., P.Balasubramanie, 2012. "Integration of genetic algorithm with tabu search for job shop scheduling with unordered subsequence exchange crossover" ,J. Comput. Sci., 8: 681-693. DOI: 10.3844/jcssp.2012.681.693
[10] Xia, Z.Wu, "An effective hybrid optimization approach or multi-objective flexible job-shop scheduling problems", Computer Ind. Eng. 48(2), 2005, pp. 409–425.
[11] J.Gao, L.Sun, and M.Gen, "A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems", Comput. Oper. Res. 35(9), 2008, pp. 2892–2907.
[12] Thamilselvan Rakkiannan and Balasubramanie Palanisamy "Hybridization of Genetic Algorithm with Parallel Implementation of Simulated Annealing for Job Shop Scheduling" , American Journal of Applied Sciences 9 (10): 1694-1705, 2012 ISSN 1546-9239 © 2012 Science Publication
[13] Thread hirarchy", CUDA C Programming Guide, (Version 7.5) NVidia, 2015, [On-line] pp. 9. Available: http://docs.nvidia.com/ cuda/pdf/ CUDA_C_Programming_Guide.pdf [May 10, 2016].
[14] Zhang Guohui, Shao Xinyu, "An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem," Computers & Industrial Engineering 56, 2009, pp.1309–1318.
[15] P.Fattahi, M. Manesh, and A. Roshani, "A new solution seed for job shop scheduling problem," Appl. Mech. Mater 110–116, 2012, pp. 3899–3905
[16] Thamilselvan Rakkiannan and Balasubramanie Palanisamy "Hybridization of Genetic Algorithm with Parallel Implementation of Simulated Annealing for Job Shop Scheduling",American Journal of Applied Sciences 9 (10): 1694-1705, 2012 ISSN 1546-9239
[17] Abdelhakim AitZai,"Parallel branch-and-bound and parallel PSO algorithms for job shop scheduling problem with blocking",Int. J. Operational Research, Vol. 16, No. 1, 2013

