Hybrid Algorithm using Genetic Algorithm and Cuckoo Search Algorithm for Job Shop Scheduling Problem

Ala'a Abu-Srhahn[1] and Muhannad Al-Hasan[2]

[1,2]Computer Science Department, Zarqa University
Zarqa 13132, Jordan

Abstract

Job shop scheduling is an important and computationally difficult problem. The problem of job scheduling is known to be NP-complete. Genetic algorithm (GA) is one of the widely used techniques for constrained optimization. And its produce good results compared to other techniques. A disadvantage of GA, though, is that they easily become trapped in the local minima. In this paper, a Cuckoo Search Optimizer (CSO) is used along with a GA in order to avoid the local minima problem and to benefit from the advantages of both types of algorithms, 2-opt operation is adopted to improve the results. It minimizes the makespan and the scheduling can be used in scientific computing and high power computing. Our results have been compared with Ant Colony Optimization Algorithm (ACO) to show the importance of the proposed algorithm.

Keywords: *Job shop scheduling, cuckoo search optimizer, genetic algorithm, makespan, Ant Colony Optimization Algorithm.*

1. Introduction

One of the most well-known problems in both fields of production management and combinatorial optimization is the Job Shop Scheduling Problem (JSSP). In this paper n-by-m JSSP has been described as follows: our objective is to minimize the completion time of processing all jobs by scheduling n jobs on m machines. Each job comprises a set of operations which must each be done with predetermined processing sequence for different specified processing times, in a given job-dependent order. Operations of the same job must be on each machine exactly once, so different tasks of the same job are not processed concurrently [17]. In order to increasing production efficiency, reducing cost and improving product quality we need efficient methods for solving JSSP. Moreover, JSSP can be described as a problem of the class of NP-complete problems. In the general JSSP, there are n jobs which are to be processed on a set of m machines and there is no any exact algorithm can be applied to solve JSSP even when the problem scale is small, so researches have drawn the attention of JSSP because of its theoretical, computational, and empirical significance since it was introduced [9][14].

Since the JSSP is a complex topic, a dynamic programming is only applicable to modest scale problems by exact techniques, such as branch and bound [5]. Most of such techniques failed to obtain good solutions in case of trying to solve large scale problems because of the huge memory and lengthy computational time required. On the other hand, there are attractive alternatives to solve large scale problems such as, heuristic methods, include dispatching priority rules, shifting bottleneck approach and Lagrangian relaxation. With the development of new techniques from the field of artificial intelligence, much attention has been devoted to meta-heuristics. Meta-heuristics is one of the main classes of the construction and improvement heuristic, such as tabu search and simulated annealing. Another main class of meta-heuristic is the population based heuristic. Population based algorithms include Genetic Algorithm (GA) [11], Particle Swarm Optimization (PSO) [8], and et al., are such Successful examples.

In this paper, a new hybrid algorithm is proposed, combining the advantages of GAs and the CSOs to solve the JSSP.

The main contribution of this paper is,

- The proposal of a Hybrid algorithm which combines the advantage of GA and CSO.
- The performance comparison of the Hybrid algorithm with Ant colony optimization, and GA.

2. Problem Definition

An efficient algorithm is necessary for solving combinatorial optimization problem. In this Paper, a hybrid algorithm has been proposed for job scheduling which will combine the advantages of CSO and GA [1][15].

Definition: we have N jobs and M machines. Each and every job has its own order of execution that has to be executed on M machines. Each job has its own starting time[18]. Our algorithm objective is to minimize the makespan and it can also be used for job scheduling in scientific and high power computing [8].

Some of the assumptions for the job scheduling problem are:

- Jobs should be finite set.
- Each and every job contains a series of operations that needs to be performed.
- Machines should be finite set.
- All the machines are capable of handling only one operation at a time.

Some of the constraints are:

- No job should visit the same machines twice.
- No condition among operation of various jobs.
- Pre-emption type of operation is not allowed.
- A single machine is capable of handling individual job at a time.

No machine fails during its operation.

3. Metaheuristic Algorithms

A metaheuristic can be defined as an iterative generation process that guides a subordinate heuristic by combining intelligently different concepts to explore and exploit the search International Journal of Advanced Science and Technology space [11][13]. To find nearly optimal solutions, information should be structured using learning strategies [10]. This section describes the selected algorithms, CSO, GA, and ACO [19].

3.1 Cuckoo Search Optimizer Algorithm (CSO)

Yang and Deb (2009, 2010) developed the CSO algorithm based on the Lévy flight behaviour and brood parasitic behaviour [8]. The CSO algorithm has been proven to deliver excellent performance in function optimization, engineering design, neural network training, and other continuous target optimization problems and has solved the knapsack and nurse-scheduling problems.

Cuckoo birds have an aggressive reproduction in which females hijack and lay their fertilized eggs in other birds' nests. If the host bird discovers that the egg does not belong to it, it either throws away or abandons its nest and builds a new one elsewhere [11].

According to Yang and Deb (2010), the CSO algorithm is based on three assumptions:

- Each cuckoo lays one egg at a time and places it in a randomly chosen nest.
- The best nests with the highest quality of eggs (solutions) carry over to the next generations.
- The number of available host nests is fixed, and a host has a probability $p_a \in (0,1)$ of discovering an alien egg. In this case, the host bird either throws out the egg or abandons the nest to build a new one in a different location.

The third assumption can be approximated as a fraction: p_a of the n nests replaced with new nests (with new random solutions at different locations).

Lévy flight behaviour, rather than simple random walk behaviour, can be used to increase the performance of the CSO. The following formula can describe Lévy flight behaviour when generating new solutions $x_i(t+1)$ for the i^{th} cuckoo [10]:

$$x_i(t+1) = x_i(t) + \alpha \oplus le'vy(\lambda) \tag{1}$$

Where $\alpha > 0$ is the final size that has to be related to the problem of interest scale, and the product $\oplus$ refers to an entry-wise multiplication.

The formula that describes the Lévy flight behaviour in which the step lengths fit a probability distribution is:

$$le'vy \oplus u = t^{-\lambda} \tag{2}$$

According to this formula, cuckoo birds' consecutive jumps or steps mainly form a random walking process that corresponds to a power-law step-length distribution with a heavy tail.

3.2 Genetic Algorithm (GA)

Artificial intelligence research within the computer science field produced GA, a heuristic search tool designed to mimic the natural process of evolution. This heuristic, or so-called metaheuristic, is commonly used to generate useful solutions for optimization and search problems, often employing the natural techniques of evolution, such as inheritance, mutation, selection and crossover. [10][12] John Holland developed the formal theory of GA in the 1970s, and continued improvements to the price and performance value have made GA attractive for many problem-solving optimization methods [11]. GA have been shown to perform well in mixed (continuous and discrete) combinatorial problems. Although GA easily become trapped in local optima, they are computationally expensive and a probabilistic one. A GA begins with a set of solutions represented by a group of chromosomes called the population. A new population can be generated by International Journal of Advanced Science and Technology borrowing solutions from the current population or by applying genetic operators such as selection, crossover, and mutation to current population. The new population must be better than the old one [9].

The function of genetic operators warrants more detailed attention. The selection operator picks two parent chromosomes from the population based on their fitness to participate in the next operations, crossover and mutation[16]. These steps are considered the most important in a GA because they have a positive impact on the overall performance. First, parents form new offspring (children) through crossover probability. Shortly after, the mutation operator randomly exchanges alleles, as occurs in nature. To work well, GA require the definition of three important aspects [11]: the objective function, the genetic representation and its implementation, the genetic operators and their implementation.

Algorithm 1 Describes the genetic algorithm

```
Began:
  choose initial population.
  Initialize max_genration
  evaluate each individual's fitness.
  determine population's average fitness.
  While ( i< max_genration)
     select best-ranking individuals to reproduce.
     mate pairs at random.
     apply crossover operator.
     apply mutation operator.
     evaluate each individual's fitness.
    determine population's average fitness.
    i=i+1.
  End
End
```

3.3 Ant Colony optimization Algorithm (ACO)

The concept of ACO first emerged in the early 1990s [4] with the goal to simulate the behaviour of ants in nature: Ants wander randomly until they find food and then return to their colony, all the while laying down pheromone trails, or chemical substances that attract other ants searching for food. Once ants identify trails leading to food, they stop wandering randomly and follow the trail with the most pheromones. The ants continue to lay down pheromones [6], reinforcing this path. A path's attractiveness determines the quantity of pheromones. The more attractive a trail, the more ants travel it while laying down more pheromones, thus attracting even more other ants [7]. Since pheromones operate through evaporation, this process depends on the time. Whenever a path ceases to lead to food and is no longer used, the pheromones evaporate, and ants move onto other trails.

Algorithm 2 Introduces the ant Colony Optimization algorithm

```
Began:
  Initialize the base attractiveness, $\tau$, and visibility, $\eta$, for each edge;
     for (i < IterationMax)
       for each ant do
         choose probabilistically (based on previous equation) the next state to move into.
          add that move to the tabu list for each ant.
       repeat until each ant completed a solution.
     End
     for each ant that completed a solution do
       update attractiveness $\tau$ for each edge that the ant traversed.
     End
        if (local best solution better than global solution)
         save local best solution as global solution;
       End
     End
 End
```

4. A hybrid algorithm for jssp

The suggested algorithm combines the advantages of GA and CSO and overcomes the main disadvantage of GA easily becoming trapped in the local minima through the CSO, which performs the local search faster than the GA. Additionally; the CSO has only a single parameter, along with population size [3]. A 2-opt operation is adopted to improve and promote the results. The main steps are introduced in algorithm 3.

Algorithm 3: Solving JSSP using the suggested hybrid algorithm.

```
Begin
  Initialization - Job creation time, starting time
  Find out the number of task T that need to be scheduled.
  Initialization of nests and random initial solution
  Optimize initial solutions and saved in the bulletin board.
  Evaluate the makespan (fitness) of solutions Fi;
     While (t <MaxGeneration)
       Get a cuckoo randomly by Levy flights;
            Evaluate its quality/fitness Fi;
       Choose a nest among n (say, j) randomly;
     If Fi<Fj
        Replace j by the new solution;
     End
       GA operations {
         Selection: create matting pool
         Production: Mutation (flip, swap, slide)
         Evaluate population };
        Host birds abandon pa in(0,1)nests, and search pa new nests;
        Refresh the bulletin board and keeping the best solutions (and nests).
        Rank the solutions, and find the best (solution).
        t =t +1;
    End While
End
```

5. Experimental Results

In order to show the importance of the proposed algorithm, it has applied to different dataset then used CSO and GA for comparison in terms of makespan and time needed to run the algorithms, the table 1 show the data set that contain 2 job and 3 machines.

Table 1: Data set of 2 jobs and 3 machines

Job		**1**			**2**		
Operation		**1**	**2**	**3**	**1**	**2**	**3**
Machine	**1**	**5**			**6**		
	2		**7**				**8**
	3			**10**		**12**	

Figures 1, 2, and 3 show the result of apply the algorithms into the dataset shown in the table 1.

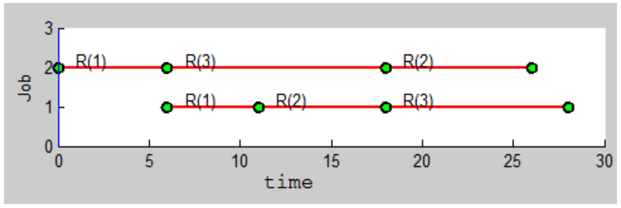

Fig. 1 Final results of hybrid algorithm.

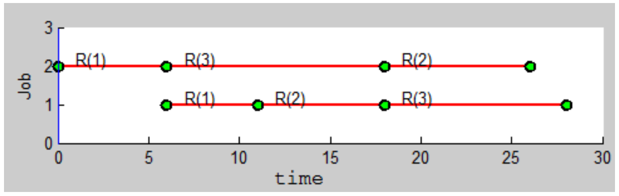

Fig. 2 Final results of genetic algorithm (GA).

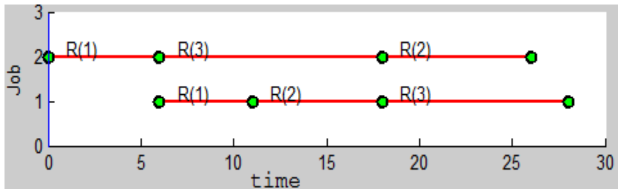

Fig. 3 Final results of Ant Colony Optimization Algorithm (ACO).

Table 2 shows the time, and makespan of GA, ACO, and hybrid algorithm.

Table 2: Time and makespan of the algorithms

	Hybrid Algorithm	***GA***	***ACO***
Time	**0.056522**	**0.133070**	**0.138325**
Makespan	**26**	**28**	**28**

The algorithms are applied on some well-studied benchmarks. In this paper, some problems that were contributed to the OR-Library are selected. The instances FT06, FT10, and FT20 are designed by Fisher and Thompson (1963), and instances LA01 to LA16 are designed by Lawrence (1984). The designers used them to compare performances of some heuristics and found these problems to be particularly difficult [1][2]. So these problems have been used as benchmarks for study with different methods by many researches[9].the table below show the comparisons of makespan between the proposed algorithm and others algorithm.

Table 3: Comparisons of makespan between the proposed algorithm and other algorithms

Ins	***Size***	***Hybrid Algorithm***	***GA***	***ACO***
Mt10	10×10	930	944	952
Mt20	20×5	1165	1169	1196
FT06	6×6	55	55	60
FT10	10×10	938	938	941
FT20	20×5	1165	1169	1175
La01	10×5	666	674	680
LA03	10×5	597	604	604
La04	10×5	590	597	601
La05	10×5	593	597	597
La06	15×5	926	926	931
La07	15×5	890	890	890
La08	15×5	863	874	880
La09	15×5	951	960	962
La10	15×5	958	958	958
La13	20×5	1150	1154	1159
La14	20×5	1292	1300	1307
La15	20×5	1207	1209	1212
La16	10×10	945	950	950
LA35	30×10	1888	1903	1958
LA36	15×15	1268	1279	1291

4. Conclusions

For the JSSP, makespan is one of the most important factors that the algorithm try to minimize it. A hybrid algorithm to minimize the makespan for JSSP has been presented, GA, and ACO utilized for the same purpose. The algorithms were tested using well known datasets in order to verify the validity of the proposed algorithm. The results show that the hybrid algorithm yields the best solutions as measured by makespan. The experimental results show that the proposed algorithm is effective and performs better than the compared algorithms.

References

[1] A. Abraham, R. Buyya, and B. ,Nath, B., “Nature’s heuristics for scheduling jobs in computational Grids”, Proceedings of the 8th IEEE International Conference on Advanced Computing and Communication, 2000, pp. 45–52.

[2] A. Abu-Srhan, and Al Daoud E., “A Hybrid Algorithm Using a Genetic Algorithm and Cukoo Search Algorithm to Solve the Traveling Salesman Problem and its Application to Multiple Sequence Aligment,” International Journal of Advanced Science and Technology, vol. 61, no.4, 2013, pp. 29-38.

[3] B. Al-Dulaimi, and H. Ali, “Enhanced Traveling Sale sman Problem Solving by Genetic Algorithm Technique (TSPGA),” World Academy of Science, Engineering and Technology, vol. 14, 2008, pp. 296-302.

[4] R. Babukartikl, and P. Dhavachelvan, “Hybrid Algorithm using the advantage of ACO and Cuckoo Search for Job Scheduling,” International Journal of Information Technology Convergence and Services (IJITCS), 2012, Vol.2, no.4.

[5] I. Brezina, and ková Z., “Solving the Travelling Salesman Problem Using the Ant Colony,” Management Information Systems, 2011, vol. 6, no. 4, pp. 10-14.

[6] J. Chen , et.al., “Flexible job shop scheduling with parallel machines using Genetic Algorithm and Grouping Genetic Algorithm,”, Expert Systems with applications, 2012, Vol. 39.

[7] S. Balin, “Non-identical parallel machine scheduling using genetic algorithm”, Expert Systems with applications, 2011, Vol 38, no.6, pp. 6814-6821.

[8] J. Magalhães-Mendes, ”A Comparative Study of Crossover Operators for Genetic Algorithms to Solve the Job Shop scheduling Problem”, WSEAS transactions on computers, Vol. 12, No. 4, 2013, pp. 164-173.

[9] H. Manar, and F. Shameem, “A Survey of Genetic Algorithms for the University Timetabling Problem”, International Conference on Future Information Technology IPCSIT, 2011, Vol.13,.

[10] M. Melanie, “An Introduction to Genetic Algorithms”, MIT Press, 1996.

[11] M. Omar, A. Baharum, and Y. Abu Hasan, “A job-shop scheduling problem (jssp) using genetic algorithm (ga),” The 2nd IMT-GT Regional Conference on Mathematics, Statistics and Applications, 2006, pp 13-15.

[12] I. Osman, and G. Laporte, “Metaheuristics:A bibliography,” Annals Operations Research, vol. 63, 1996, pp. 513-623.

[13] F. Pezzella, G. Morganti, and G. Ciaschetti, “A genetic algorithm for the flexible Job-shop Schduling”, Computer & Operations Research, Vol 35, no. 10, 2008, pp. 3202-3212.

[14] Z. Pooranian, M. Shojafar, J. Abawajy, and M. Singhal, "GLOA: A New Job Scheduling Algorithm for Grid Computing," International Jorunal of Interactive Multimedia and Artificial Intelligence, vol. 2, no. 1, 2013, pp. 59-64.

[15] R. Qing-dao-er-ji, and Y. Wang, “A new hybrid genetic algorithm for job shop scheduling problem”, Computer & Operations Research, Vol 39, no. 10, 2012, pp. 2291-2299.

[16] S. Sumathi, and P. Surekha,“PSO and ACO based approach for solving combinatorial Fuzzy Job Shop Scheduling Problem,” International Journal of Computer Technology and Applications, Vol 2 ,no. 1, 2011, pp.112-120.

[17] L. Sun, X. Cheng, and Y. Liang, “ Solving Job Shop Scheduling Problem Using Genetic Algorithm with Penalty Function,” International Journal of Intelligent Information Processing, vol. 1, no. 2, 2010, pp. 65-77.

[18] E. Valian, S. Mohanna, and S. Tavakoli, “Improved Cuckoo Search Algorithm for Global Optimization”, International Journal of Communications and Information Technology, IJCIT, vol. 1, no. 1, 2011, pp. 1-62,.

[19] X. Yang, and S. Deb, “Cuckoo Search via Levy Flights,” Proceedings of World Congress on Nature & Biologically Inspired Computing, 2009, pp. 210-225,.

Alaa Abu Srhan Received her BSc From Al Balqa University, Faculty of Engineering, MSc from Zarqa University, Faculty of Science and Information Technology, Jordan, 2014. Her research interest includes optimization, machine learning and image processing.

Muhannad Al-Hasan received his BSc from King Saud University, 1990, MSc in Medical Physics from Surrey University, UK, 1991, and his PhD in Computer Science from University of East Anglia, UK, 2006. He is assistant professor. His research interests include image processing and Bioinformatics.