MULTI PARENTS EXTENDED PRECEDENCE PRESERVATIVE CROSSOVER FOR JOB SHOP SCHEDULING PROBLEMS

Ong Chung Sin[1], Noor Hasnah Moin[2] and Mohd Omar[3]
[1, 2, 3]Institute of Mathematical Sciences, Faculty of Science
University of Malaya, 50603 Kuala Lumpur
Email: *[1]csin14@siswa.um.edu.my, [2]noor_hasnah@um.edu.my, [3]mohd@um.edu.my*

ABSTRACT

Job Shop Scheduling Problem (JSSP) is one of the hard combinatorial scheduling problems. This paper proposes a genetic algorithm with multi parents crossover called Extended Precedence Preservative Crossover (EPPX) that can be suitably modified and implemented with, in principal, unlimited number of parents which differ from conventional two parents crossover. JSSP representation encoded by using permutation with repetition guarantees the feasibility of chromosomes thus eliminates the legalization on children (offspring).The simulations are performed on a set of benchmark problems from the literatures and they indicate that the best solutions have the tendencies to be appeared by using 3-6 numbers of parents in the recombination. The comparison between the results of EPPX and other methodologies show the sustainability of multi parents recombination in producing competitive results to solve the JSSP.

Keywords: Job Shop Scheduling Problems, Genetic Algorithms, Multi Parents Crossover, Local Search

1.0 INTRODUCTION

Job Shop Scheduling Problem (JSSP) is one of the well-known hard combinatorial scheduling problems.It is not only a NP-hard problem, but also one of the most computationally difficult combinatorial optimization problems considered to date. This intractability is one of the reasons why the problem has been widely studied. Metaheuristic is one of the "approximation methods" that are designed to tackle complex optimization problems where other optimization methods have failed to be either effective or efficient [1]. Genetic Algorithms (GA) which is based on biological evolution is one of the metaheuristics that has been successfully applied to JSSP [2].

Since the first used of GA based algorithm to solve the JSSP proposed by Davis [3], GA has attracted many researchers to improve the efficiency of the scheduling method and frequently being used to solve scheduling problem. Various GA strategies have been introduced to increase the efficiency of finding the optimal or near optimal solutions for JSSP [4]. In the GA strategies, hybridization of GA with other methods or local search methods has provided good results in solving the problems. GA capitalizes on the strength of the local search in locating the optimal or near optimal solutions. Specifically, the local search procedure of Nowicki and Smutnicki [5] is embedded into GA by Gonçalves et al. [6] due to the effectiveness of the local search that increases the performance of GA.

In addition to hybridization, the structure of the GA can be modified and enhanced to reduce the problems often encountered such as premature convergence. Park et al. [7] retarded the premature convergence by using parallelization of GA (PGA) to find the near optimal solutions and Ripon [8] proposed hybrid approach to search the nearly-optimal and non-dominated solutions for JSSP with better convergence. Watanabe [9] proposed the use of crossover search phase into the GA with search area adaption. This modified GA has capacity for adapting to the structure of the solutions space. Tail redundancy of chromosome is a part of the problems in GA. Thus, heuristic method is embedded into crossover by Ripon et al. [10] to reduce the tail redundancy of chromosome when implementing crossover operation.

All these researches show that GA is not restricted to a single procedure and perform well by modifying the GA structure or hybridization with other methods to increase the accuracy of searching solutions. Particularly, recombination operators, especially crossover operators play important roles in the structure of GA. Crossover

between two parents are traditionally adopted in GA [11] but the GA can be modified accordingly to suit the problem at hand including selecting several numbers of parents for the crossover operation which we called it as multi parents crossover.

The application of multi parents recombination can be found in different research areas. Mühlenbein and Voigt [12] proposed Gene Pool Recombination (GPR) in solving discrete domain problems. Eiben and Kemenade [13] introduced the diagonal crossover as the generalization of uniform crossover and one-point crossover in GA for numerical optimization problems. Wu et al. [14] proposed multi parents orthogonal recombination to identity of an unknown image contour. Tsutsui and Jain [15] proposed multi-cut and seed crossover for binary coded representation and Tsutsui et al. [16] proposed simplex crossover for real coded GA. The crossover operators that were used in those areas show the good search ability of the operator but are very problem dependent.

The above literatures indicated the ascendency of multi parents crossover over two parents crossover. Although multi parents crossover has been used in different fields, to the best of our knowledge, only limited numbers are applied to combinatorial scheduling problems especially JSSP. In particular, Eiben et al. [17] proposed multi parents for the adjacency based crossover and Ting et al. [18] developed Multi Parents extension of Partially Mapped Crossover (MPPMX) for the Travelling Salesman Problems (TSP). Although the experimental results point out that adjacency based crossover of multi parents has no tangible benefit, but MPPMX show significant improvement in the use of multi parents crossover. In other words, one would expect that by biasing the recombination operator the performance of the GA would improve.

In this paper, we proposed Extended Precedence Preservative Crossover (EPPX) as a multi parents crossover. EPPX is a variation of the Precedence Preservative Crossover (PPX) proposed by Bierwirth et al. [19] which is one of the crossovers that perform well to find the solutions for the JSSP due to its capability to preserve the phenotypical properties of the schedules. EPPX as crossover operator will retain this advantage in the GA. The aim of this research is to ensure the sustainability of multi parents recombination using EPPX to solve JSSP in conjunction with simple local search.

The rest of the paper is organized as follows. In the next section, the JSSP is described in details and this is followed by presentation of the integration of JSSP with GA which also describes the proposed multi parents crossover. Section 4 describes the hybridization of GA with local search techniques and section 5 provides experimental results and analysis and finally, conclusions are drawn in section 6.

2.0 JOB SHOP SCHEDULING PROBLEM (JSSP)

The JSSP can be defined as a set of n jobs needs to be processed on a set of m machines. A job consists of a set of operations J, where O_{ij}, represents the $j^{th}(1 \leq j \leq J)$ operation of the $i^{th}(1 \leq i \leq n)$ job. The technological requirement for each operation processing time is denoted as p_{ij} and a set of machines is denoted by $M_k(1 \leq k \leq m)$.

Precedence constraint of the JSSP is defined as:

- Operation j^{th} must finish before operation $j^{th}+1$ in the job.
- A job can visit a machine once and only once.
- Only one operation can be processed in the machine at a time for one time.
- The delay time for the job transfer machine will be neglected and operation allocation for machine will be predefined.
- Preemption of operations is not allowed.
- There are no precedence constraints among the operations of different jobs.
- Neither release times nor due dates are specified.

The main objective of JSSP is to find the minimum makespan for the scheduling. The finish time of job i is represented by F_{iJ}. The completion of the whole schedule or the makespan is also the maximum finish time in the set of the jobs i. Therefore, the makespan denoted as C_{max} is expressed as follow:

$$C_{\max} = \max\left(F_{iJ}\right) \tag{1}$$

Let *G(k)* be the set of operations being processed in machine *k*, and let

$$X_{O_{ij},k} = \begin{cases} 1 & \text{if } O_{ij} \text{ has been assigned to machine } k \\ 0 & \text{otherwise} \end{cases}$$

The conceptual model of the JSSP can be expressed as follows.

$$\text{Minimize} \quad \left\{\max\left(F_{iJ}\right)\right\} \tag{2}$$

$$F_{ij} \le F_{ij+1} - p_{ij+1} \quad , j = 1,2,\ldots J, \text{ for all } i \tag{3}$$

$$\sum_{O_{ij} \in G(k)} X_{O_{ij},k} \le 1 , \text{ for all } k \tag{4}$$

The objective function represented by Eq. (2) minimizes the maximum finish time in the set of the jobs, therefore minimizes the makespan. Eq. (3) satisfies precedence relationships between operations and Eq. (4) imposes that an operation can only be assigned to a machine at a time. The problem is to determine a schedule that minimizes the makespan, that is, to minimize the time required to complete all jobs.

An example of 3 jobs and 3 machines, and their sequences are given in Table 1.

Table 1. Example for 3 jobs 3 machines problem

	Job	**Operation routing**		
		1	**2**	**3**
Processing time	1	3	3	2
	2	1	5	3
	3	3	2	3
Machine sequence	1	M1	M2	M3
	2	M1	M3	M2
	3	M2	M1	M3

3.0 GENETIC ALGORITHMS

GA is search heuristic that mimics the evolutionary processes in biological systems. Evolutionary processes such as reproduction, selection, crossover and mutation which inspired by natural evolution are used to generate solutions for optimization problems. Those techniques are translated into the form of computer simulations. GA begins with a population, which represents a set of potential solutions in the search space. It attempts to combine the good features in each individual in the population using random search information exchange in order to construct individuals which are better suited than previous individuals. Through the evolution, individuals that are poor tend to be

replaced by fitter individuals to generate a new population which represents a better estimate of the desired optimal solutions.

3.1 Representation

Representation of JSSP in chromosome can be classified in two approaches: direct and indirect [2]. Direct approach is represented as binary string and it evolves to find the better schedule. Indirect approach encodes an integer representation of the jobs in the chromosome. In this study, the chromosome is encoded using indirect approach where it is represented as permutation integer with repetition [20]. In this representation, number $i = 1,2,3$ ... which represent the number of jobs and i is repeated according to the number of operations required. Fig. 1 illustrates the representation of 3 jobs and 3 operations/machines. The chromosome is represented as [1 2 3 3 2 2 3 1 1] , where numbers 1, 2, and 3 in the chromosome represents job 1, 2, and 3 respectively. Each job consists of three operations so it is repeated three times in the chromosome. The chromosome is scanned from left to right, and at the j-th occurrence of a job number refers to the j-th operation in the technological sequence of this job. The chromosome created is always feasible and legal.

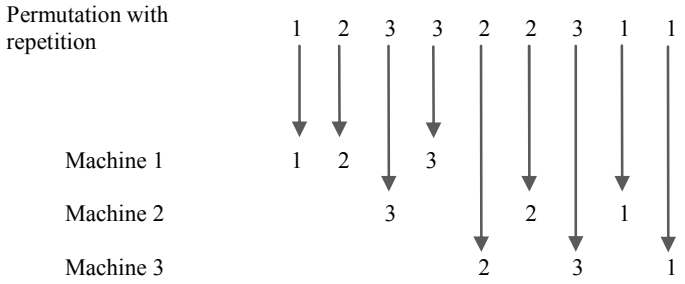


Fig. 1. Permutation with repetition representation for 3 jobs 3 machines

3.2 Objective Function

In the JSSP, each job has different finish time due to the different schedule operation time. C_{max} represents the maximum completion in the scheduling (please refer to Eq. (1)). The objective is evaluated for each chromosome and it also determines the ranking of the chromosome which is used in the process of selection. We use non-linear ranking to rank the chromosomes and each chromosome competes with each other and the selected chromosome will survive to the next generation based on the objective function (fitness value). Chromosome with greater fitness indicates the greater probability to survive. The highest ranking chromosome in a population considers as the best solution. It is noted that the lowest makespan is given the highest ranking.

3.3Selection

In this method, we use the Stochastic Universal Sampling (SUS). This fitness based proportionate selection will choose the chromosomes into recombination process with minimum spread and zero bias. After the recombination process, elitism strategy is applied and 10% of the best fitness new offspring replace the 10% of the worst individuals in the previous population to generate a new population.

3.4 Proposed Extended Precedence Preservative Crossover (EPPX)

In GA, there are no limitations that the recombination process needs only to be between two parents but multi parents consisting more than two parents is also acceptable. Therefore, some of the multi parents' crossover operators are extended from the two parents' crossover operators [14, 15, 18] for recombination process. As mentioned before, EPPX is an extension of PPX. A crossover mask in the form of a vector is generated randomly to determine the genes in which parent, specified in the mask, to be selected for recombination. The multi parents will then recombine into a single offspring (Fig. 2 (a)). Starting from the first element on the mask vector, the number 1 in the first element of the mask vector indicates that the first gene in parent 1 is selected. In general, the mask vector

indicates from which parent the element is to be selected. In this example, the selected job (job 3) is selected and eliminated in the other parents (Fig. 2 (b, c)). The second element in the mask indicates that the first element (after deletion) is to be selected also from parent 1 (Fig. 2 (c)). The third element in the mask shows that the first element in parent 3 is selected next (Fig. 2(d)). The process continues until all the elements in the mask have been examined. Fig. 3 shows the flow chart for multi parents crossover for JSSP.

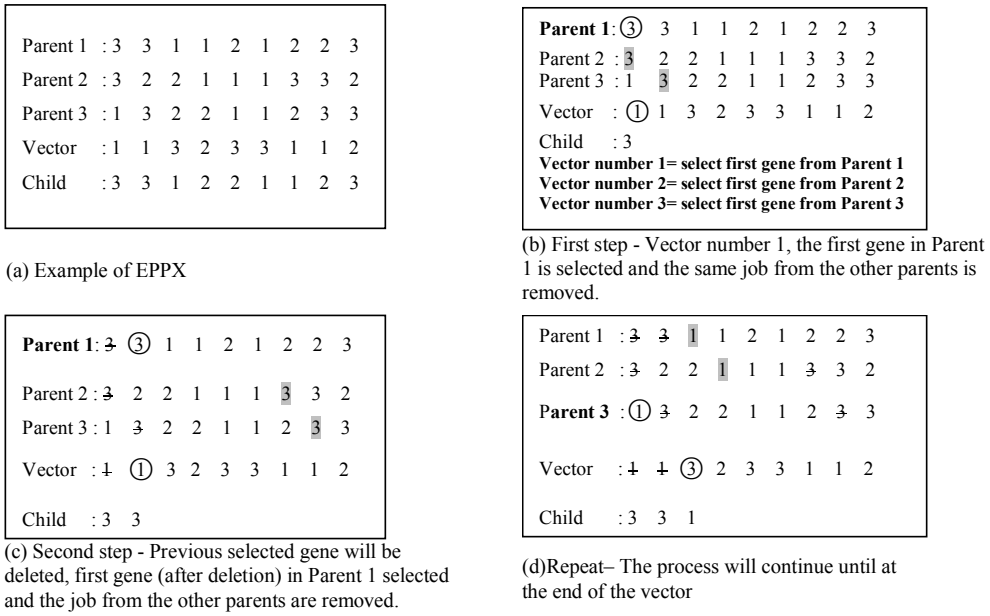


Fig. 2. EPPX crossover

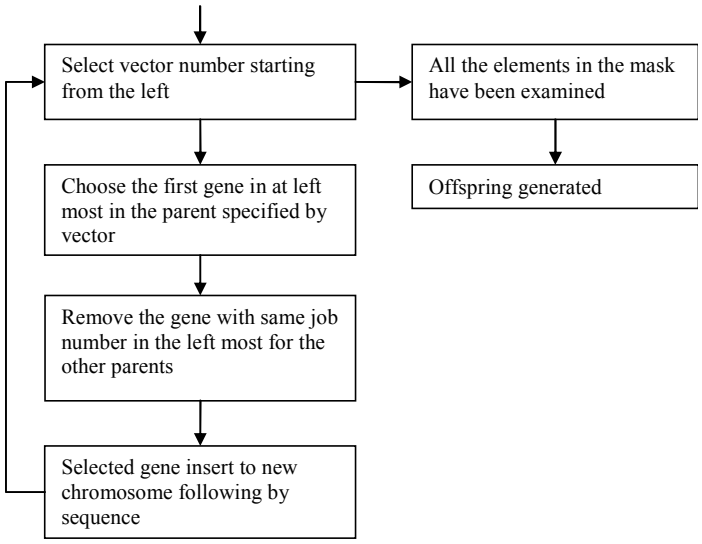


Fig. 3. Flow chart for EPPX

3.5Mutation

Mutation is a genetic operator, analogous to the biological mutation, which is used to maintain genetic diversity from one generation of a population of chromosomes to the next. The purpose of mutation in GA is to diversify, thus allowing the algorithm to avoid local minima by preventing the population of chromosomes from becoming too similar to each other, thus slowing or even stopping the evolution. This reasoning also explains the fact that most GA systems tend to avoid taking the fittest of the population only when generating the next population but using rather a random selection (or pseudo-random with a weighting toward those that are fitter). In this study, the mutation is applied by selecting two genes in different positions and different job (because of the precedence constraint) from the same chromosome to be swapped. The process will be repeated if two genes selected are at the same position or the same job. Fig. 4 illustrates the swapping of the two genes in the chromosome.

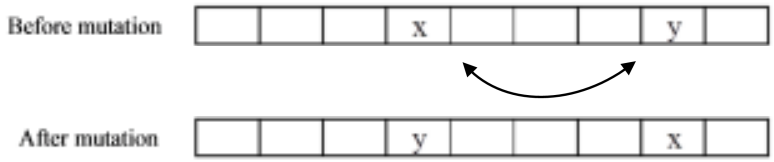


Fig. 4.Mutation by swapping two genes in the chromosome

4.0 SCHEDULE BUILDER

A feasible schedule consists of three types: semi-active, active and non-delay schedule [21]. The optimal solution of the scheduling always lies in the active schedule with no permissible left shift in the schedule is possible as illustrated in Fig. 5. Experimental results from Yamada and Nakano [22] pointed out that recombination of such active schedule chromosome would produce good solutions. Fig. 5 illustrates the interaction of the schedules.

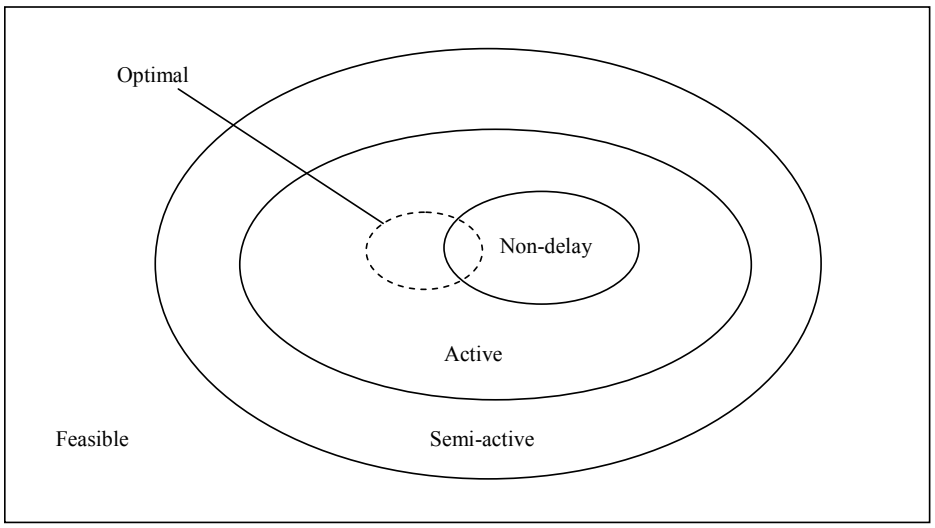
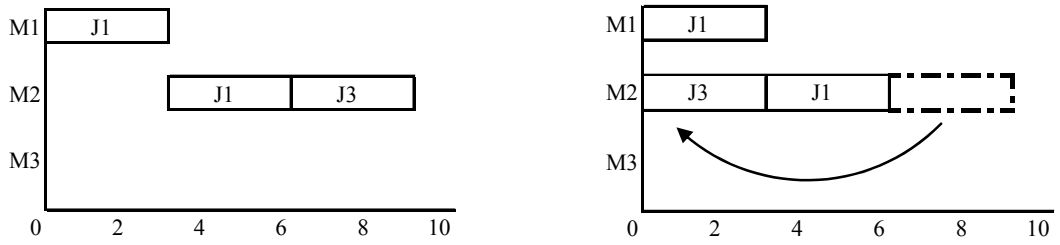


Fig. 5. Relationship of semi-active, active, and non-delay schedules

In order to build an active schedule chromosome, the chromosome needs to be decoded into feasible schedule. This is achieved by constructing a schedule builder that performs a simple local search. Scheduling can be built by selecting the gene of the chromosome from left to right and inserting into schedule with a simple schedule builder to find the earliest completion time and then deleted from the chromosome [23]. While placing the job in the schedule, it must meet the technological requirement and precedence constraints. During the decoding process, an operation

will fill the first possible blank in the interval time before appended as last operation in the machine. Fig. 6(a) illustrates the scheduling without local search and the job will be placed by following the sequences encoded in the chromosome [1 1 3 ...]. Applying the local search will detect a shift which permits job 3 to be placed before job 1 in machine 2 (Fig. 6(b)). Left shifted operation causes the corresponding shift in the chromosome to be modified to [1 3 1 ...].



(a) Semi active schedule without simple local

(b) Active schedule after simple local search

Fig. 6. Local search procedure

5.0 COMPUTATIONAL RESULTS

5.1Experiment Setup

The benchmarks are taken from the selected instances from OR library [24] with different sizes. The ft10 and ft20 considered as difficult computational problem especially ft10 problem referred as "notorious" problem because it remains unsolved for 20 years. The ten tough problems proposed by Applegate and Cook [25] are more difficult than the ft10 problem. The table shows the selected instances from the library with their best known solutions (BKS). Note that instances with asterisks are a part of ten tough problems.

Table 2. Instances of JSSP

Instances	Size			BKS
	Jobs	x	Machines	
ft 06	6	x	6	55
ft 10	10	x	10	930
ft 20	20	x	5	1165
la 01	10	x	5	666
la 10	15	x	5	958
la 21*	15	x	10	1046
la 27*	20	x	10	1235
la 38*	15	x	15	1196

Due to the different structure of the JSSP, the number of individuals in the population is calculated based on the number of operations. Some problems have very large solution space which contains many variables and large range of permissible values for the solutions. Therefore a fixed population is probably not enough because it simply doesn't represent large enough space to sample the solution space, so the population will vary based the size of instances. The other parameters are fixed for all problems (Table 3). The tests were run 30 independent times.

Table 3. Parameters for GA

Parameter	Value
Number of individuals	Twice of the total operations
Selection	Stochastic Universal Sampling (SUS)
Crossover rate	0.7
Mutation rate	0.4
Elitism	Replace 10% best offspring to the previous population
Maximum number of generations	500

5.2 Discussion of the Results

In the Fig. 7, the average computational times for the unsolved problems (ft10, ft20, la21, la27, and la38) are polynomial function of the number of parents for crossover. Multi parents are recombined to produce a single child resulting in a decreasing rate of computational time with the increasing number of parents.

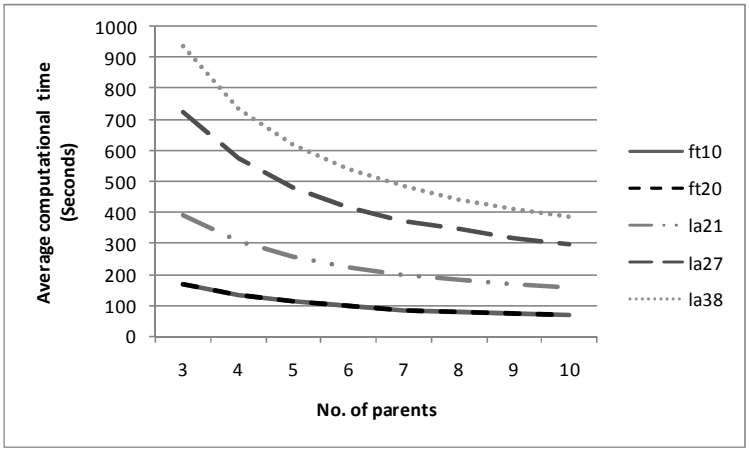


Fig. 7. Average computational times versus number of parents for ft10, ft20, la21, la27, and la38 instances

From Fig. 8, on average, ft10, ft20, la21, la27, and la38 their average deviation (in %) with respect to the BKS are increasing when the number of parents for crossover increases.

It can be concluded from Fig. 7 and Fig. 8 that the larger numbers of parents for crossover do not necessarily provide good quality for total solutions although the computational times are reduced. In order to provide a measurement for suitable number of parents in recombination, we select the best 3 solutions in the selected instances and this is illustrated in Fig. 9. It shows that the top 3 best solutions are mostly between 3-6 numbers of parents for crossover which mean the number of parents that are larger than 6 are less likely to produce good solutions.

To illustrate the effectiveness, the proposed multi parents crossover is compared with the other algorithms such as: Greedy Randomized Adaptive Search Procedure (GRASP) [26] and Priority Rule based Genetic Algorithm (P-GA) [27]. It is noted that P-GA uses two parents crossover. Table 4 lists for each test instances, its name, size (number of jobs and number of machines) , the best known solutions (BKS) and the best solutions found by EPPX (Best) with the multi parents that obtain the best solutions (MP), GRASP, and P-GA. It can be seen from Table 4 that for small problems, all the algorithms obtained the BKS. For relatively large problems, the multi parents EPPX is able to perform better in la21 and la27 compared to GRASP and P-GA.

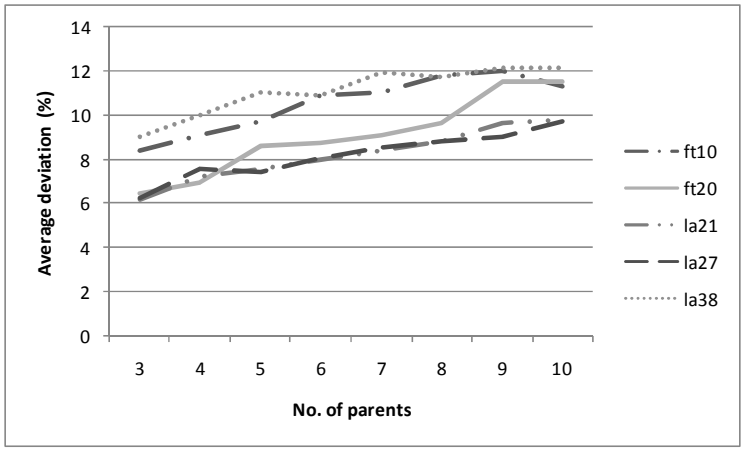


Fig. 8. Average deviations versus number of parents for ft10, ft20, la21, la27, and la38 instances

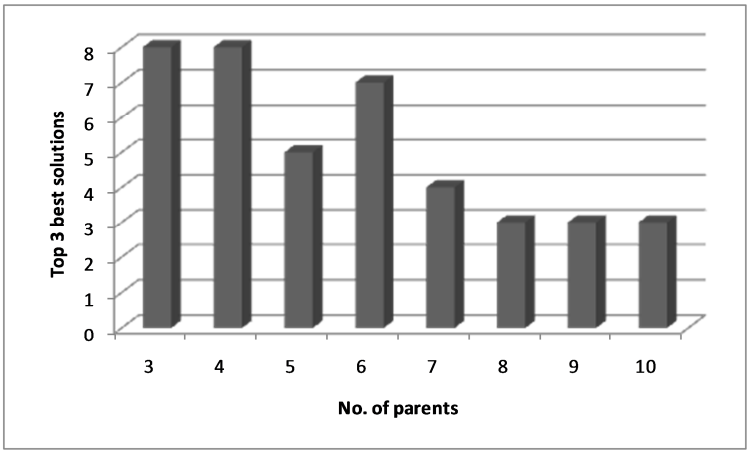


Fig. 9. Top 3 best solutions that acquired by different number of parents

Table 4. Comparison with different methods

Instances	Size	BKS	EPPX		GRASP	P-GA (using 2 parent crossover)
			Best	MP		
ft06	6 x 6	55	**55**	3-10	**55**	**55**
ft10	10 x 10	930	950	6	938	960
ft20	20 x 5	1165	1204	3	1169	1249
la01	10 x 5	666	**666**	3-10	**666**	**666**
la10	15 x 5	958	**958**	3-10	**958**	**958**
la21	15 x 10	1046	**1082**	3	1091	1139
la27	20 x 10	1235	**1287**	3	1320	1378
la38	15 x 15	1196	1268	3	**1267**	1296

6.0 CONCLUSION

This paper presents a multiparents crossover EPPX for JSSP. Various sizes instances were tested to evaluate the performance of EPPX. Experimental results show that the proposed multi parents crossover with different number of parents able are to achieve better or comparablesolutions. It is found that the number of parents between 3 to 6 is able to obtain good solutions for JSSP.

The significant improvement is obtained for difficult problems la21 and la27 and it shows that multi parents crossover is suitable to be implemented in a GA instead of using two parents crossover.In future works, suitable number of parents for crossover is used in conjunction with more efficient local search to increase efficiency of GA to solve JSSP.

ACKNOWLEDGMENTS

This work has been supported by University of Malaya Research Grant Scheme UMRG-RG116-10AFR.

REFERENCES

[1] S. Ólafsson, “Metaheuristics”. *Handbooks in operations research and management science*, Vol. 13,2006, pp. 633-654.

[2] R. Cheng, M. Gen, Y. Tsujimura, “A Tutorial Survey of Job-Shop Scheduling Problems using Genetic Algorithms-I. Representation”, *Computers & Industrial Engineering*, Volume 30, Issue 4, September 1996, pp. 983-997.

[3] L. Davis, “Job shop scheduling with genetic algorithms”, Proceedings of the *1st International Conference on Genetic Algorithms*,1985, pp. 136-140.

[4] A. Jones, L.C. Rabelo, A.T. Sharawi, “Survey of Job Shop Scheduling Techniques”, *Wiley Encyclopedia of Electrical and Electronics Engineering*, 27 December 1999.

[5] E. Nowicki and C. Smutnicki,“A fast taboo search algorithm for the job shop problem”. *Management Science*, vol. 42, issue 6, 1996, pp. 797-813.

[6] J.F. Goncalves, J.J de M. Mendes and M.G.C. Resende, "A Hybrid Genetic Algorithm for the Job Shop Scheduling Problem", European Journal of Operational Research, Volume 167, Issue 1, 16 November 2005, pp. 77-95.

[7] B.J. Park, H.R. Choi and H.S. Kim, "A Hybrid Genetic Algorithm for the Job Shop Scheduling Problems", *Computer & Industrial Engineering*, Volume 45, Issue 4, December 2003, pp. 597-613.

[8] K.S.N. Ripon, "Hybrid Evolutionary Approach for Multi-Objective Job-Shop Scheduling Problem", *Malaysian Journal of Computer Science*, Volume 20, No.2, 2007.

[9] M. Watanbe, K. Ida, M. gen, "A Genetic Algorithm with modified Crossover Operator and Search Area Adaptation for the Job Shop Scheduling Problem",*Computers & Industrial Engineering*, Volume 48, Issue 4, June 2005, Pages 743–752.

[10] K.S. N. Ripon, N.H. Siddique and J. Torresen, "Improved Precedence Preservation Crossover for Multi-Objective Job Shop Scheduling Problem", *Evolving Systems*, Volume 2, Issue 2, June 2011, pp. 119-129.

[11] R. Cheng, M. Gen and Y. Tsujimura, "A Tutorial Survey of Job-Shop Scheduling Problems using Genetic Algorithms, Part II: Hybrid Genetic Search Strategies", *Computers & Industrial Engineering*, Volume 36, Issue 2, April 1999, pp. 343-364.

[12] H. Mühlenbein and H.M. Voigt," Gene pool recombination in genetic algorithms", In Proc. of the *Metaheuristics Conference*, 1995. Kluwer Academic Publishers, Norwell, USA.

[13] A. E. Eiben and , C. H Van Kemenade," Diagonal crossover in genetic algorithms for numerical optimization", *Control and Cybernetics*, Vol. 26, 1997, pp. 447-466.

[14] A. Wu, P.W.M. Tsang, T.Y.F. Yuen, and L.F. Yeung,"Affine invariant object shape matching using genetic algorithm with multi-parent orthogonal recombination and migrant principle", *Applied Soft Computing*, vol. 9, issue 1, 2009, pp. 282-289.

[15] S. Tsutsui & L.C. Jain ," On the effect of multi-parents recombination in binary coded genetic algorithms", In *Knowledge-Based Intelligent Electronic Systems, 1998. Proceedings KES '98. 1998 Second International Conference* on (Vol.3, pp. 155 – 160). IEEE.

[16] S. Tsutsui, M. Yamamur, and T. Higuchi,"Multi-parent recombination with simplex crossover in real coded genetic algorithms", In Proceedings of the *genetic and evolutionary computation conference*, Vol. 1,1999, pp. 657-664.

[17] A. E. Eiben, P. -E. Raué and Zs. Ruttkay, "Genetic Algorithms with Multi-Parent Recombination*", Parallel Problem Solving from Nature — PPSN III.Lecture Notes in Computer Science*, 1994, Volume 866/1994, 78-87.

[18] C.K. Ting, C.H. Su and C.N. Lee, "Multi-Parent Extension of Partially Mapped Crossover for Combinatorial Optimization Problems", *Expert Systems with Applications*, volume 37, Issue 3, 15 March 2010, pp. 1879-1886.

[19] C. Bierwirth, Dirk C. Mattfeld and Herbert Kopfer, "On Permutation Representations for Scheduling Problems", *Lecture Notes in Computer Science*, Volume 1141/1996, 1996, pp. 310-318.

[20] C. Bierwirth, "A Generalized Permutation Approach to Job Shop Scheduling with Genetic Algorithms", *OR Spectrum*, Volume 17, Number 2-3, 1995, pp. 87-92.

[21] A. Sprecher, R. Kolisch, and A. Drexl,"Semi-active, active, and non-delay schedules for the resource-constrained project scheduling problem", *European Journal of Operational Research*, vol. 80, issue 1 , 1995, pp. 94-102.

[22] T. Yamada and R. Nakano, "A Genetic Algorithm Applicable to Large-Scale Job-Shop Problems", *Parallel Problem Solving from Nature*, vol. 2 1992, 281-290, Elsevier Science Publishers.

[23] M. Gen, Y. Tsujimura abd E. Kubota, "Solving Job-Shop Scheduling Problems by Genetic Algorithm", *System, Man, and Cybernetics, 1994. 'Humans, Information and Technology', 1994 IEEE international Conference*, vol. 2 pp. 1577-1582.

[24] J. E. Beasley, "OR-Library: Distributing Test Problems by Electronic Mail". *The Journal of the Operational Research Society*, Vol. 41, No. 11 (Nov., 1990), pp. 1069-1072.

[25] D. Applegate and W. Cook, "A Computational Study of the Job-Shop Scheduling Problem". *ORSA Journal on Computing*, Vol. 3, No. 2, Spring 1991, pp. 149-156.

[26] S. Binato, W.J Hery, D.M. Loewenstern, and M.G.C Resende,"A GRASP for job shop scheduling", *Essays and surveys in metaheuristics*, 2001, pp. 59-79.

[27] U. Dorndorf and E. Pesch,"Evolution based learning in a job shop scheduling environment", *Computers & Operations Research*, vol. 22, issue 1, 1995, pp. 25-40.