MULTIRECOMBINATION AND DIFFERENT REPRESENTATION IN EVOLUTIONARY ALGORITHMS FOR THE FLOW SHOP SCHEDULING PROBLEM

BAIN M., PANDOLFI D.,VILANOVA G.
Proyecto UNPA-29/B017[1]
División Tecnología
Unidad Académica Caleta Olivia
Universidad Nacional de La Patagonia Austral
Ruta 3 Acceso Norte s/n
(9011) Caleta Olivia – Santa Cruz - Argentina
e-mail: {mebain@infovia.com.ar, dpandolfi@uaco.unpa.edu.ar, gvilanov@satlink.com}
Phone: + 54 297 854888 / 297 4550654
Fax : +54 297 854888

GALLARD R.
Proyecto UNSL-338403[2]
Departamento de Informática
Universidad Nacional de San Luis
Ejército de los Andes 950 - Local 106
(5700) - San Luis – Rep. Argentina
e-mail: rgallard@unsl.edu.ar
Phone: + 54 652 20823
Fax : +54 652 30224

ABSTRACT

The flow shop scheduling problem (FSSP) has held the attention of many researchers. In a simplest usual situation, a set of jobs must follow the same route to be executed on a set of machines (resources) and the main objective is to optimize some performance variable (makespan, tardiness, lateness, etc.). In the case of the makespan, it have been proved that when the number of machines is greater than or equal to three, the problem is NP-hard.

EC is an emergent research field, which provides new heuristics to problem optimization where traditional approaches make the problem computationally intractable, is continuously showing its own evolution and enhanced approaches included latest multi-recombinative methods involving *multiple crossovers per couple* (MCPC) and *multiple crossovers on multiple parents* (MCMP).

The present paper discusses the new multi-recombinative methods and shows the improvement of performance of enhanced evolutionary approaches under *permutation* and *decode* representation. Results of the methods proposed for each chromosome representation are here contrasted and results are shown.

KEYWORDS: Evolutionary algorithms, Multiple Crossovers, Multiple Parents, Scheduling.

[1] The Research Group is supported by the Universidad Nacional de La Patagonia Austral.
[2] The Research Group is supported by the Universidad Nacional de San Luis and the ANPCYT (National Agency to Promote Science and Technology).

1. INTRODUCTION.

Evolutionary algorithms have been successfully applied to solve flow shop scheduling problems [18,20]. Important research has been made in evolutionary computation, to maintain a good balance between exploration and exploitation of solutions while searching in a problem space. New trends to enhance evolutionary algorithms make use of multirecombination, allowing multiple crossovers on the selected pair of parents. This novel approach is called *multiple-crossovers-per-couple* (MCPC) [5]. A later extension, called *multiple-crossovers-on-multiple-parents* allows multirecombination on more than two parents [7], where instead of applying crossover once on a pair of parents this feature applies n_1 crossover operation on a set of n_2 parents. An important property of this approach is revealed when observing the final population: all individuals are much more centred surrounding the optimum. This is an important issue when an application requires provision of multiple alternative near-optimal solutions.

As FSSP is a permutation schedule problem, a permutation representation of chromosomes is a natural one. In this case adequate genetic operators should be used to obtain valid offspring. Another way to face a problem involving permutations is by using *decoders* (Grefenstette [10]). Under this approach a chromosome gives instructions to a decoder on how to build a feasible solution.

This paper compares the performance of evolutionary algorithms under permutation and decode representation with diverse recombination approaches.

2. EVOLUTIONARY ALGORITHMS IN SCHEDULING PROBLEMS.

Evolutionary algorithms (EAs) have been successfully applied to solve flow-shop problems. Because of the flow shop is in essence a permutation schedule problem, a permutation of jobs can be used as the representation scheme of chromosomes. This implies the use of well-suited genetic operators (crossover and mutation) in order to provide always valid offspring. Another way to face the problem is by using decoders, in which case the traditional recombinative operators (one-point-crossover, uniform crossover) could be applied.

2. 1. PERMUTATION REPRESENTATION.

Tsujimura et al [20] provided evidence of the performance of genetic algorithms (GAs) contrasted with conventional approaches using well known crossover operators such as *partially-mapped crossover* (PMX) [9], *order crossover* (OX) [2] and *cycle crossover* (CX), [15]. Because of the flow-shop problem is essentially a permutation schedule problem, a permutation can be used as the representation scheme of chromosomes, which is the natural one for a sequencing problem. For example, let the k^{th} chromosome be: v_k = [10 12 9 11]. This means that the job sequence is $j_{10}, j_{12}, j_9, j_{11}$. The permutation representation, also called *order representation*, may lead to illegal offspring if the traditional one-point crossover operator is used. Consequently, during the past decades, several crossover operators have been proposed for permutation representation, such as the above mentioned PMX, OX and CX.

Reeves [18] proposed a hybrid approach, which inserts a chromosome as a seed in the initial population generated by the NEH heuristic algorithm [14]. He suggested genetic operators in his implementation what he called *one-cut-point crossover* (OCPX). It consists of choosing one-cut-point randomly, and then taking the pre-cut section of the first parent and filling up the offspring by taking, in the order they appear, legal genes from the second parent. He tried two types of mutation. The first one is an *exchange mutation,* which was a simple exchange of two genes of the chromosome, chosen at random. The other, a *shift mutation* was a shifting of one gene (chosen randomly) to the right or left a random number of places. After a few experiments, Reeves observed that shift mutation seemed to be better than exchange. Reeves tested his GA on Taillard's benchmarks [19] and concluded that simulated annealing algorithms and GAs produce comparable results for the *flow-shop sequencing* problem for most sizes and types of problems, but GAs perform relatively better for large problems and reach a near-optimal solution more quickly.

2.2. DECODER REPRESENTATION.

Another way to face a problem involving permutations is by using *decoders* (Grefenstette [10]). Under this approach a chromosome gives instructions to a decoder on how to build a feasible solution. Even if decoders are mainly used in other constrained problems, we discuss a decoding scheme based on ordinal representation because it is easy to implement and produces feasible offspring under different conventional crossover methods making unnecessary the use of penalties or repair functions. Here a chromosome is an n-vector where the i^{th} component is an integer in the range 1.. (n-i+1). The chromosome is interpreted as a strategy to extract items from an ordered list L and build a permutation. For example if L = (1, 2, 3, 4) then chromosomes [3121] and [2211] are interpreted as permutations [3142] and [2314], respectively.

3. MULTIRECOMBINATION

In EAs the common approach is to operate once on each mating pair after selection. Such procedure is known as the SCPC (Single Crossover Per Couple) approach. But in nature when the mating process is carried out, crossover is applied many times and the consequence is a multiple and variable number of offspring.

Multiple crossover per couple (MCPC) [5] is a novel crossover method. It was applied to optimize classic testing functions and some harder (non-linear, non-separable) functions. For each mating pair MCPC allows a variable number of children. It is possible to choose for insertion in the next generation the best, a randomly selected or all of the generated offspring. In those earlier works it was noticed that in some cases MCPC found better results than those provided by SCPC. Also a reduced running time resulted when the number of crossovers per couple increased, and best quality results were obtained allowing between 2 and 4 crossover per couple. However in some cases the method increased the risk of premature convergence due to a loss of genetic diversity. To overcome this problem further successful approaches were undertaken [6]. Moreover, seeking for exploitation of a greater sample of the problem space, as an extension the multi-recombination can be applied to a set of more than two parents.

In Eiben's *multiparent* (MP) approach [3,4], offspring creation is based on a larger sample from the search space and consequently larger diversity is supplied. This can help to avoid premature convergence. Eiben used three scanning crossover (SX) methods; *uniform scanning crossover*, *occurrence based scanning* and *fitness based scanning* generating a single offspring. In the first method each gene in the child is provided from any of the corresponding genes in the parents with equal probability. The second selects that gene value which occurs more frequently in a particular position of the parent's chromosomes. The third chooses the value to inherit being proportional to the fitness value of the parents. On different function optimization different versions of scanning crossover showed different behaviour.

Following this idea and to improve performance, *multiple crossovers on multiple parents* (MCMP) allows multiple recombination of multiple parents under *scanning crossover* (SX), expecting that exploitation and exploration of the problem space be adequately balanced .

4. EXPERIMENTS AND RESULTS

Experiments were divided into two main groups each one corresponding to a representation scheme.

In the case of *permutations* and according to Tsujimura's and Reeves's works we tested six different approaches contrasting the conventional and multiple recombination methods:

- *SCPC-T*, Tsujimura,s approach. Uses OX, CX or PMX.
- *SCPC-R*, Reeves's approach. Uses OCPX and insertion of seeds in the initial population.
- *MCPC-T*, Multiple crossovers per couple using OX, CX or PMX.
- *MCPC-R*, Multiple crossovers per couple using OCPX and insertion of seed in the initial population.
- *MP*, Multiple parents using *controlled uniform scanning crossover* (CUSX).
- *MCMP*, multiple crossovers on multiple parents using (CUSX).

A total of 36 different experiments were designed. For each instance a series of ten runs was performed. Experiments consisted of varying parameters such as the number of crossover n_1, the number of parents n_2, the number of offspring to be inserted in the next generation n_3, the number of seeds used in the initial population n_4, and the type of crossover used. Besides all the EAs used the following parameter settings:

Population size	100
Crossover Probability	0.65
Mutation Probability	0.01
Maximum No. of Generations	100
Elitism	Yes

In the case of *decoders* the experiments also used different crossover methods. They were *one point crossover* (OPX), *uniform crossover* (UX), *uniform scanning crossover* (USX), *fitness based scanning crossover* (FBSX) and suitable combinations of them. A single mutation operator, *big creep mutation* (BCM) was used. Six different approaches contrasting the conventional and multiple recombination methods were used:

- *SCPC*: Uses OPX and BCM
- *MCPC-1*: $n_1 = 2$, $n_2 = 2$, using combined OPX and UX alternatively for each crossover operation, BCM, and a single canonical decoding list *L*.
- *MCPC-2*: $n_1 = 2$, $n_2 = 2$, using combined OPX and UX alternatively for each crossover operation, BCM, and randomly generated decoding lists.
- *MCMP-1*: $n_1 = 4$, $n_2 = 4$, using USX, BCM, and randomly generated decoding lists.
- *MCMP-2*: $n_1 = 4$, $n_2 = 4$, using FBSX, BCM, and randomly generated decoding lists.
- *MCPC-3*: $n_1 = 4$, $n_2 = 4$, using combined USX and FBSX alternatively for each crossover operation, BCM, and randomly generated decoding lists.

All the 12 above-indicated approaches were tested for four Taillard´s benchmarks [19] for the flow shop problem. Given *n* jobs and *m* machines we run the experiments for the many instances of each of the following (*n* x *m*) problem sizes: 20x5, 20x10, 50x5, 50x10. For each instance a series of at least ten runs was performed. All the EAs used the following parameter settings:

Population size	100
Crossover Probability	0.3
Mutation Probability	0.1
Maximum No. of Generations	600
Elitism	Yes

As an indication of the performance of the algorithm the following relevant variables were chosen:

- **Ebest** : It is the percentile error of the best found individual when compared with the benchmark upper bound for the optimal makespan. It gives us a measure of how far we are from that upper bound.

- **Epop** : It is the percentile error of the population mean fitness when compared with benchmark upper bound for the optimal makespan. It tell us how far the average individual is from that upper bound makespan benchmark.

Tables 1, and 2 show the results obtained for the considered performance variables for the distinct problem sizes under each approach. In these tables mean values for the performance variables from the corresponding selected instances (four to ten for each problem size) and experiments are indicated. In the first column, referring to the approach used, remarks (p) and (d) indicates permutation or decode representation, respectively. Columns 2 to 5 show results corresponding to different problem sizes. Columns 6 and 7 give average results for 5 and 10 machine problems while column 8 shows global average results. Boldfaced values are the best achieved. Figures 1, 2 and 3 show the global results of this work.

Approach	20x5	50x5	20x10	50x10	5 Machines	10 Machines	Average
MCMP (p)	5.87	3.73	9.42	10.82	4.80	10.12	7.46
MCMP-2 (d)	**4.83**	**3.17**	12.93	11.16	**4.00**	12.04	8.02
MCPC-R (p)	6.07	3.55	**9.12**	**10.42**	4.81	**9.77**	**7.29**
MCPC-2 (d)	5.31	3.64	13.96	12.00	4.47	12.98	8.73
SCPC-R (p)	7.25	4.88	11.60	12.63	6.07	12.12	9.09
SCPC (d)	8.37	5.41	18.25	14.29	6.89	16.27	11.58

Table 1. Mean Ebest values for each problem size under each approach

Table 1 indicates that the difficulty to find near optimal solutions increases as long as the ratio *n*/*m* decreases. Consequently, for any of the evolutionary approaches considered, a better performance is observed for problem sizes including 5 machines than those including 10 machines.
As it can be observed best results corresponds to multirecombinated approaches (MCMP-2, and MCPC-R), but permutation representation outperforms decode representation. And this behaviour is clearer in the average for all problem sizes.

Approach	20x5	50x5	20x10	50x10	5 Machines	10 Machines	Average
MCMP (p)	**7.87**	**4.40**	**10.80**	11.30	**6.14**	11.05	**8.59**
MCMP-2 (d)	10.47	9.88	20.27	20.97	10.17	20.62	15.39
MCPC-R (p)	9.20	4.60	10.97	**11.10**	6.90	**11.04**	8.97
MCPC-2 (d)	15.37	11.98	25.24	23.01	13.68	24.13	18.90
SCPC-R (p)	12.30	6.33	15.30	14.85	9.32	15.08	12.20
SCPC (d)	27.04	18.33	36.50	28.37	22.68	32.44	27.56

Table 2. Mean Epop values for each problem size under each approach

As it can be observed in table 2 best *Epop* results corresponds again to multirecombinated approaches (MCMP and MCPC-R), both corresponding to permutation representation.

5.CONCLUSIONS.

This work introduced latest variants of multi-recombinative approaches, MCPC and MCMP, applied to the Flow Shop Scheduling Problem. Both approaches were tested under different chromosome representation: permutations and decoders. Two groups (one per representation) of six variants of considering SCPC, MCPC and MCMP, with adequate crossover and mutation operations, and parameter settings were contrasted. All the evolutionary approaches were tested for many instances of selected FSSP problem sizes.

At the light of these results we can conclude that:

- All methods including multirecombination outperform SCPC regarding quality of results (best and average individuals) and speed to find near optimal solutions.
- Under permutation representation better results than those obtained under decode representation, are provided.

As a consequence, further research will include self adaptation of parameters, population size and probabilities of crossover with permutation representation.

6. ACKNOWLEDGEMENTS.

We acknowledge the cooperation of the project group for providing new ideas and constructive criticisms. Also to the Universidad Nacional de San Luis, the Universidad Nacional de La Patagonia Austral, and the ANPCYT which we receive continuous support from.

7. REFERENCES

[1] Campbell H., Dudek R., Smith M., *A heuristic algorithm for the n job m machine sequencing problem*, Management Science 16, 1970, 630-637.

[2] Davis, L. *Applying adaptive algorithms to domains*. In proceeding of the International Joint Conference on Artificial Intelligence. pp. 162-164. 1985.

[3] Eiben A.E., Raué P-E., and Ruttkay Zs., *Genetic algorithms with multi-parent recombination.* In Davidor, H.-P. Schwefel, and R. Männer, editors, Proceedings of the 3rd Conference on Parallel Problem Solving from Nature, number 866 in LNCS, pages 78-87. Springer-Verlag, 1994.

[4] Eiben A.E., van Kemenade C.H.M., and Kok J.N., *Orgy in the computer: Multi-parent reproduction in genetic algorithms.* In F. Moran, A. Moreno, J.J. Merelo, and P. Chacon, editors, Proceedings of the 3rd European Conference on Artificial Life, number 929 in LNAI, pages 934-945. Springer-Verlag, 1995.

[5] Esquivel S., Leiva A., Gallard R., - *Multiple Crossover per Couple in Genetic Algorithms,* Proceedings of the Fourth IEEE Conference on Evolutionary Computation (ICEC'97), pp 103-106, ISBN 0-7803-3949-5, Indianapolis, USA, April 1997.

[6] Esquivel S., Leiva A., Gallard R.: *Couple Fitness Based Selection with Multiple Crossover per Couple in Genetic Algorithms.* Proceedings of the International Symposium on Engineering of Intelligent Systems (EIS´98), La Laguna, Tenerife, Spain Vol. 1, pp 235-241, ed. E.Alpaydin. Publishesd by ICSC Academic Press, Canada/Switzerland, February 1998.

[7] Esquivel S., Leiva H.,.Gallard R., *Multiple crossovers between multiple parents to improve search in evolutionary algorithms,* Proceedings of the 1999 Congress on Evolutionary Computation (IEEE). Washington DC, pp 1589-1594.

[8] Esquivel S., Leiva H.,.Gallard R., *Self-Adaptation of Parameters for MCPC in Genetic Algorithms,* Proceedings of the 4th Congreso Argentino de Ciencias de la Computación (CACiC'98), pp 419-425. Universidad Nacional del Comahue, Argentina, Octubre 1998.

[9] Goldberg D. E., Lingle R. – *Alleles, Loci and the TSP* – Proceedings of the First International Conference on Genetic Algorithms, Lawrence Erlbaum Assoc., NJ, 1985.

[10] Grefenstette J. J., Gopal R., Rosmaita B., Van Gutch D. – *Genetic Algorithm for the TSP* - Proceedings of the 1st Int. Conf. on Genetic Algorithms, Pittsburgh , PA, 1991.

[11] Gupta J., *A functional heuristic algorithm for the flowshop scheduling problem,* Operational Research Quarterly 22, 1971,39-48,.

[12] Michalewicz, M., *Genetic Algorithms + Data Structures = Evolution Programs.* Springer, third revised edition, 1996.

[13] Morton T., Pentico D., *Heuristic scheduling systems,* Wiley series in Engineering and technology management. John Wiley and Sons, INC, 1993.

[14] Nawaz M., Enscore E., Ham I., *A heuristic algorithm for the m-machine n-job flow shop sequencing problem.* Omega vol II, pp 11-95, 1983.

[15] Oliver I. M., Smith D. J., Holland J. R. C. – *A Study of Permutation Crossover operators on the Travelling Salesman Problem* - Proceedings of the 2nd Int. Conf. on Genetic Algorithms, pp 224-230. Lawrence Erlbaum Assoc., NJ, 1987.

[16] Palmer D., *Sequencing jobs through a multistage process in the minimum total time-* A quick method of obtaining a near optimum, Operational Research Quarterly 16, 101-107, 1965.

[17] Pinedo Michael (1995) – *Scheduling- Theory , Algorithms, and Systems.* Prentice Hall International in Industrial and System Engineering.

[18] Reeves C., *A genetic algorithm for flow shop sequencing*, Computers and Operations Research, vol 22, pp5-13, 1995.

[19] Taillard, E. *Benchmarks for basic scheduling problems*, European Journal of Operational Research, vol. 64, pp. 278-285,1993. http://wwww.idsia.ch/~eric/problems.dir/ordonnancement.dir/ordonnancement.html).

[20] Tsujimura Y., Gen M., Kubota E., *Flow shop scheduling with fuzzy processing time using genetic algorithms.* The 11th Fuzzy Systems Symposium pp 248-252. Okinawa. 1995.