MULTIPLE CROSSOVERS ON MULTIPLE PARENTS FOR THE MULTIOBJECTIVE FLOW SHOP PROBLEM

Esquivel S. C., Zuppa F., Gallard R.H.

Proyecto UNSL-338403[1]
Departamento de Informática
Universidad Nacional de San Luis (UNSL)
Ejército de los Andes 950 - Local 106
5700 - San Luis, Argentina.
E-mail: {esquivel, fede, rgallard}@unsl.edu.ar
Phone: + 54 652 20823
Fax : +54 652 30224

Abstract

The Flow Shop Scheduling Problem have been tackled using different techniques which goes from mathematical techniques like Branch and Bound to metaheuristics like evolutionary algorithms (EAs). Although in the real world this problem will be found more frequently with more than one objective, most work been done is based on a single objective. Evolutionary algorithms are very promising in this area because the outcome of a multiobjective problem is a set of optimal solutions (the Pareto Front) which EAs can provide in a single run. Yet another advantage of EA's over other techniques is that they are less liable to the shape or continuity of the Pareto Front. In this work, we show three implementations of multiobjective Evolutionary Algorithms. The first one uses Single Crossover Per Couple (SCPC), while the other two use Multiple Crossover on Multiple Parents (MCMP), continuing with previous works[7, 8]. These two methods show an enhancement on the performance of the first method. Details of implementation and results are discussed.

Keywords: Evolutionary Computation, Flow shop scheduling, multiobjective optimization, multirecombination.

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

1 - Introduction

In a multiobjective optimization problem (MOOP), a solution has a number of objective values, one per each optimizing criterion (attributes). As many of these criteria can be in conflict it is impossible to optimize any of the objective functions without degrading some of the remaining criteria. This leads to a decision-making problem for choosing a suitable solution (or set of solutions) according to higher-level organization goals.

Vilfredo Pareto [12] established that there exists a partial ordering in the searching space of a MOOP based on a domination relationship. For instance, in a maximization problem given two solutions $x = (x_1, x_2, ..., x_n)$ and $y = (y_1, y_2, ..., y_n)$, the Pareto criterion says that, x dominates y iff $x_i \geq y_i \ \forall i$ and $\exists\, j$ such that $x_j > y_j$.

In the problem space some solutions will not be dominated by any other solution and they conform the Pareto front, also known as the acceptable set, the efficient points and the Pareto optimal set. Knowledge of the Pareto front is of utmost importance when search is applied before decision making. This information provides to the judgement of a human decision-maker with the trade-offs to establish interactions among different criteria, hence simplifying the decision process to choose an acceptable range of solutions for a multicriteria problem

In our study we are interested in Permutation Flow Shop problems, where jobs must be scheduled in the same order on all machines. Our multiobjective optimization problem can be formulated as follows: Minimize $f_1(\sigma)$ and $f_2(\sigma)$ where sought solutions σ are feasible schedules and

$$f_1(\sigma) = max_{1 \leq k \leq m}\{max_{1 \leq i \leq n}\{C_{ik}\}\} \quad (1) \qquad f_2(\sigma) = \sum_{j=1}^{n} max\{0, C_j - d_j\} \qquad (2)$$

$$T_j = \max\ \{\ 0, L_j\ \}$$

In expression (1) (makespan), C_{ik} stands for the completion time of the last operation of job i in machine k, and in expression (2) (maximum tardiness), C_j indicates the completion time of the last operation of job j and d_j is the corresponding due date. Both problem are NP-Hard, which means that they cannot be easily solved for medium and large instances.

In this work we contrast results of the three proposed algorithms between them and also against the best results provided until now by Basseur [1]. In his work Basseur designed a so called *dynamic mutation Pareto Genetic Algorithm*, where different operators are used simultaneously in adaptive manner, a combined sharing technique is applied, and a hybrid approach combining the EA with local search is implemented. The idea is to know how approaches based only in multirecombination [5, 6 ,7] approximate the best found results, and which is the relative quality between them.

2. Fitness and Elitism in MOOP

The three main differences between a Single Objective Evolutionary Algorithm and a Multiple Objective Evolutionary Algorithm are how the fitness is determined, the inclusion, or not, of a sharing technique, and how the elitism is implemented [2, 3, 4].

The fitness calculation is accomplished using a Pareto Front approach. In this method, the fitness is not an absolute quality measure like in the single objective case, but it represents a quality measure of this individual with respect to the other individuals in the same population. The method for assigning fitness in this class of algorithms is as follows [10, 11]:

1) Each individual is ranked according to the number of individuals by which it is dominated.
2) A sorting of the individuals according to its ranking in the population is made,
3) A fitness is assigned to each individual using a linear function.

Elitism in a multiobjective Evolutionary Algorithm means maintaining the Pareto Front obtained up until the current time (called Current Pareto Front).

3. Alternative Recombination Methods

In what follows we will describe each of the recombination approaches applied on the same benchmarks selected by Basseur. In all our EAs, an auxiliary structure maintains the current Pareto front. To help evolution, all individuals in the current Pareto Front are inserted in the population, after each generation. In other words, after the recombination cycle, offspring go to an intermediate population. Now they are ranked according to dominance. Worst individuals of this intermediate population are replaced by the individuals in the current Pareto front, and the intermediate population becomes the new population. In this way elitism is implemented. Then the fitness calculation proceeds, and if necessary, the current Pareto front is updated. A sharing technique was not implemented. The MCMP variants include the use of a stud (breeding individual) which due to its best characteristics contributes with its genetic material by repeatedly mating the remaining parents in the mating pool.

Multiobjective SCPC

In this approach a simple evolutionary algorithm is implemented. A single crossover is applied to each couple selected with proportional selection.

Multiobjective MCMP – 1

The first multiple crossover operation was implemented in the following way: First it makes as many rankings as objectives are (in this case two). Then it selects a set of parents for each objective using Proportional Selection and then finds the stud for each group. Once the studs are selected, multiple crossovers between each stud and the rest of the individuals of the other group are performed. In this way breeding among different “species” is forced. An auxiliary structure which initially holds the pairs of parents is used to retain the best individuals. Each time a crossover is performed, the corresponding parent is only replaced in the case that the child dominates it. Once all the children are obtained, they are inserted into the next population following this scheme: The algorithm first selects those offspring, which are classified so far, as *globally* non-dominated. If none fulfilling this condition exists then half of the *m* newly generated offspring are inserted, selecting them according to their dominance ranking.

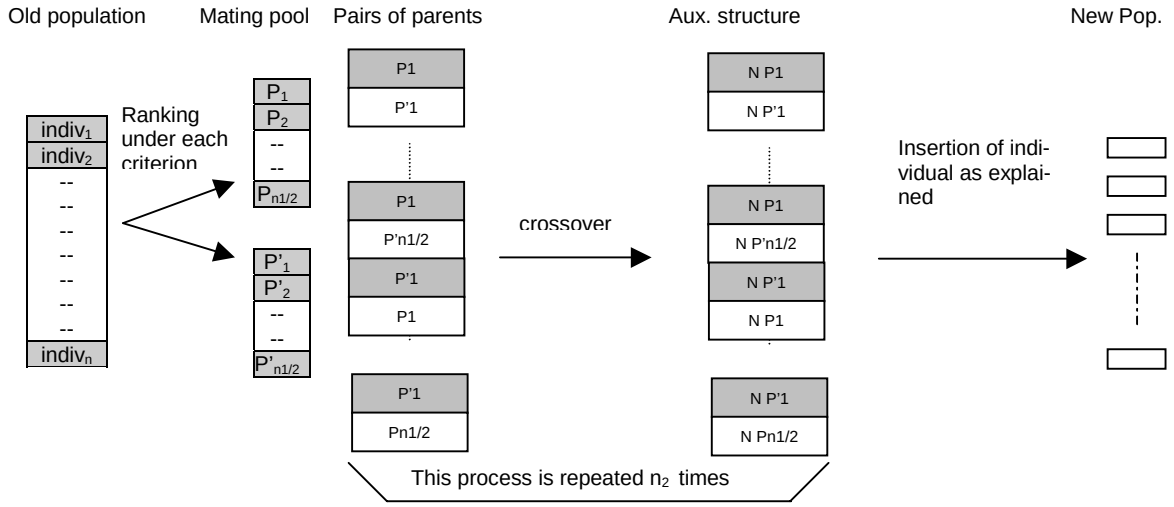

Fig. 1. The multirecombination process in MCMP-1

Multiobjective MCMP – 2

The second multiple crossover operation was implemented in the following way: First it make as many rankings as objetives are (in this case two). Then it selects a set of parents for each objective using Proportional Selection and finds the stud for each group. Once the studs are selected, multiple crossover between each stud and the rest of the individuals in the other groups are performed (until here is the same method as MCMP-2). Once all the children individuals are obtained, they are inserted into the next population following this scheme: The algorithm first selects those offspring, which are classified so far, as *globally* non-dominated. If none fulfilling this condition exists then half of the *m* newly generated offspring are inserted, selecting first those that are non-dominated within the new offspring subset and completing *m/2* insertions by random selection if necessary.

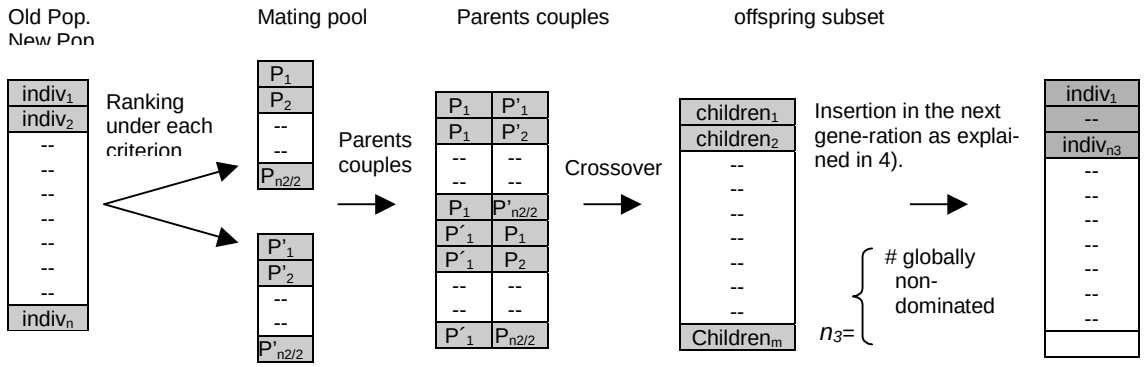

Fig. 2. The multirecombination process in MCMP-2

3 – Contribution as a comparative performance measure

We will use this measure as defined by Basseur [1]. In his paper he says that: "…the contribution of a set of solutions PO_1 relatively to a set of solutions PO_2 is the ratio of non-dominated solutions produced by PO_1.

Let C be the set of solutions in $PO_1 \cap PO_2$
Let W_1 (resp. W_2) the set of solutions in PO_1 (resp. PO_2) that dominate some solutions of PO_2
Let L_1 (resp. L_2) be the set of solutions in PO_1 (resp. PO_2) that are dominated by some solutions of PO_2 (resp. PO_1)
Let N_1 (resp. N_2) be the other solutions of PO_1 (resp. PO_2) : $N_i = POi \setminus (C \cup Wi \cup Li)$
Let PO^* be the set of Pareto solutions of $PO_1 \cup PO_2$. So, $||PO^*|| = ||C|| + || W_1 || + ||N_1|| + ||W_2|| + ||N_2||$
The contribution of the algorithm PO_1 relatively to PO_2 is given by

$$C(PO_1, PO_2) = \frac{\frac{\|C\|}{2} + \|W_1\| + \|N_1\|}{PO^*}$$

… the contribution of the two sets of solutions PO_1 of PO_2 of figure3, is evaluated: solutions of PO_1 (resp. PO_2) are represented by circles (resp. Crosses). We have C(PO_1, PO_2) = 0.7 and C(PO_2, PO_1) = 0.3…"

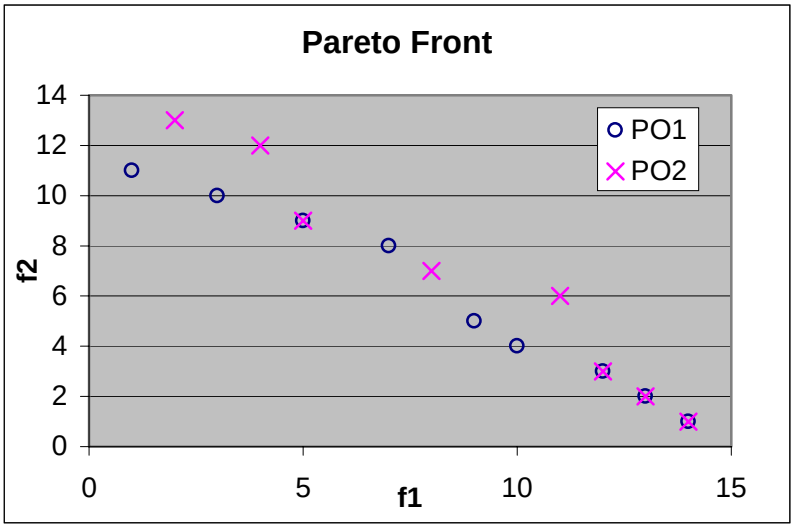

Fig. 3. Contributions example (C = 4, W_1 = 4, W_2 = 0, N_1 = 1 N_2 = 1)

Consequently as C(PO_1, PO_2) + C(PO_2, PO_1) = 1 this measure of performance tell us that two algorithms are equivalent when building the Pareto front if both contributions are equal to 0.5.

4 – Experiments and results

All the experiments have a fixed number of generations, the same rate of probabilities and population size, and as the representation adopted is a permutation of jobs PMX was the crossover method. The following table shows the parameter settings, which were determined as the best after a set of initial trials.

Algorithm	SCPC	MCMP -1	MCMP - 2
Population size	100	100	100
Generations	30000	30000	30000
Crossover	PMX	PMX	PMX
Mutation	SHIFT	SHIFT	SHIFT
Number of Parents	2	4	4
Number of crossovers	1	6	6
Probability of crossover	0.65	0.65	0.65
Probability of mutacion	0.3	0.3	0.3

The following tables and graphics show the contribution values and the Pareto fronts of the corresponding algorithms

	SCPC	MCMP - 1	MCMP - 2
20 x 5	0.000000	**0.500000**	0.000000
20 x 10	0.027027	0.171429	0.263158
20 x 20	0.250000	0.300000	0.232143
Average	0.092333	0.323809	0.165100

Table 1. Comparison between the contributions values of the different EAs implemented and the benchmark.

Table 1 indicates that MCMP-1 reaches the benchmark in the 20x5 instance and it is the best performer in average.

	MCMP - 1	MCMP - 2
20 x 5	**1.000000**	0.750000
20 x 10	0.866667	**0.948718**
20 x 20	0.380000	**0.600000**
Average	0.748889	0.766239

Table 2. Comparison of the contribution values of the MCMP EA's and the SCPC EAs

Table 2 shows that both multirecombined evolutionary algorithms dominate most points created by SCPC.

In figure 4 we can see that for the 20x5 problem size all EAs using MCMP outperformed SCPC. It also can be seen that MCMP-1 is the best performer in this size (this also can be seen in both tables of contribution values).

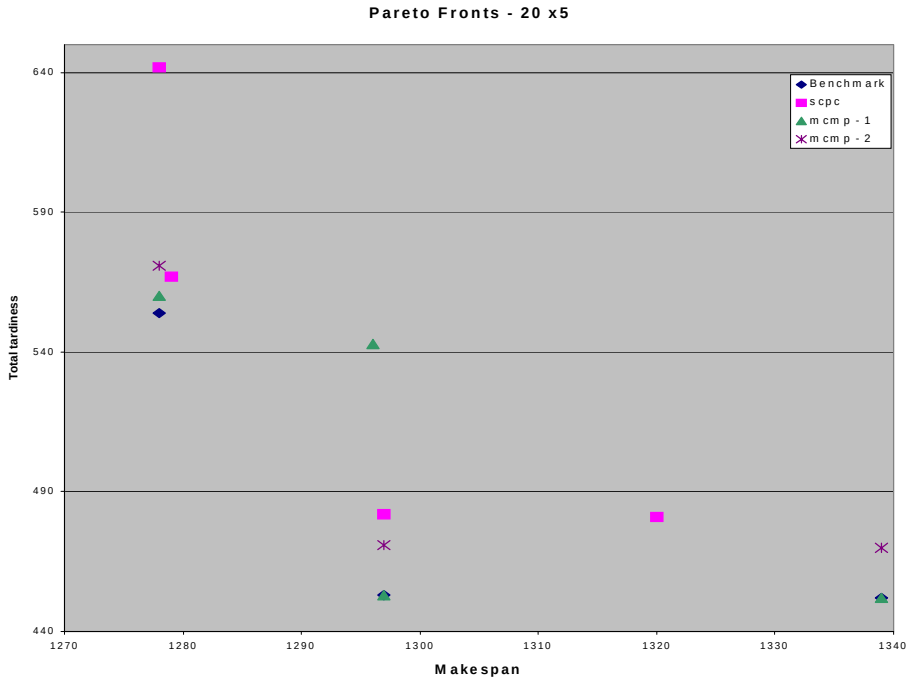

Fig. 4. Pareto fronts built by the alternative algorithms for the 20x5 problem size

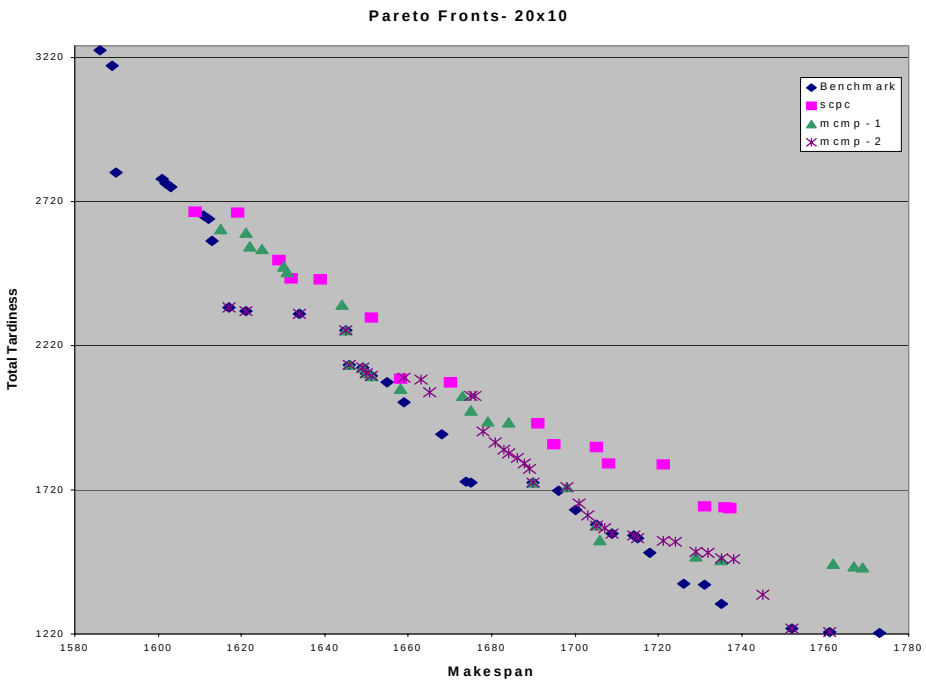

Fig. 5. Pareto fronts built by the alternative algorithms for the 20x10 problem size

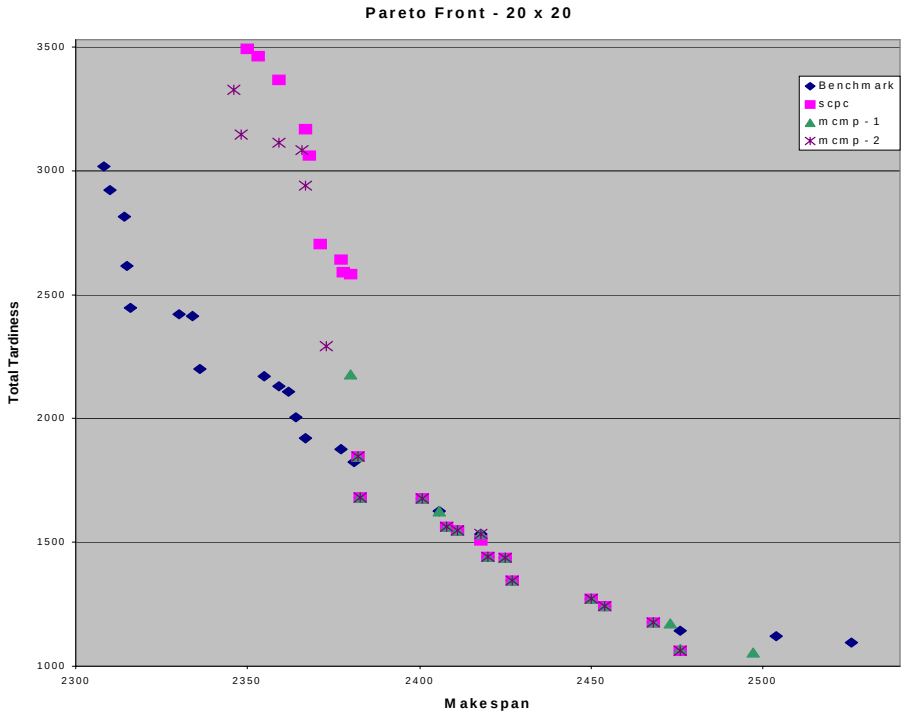

Fig. 6 Pareto fronts built by the alternative algorithms for the 20x20 problem size

In figure 5, we can see the same thing as in figure 4, that is to say the points obtained by the SCPC algorithm are almost always dominated by the EAs using MCMP. In particular MCMP-2 seems to be the algorithm with the best behaviour. This also could be corroborated in the contribution values of table 2

Figure 6 is more difficult to analyze. At first glance, MCMP-1 seems to be the best performer. Nonetheless this EA loses a big portion of the Pareto Front. This is why MCMP-1 provides better results in the first table, but MCMP-2 provides better ones in the second table.

The algorithms that use multirecombination are clearly better performers than the single recombination approach. The reasons for this could be: first, the advantages seen in single objective problems are also applied for multiobjective problems, that is to say that the best individuals of the population are mated with the other individuals more than once, giving the possibility of exploiting them better. The other reason, is that the parents and the two studs chosen for the recombination operation are selected according to different criterias (objectives). This allows that different 'properties' of these individuals are mixed more than once. The comparison between MCMP-1 and MCMP-2 is not so clear, because the results are very much alike. Nonetheless, MCMP-2 performs better than MCMP-1, particularly in larger problem sizes, besides been faster than MCMP-1. If the first instance, which only provides 3 solutions, is not taken into account the other sizes show a better performance in the graphics, where the front is larger and in the contribution results are better. The benchmark provides very good results for many reasons, but the main one in our point of view is the inclusion of two mutation operators alterna-

tively applied. We conjecture that because in past experiments using both Random Exchange Mutation and Shift Mutation it was observed that for some sizes, the first method was much better but for other sizes it was the opposite.

5 - Conclusions

An important issue in this work is that no sharing technique was used. The application of a sharing technique is, according to many authors, fundamental in multiobjective optimization. In these experiments we did not use it because the objective was to determine the raw potential of multirecombination when dealing with MOOP for Flow Shop Scheduling.

As a result of this work we determined that all algorithms using MCMP behave better than the algorithm that uses SCPC. Both, in the contribution values and in the observation of the graphics, it can be seen that these solutions are closer to the solutions provided by the evolutionary algorithm that was taken as benchmark.

If we want to establish a comparison between the algorithms that use MCMP, considering only the first table (contribution with respect to the benchmark) then MCMP-1 shows the best behaviour. But if we consider the second table, (contribution with respect to the SCPC algorithm) MCMP-2 behaves better. If we look at the graphics, we see that MCMP-2 behaves poorly in the smallest instance, but it is the best in the next ones.

Another point to remark, is the relative speed of these algorithms. MCMP-2 is at least five times faster than MCMP-1. This is due to the number of new individuals that are inserted each time a crossover operation is performed.

Future work will be devoted to include sharing techniques and adaptive mutation methods together with multirecombination to improve performance.

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 and the ANPCYT from which we receive continuous support.

7 - Bibliography

1. Basseur M., Design of multiobjective evolutionary algorithms: Aplication to the Flow Shop scheduling problem. Congress on Evolutionary Computation, 2002.
2. Coello Coello C., An updated survey of evolutionary multiobjective optimization, Proceedings of the 1999 Congress on Evolutionary Computation (IEEE). Washington DC, pp 3-13.
3. Coello Coello C., A comprehensive survey of evolutionary-based multiobjective optimization techniques. http://www.lania.mx/~ccoello/EMOO

4. Deb K., Multi-objective genetic algorithms: problem difficulties and construction of test problems. Evolutionary Computation, 1999 by the MIT.
5. Eiben A., Raue P.-E., Ruttkay Zs., Genetic Algorithm with multiparent recombination, Proc of Parallel Solving from Nature Nr 3, LNCS 866, Springer Verlag, pp 78-87, 1994.
6. Esquivel S., Ferrero S.,.Gallard R. Multirecombinated evolutionary algorithms to solve multiobjective job shop scheduling. Proceeding of Primer Congreso Argentino de Ciencias de la Computación, 2000.
7. Esquivel S., Zuppa F., Gallard R., Multirecombinated Evolutionary Algorithms for the Flow Shop Scheduling Problem. PPSN IV. P. 263, 272.
8. Esquivel S., Zuppa F., Gallard R., Contrasting Conventional and Evolutionary Approaches for the Flow Shop Scheduling Problem. EIS II.
9. Esquivel S.C., Zuppa F., Gallard R. Multiple Crossovers, Multiple Parents and the Stud for Optimization in the Flow Shop Scheduling Problem, LIDIC, SCI 2001.
10. Fonseca C. M., Fleming P. J. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization Proc. of the 5th In. Conf. on Genetic Algorithms, pp 416-423, Urbana- Champaign, IL, Morgan Kaufmann, 1993.
11. Horn.J., Multicriterion decision making, Handbook of Evolutionary Computation. F1.9:1-9:15, Oxford University Press, 1997.]
12. Pareto V., Cours d'Economie Politique, 1896, Switzerland, Lausanne: Rouge.