AN ANALYSIS ON SELECTION METHODS AND MULTIRECOMBINATION IN EVOLUTIONARY SEARCH WHEN SOLVING THE JOB SHOP SCHEDULING PROBLEM

Stark N., Salto C., Alfonso H.
Proyecto UNLPAM-09/F015[1]
Departamento de Informática - Facultad de Ingeniería
Universidad Nacional de La Pampa
Calle 110 esq. 9
(6360) General Pico – La Pampa – Rep. Argentina
e-mail: { nstark, saltoc, alfonsoh }@ing.unlpam.edu.ar
Phone: +54 2302 422780/422372, Ext. 6302

Gallard R.
Laboratorio de Investigación y Desarrollo en Inteligencia Computacional (LIDIC)[2]
Departamento de Informática
Universidad Nacional de San Luis
Ejército de los Andes 950 - Local 106
(5700) - San Luis -Argentina
e-mail: rgallard@unsl.edu.ar
Phone: +54 2652 420823
Fax : +54 2652 430224

ABSTRACT

Evolutionary algorithms (EAs) can be used as optimisation mechanisms. Based on the model of natural evolution, they work on populations of individuals instead of on single solutions. In this way, the search is performed in a parallel manner. During the last decades, there has been an increasing interest in evolutionary algorithms to solve scheduling problems. One important feature in these algorithms is the selection of individuals.

Selection is the operation by which individuals (i.e. their chromosomes) are selected for mating. To emulate natural selection, individuals with higher fitness should be selected with higher probability, and thus it is one of the operators where the fitness plays an important role. There are many different models of selection (some are not biologically plausible). Commonly, proportional, ranking, tournament selection and stochastic universal sampling are used.

EAs considered in this work are improved with a multiplicity feature to solve the job shop scheduling problems (JSSP). The algorithm applied here, multiple crossovers on multiple parents (MCMP), considers more than two parents for reproduction with the possibility to generate multiple children. This approach uses a permutation representation for the chromosome. The objective of this work is to compare the algorithms performance using different selection mechanisms and to analyse the different crossover methods developed to apply MCMP with a permutation representation.

Keywords: evolutionary algorithms, representation, selection, crossover operator, multiplicity features.

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

1. INTRODUCTION

Evolutionary algorithms (EAs) are optimisation and search procedures inspired by genetics and the process of natural selection. Evolutionary algorithms operate on a population P of potential solutions (called individuals a_i $\hat{I}$ I, where I represents the space of all possible individuals) of the optimisation problem, which simultaneously sample the search space. The applied paradigm is the survival of the fittest individual to produce better and better approximations to a solution. The quality of an individual is measured by a fitness function $f : I$ ® R. After initialisation of the population P, a loop begins which consists in selection and recombination of its individuals until some termination criterion is reached. Each run of the loop is called a generation and $P(t) = (a_1^t ..., a_m^t) \in I^m$ (where m is the population size) denotes the population at generation t. At each generation, a new set of approximations is created by the process of *selecting individuals* according to their level of fitness in the problem domain and breeding them together using operators borrowed from natural genetics. This process leads to the evolution of populations of individuals that are better suited to their environment than the individuals that they were created from, just as in natural adaptation.

The selection operator is intended to improve the average quality of the population by giving to individuals of higher quality a higher probability to contribute with its genetic material for the next generation. Selection thereby focuses the search on promising regions in the search space. Recombination changes the genetic material in the population either by crossover or by mutation in order to exploit new points in the search space.

A well-known property of a selection operator is *selective pressure*, which can be defined as the probability of the best individual being selected relative to the average probability of selection of all individuals. During the selection step of an EA, copies of the better ones replace the worst individuals. Consequently, part of the genetic material contained in these worst individuals disappears forever. This *loss of diversity* is defined as the proportion of the population that is not selected for the next generation [3]. When the selection mechanism imposes a strong selective pressure then the loss of diversity can be high and to prevent a premature convergence to a local optimum then, either a larger population size or adequate crossover and mutation operators are needed. On the other side of the coin a small selective pressure can excessively slow the convergence rate.

Improvements in evolutionary algorithms have been found by using different variants of a multiplicity feature [7, 8, 9, 10]. M*ultiple Crossovers on Multiple Parents* (MCMP) supplies a balance in exploitation and exploration because the searching space is efficiently exploited (by the multiple application of crossovers) and explored (by a greater number of samples provided by multiple parents). MCMP provides a means to exploit good features of more than two parents selected according to their fitness by repeatedly applying one of the *scanning crossover* (SX) variants [5,6]: a number n_1 of crossovers is applied on a number n_2 of selected parents. From the n_2 produced offspring, a number n_3 of them are selected, according to some criterion, to be inserted in the next generation. Such a single population evolutionary algorithm is powerful and performs well on a broad class of problems, in particular with job shop scheduling problems (JSSP).

In general, the task of scheduling is the allocation of jobs over time when limited resources are available, where a number of objectives should be optimised, and several constraints must be satisfied. A job is determined by a predefined set of operations, and the result of a scheduling algorithm is a schedule that contains the start times and allocation of resources to each operation [4]. The specification of an appropriate schedule representation is a decisive factor to get a good performance from an evolutionary algorithm. The main difficulty in scheduling problems is to determine which chromosome representation is the appropriate one to be used. One suitable representation is known as *job based representation,* which consists of a list of jobs and a schedule

is constructed according to the sequence of jobs. Here we deal with permutations, consequently adequate genetic operators should be used in order to produce feasible offspring [11].
Evolutionary algorithms enhanced with MCMP were successfully applied to the *job shop scheduling problem* (JSSP) under others representations [13,15, 16, 18, 19]. It was shown that a greater number of crossovers for a given number of parents provide better results.
This work aims at showing the effect of applying different selection mechanisms to a set of instances of different degree of complexity, for the JSSP. Moreover, the MCMP approach is included in the algorithm and modified SX methods [6] are considered in order to produce feasible offspring. Furthermore, SX methods are contrasted with *Adjacency Based Crossover* (ABC) [6], which is specially built for this kind of representation. For the two crossover methods we used the variants *Uniform* (U-XX) where all the parents have the same chance to be donors and *Occurrence-Based* (OB-XX) where the gene values are selected according to the number of occurrences. Details and examples of each of each crossover methods can also be found in [17].
In the next section, a fast review of the diverse sampling mechanisms implemented is carried out. Section 3 gives a description for each method and a detail of the experiments. Results under different combinations are explained in section 4. Finally, section 5 reports the conclusions of this work.

2. SAMPLING MECHANISMS

2.1. Proportional Selection

In proportional selection, an individual a_i is chosen at random for mating from a population of size m according to the following probability:

$$P_{sel}(a_i) = \frac{f(a_i)}{\sum_{j=1}^{m} f(a_j)}$$

This is the simplest selection scheme also known as *roulette-wheel selection* or *stochastic sampling with replacement.* Here, individuals are mapped to contiguous segments in the real interval [0,1] in such a way that a segment corresponding to an individual has a size equal to the individual fitness. Then a random number in such interval is generated and the individual whose segment encompasses the random number is selected.

2.2. Rank-based selection

In *linear ranking,* the selective pressure can be controlled more directly than using proportional selection and consequently the search process can be accelerated remarkably. Whitley [21] pointed out that ranking acts as a function transformation assigning a new fitness value to an individual based on its performance relative to other individuals. The Baker's original linear ranking method assigns a selection probability that is proportional to the individual's rank. Here, according to Bäck [1] the mapping $rank: I \to \{1,\ldots,m\}$ is given by:

$$\forall i \in \{1,\ldots,m\}: rank(a_i) = i \quad \Leftrightarrow \quad \forall j \in \{1,\ldots,m-1\}: f(a_j) \trianglelefteq f(a_{j+1})$$

where $\trianglelefteq$ denotes the $\leq$ relation or the $\geq$ relation for minimization or maximization problems, respectively. Consequently the index i of an individual a_i denotes its rank. Hence, individuals are sorted according to their fitness resulting a_1 the best individual and a_m the worst one. Assuming that the expected value for the number of offspring to be allocated to the best individual is $\eta_{max} = mP(a_1)$ and that to be allocated to the worst one is $\eta_{min} = mP(a_m)$ then

$$P_{sel}(a_i) = \frac{1}{m}\left(\eta_{max} - (\eta_{max} - \eta_{min}) \cdot \frac{i-1}{m-1}\right)$$

As the following constraints must hold

$$P_{sel}(a_i) \geq 0 \quad \forall i \qquad \sum_{i=1}^{m} P_{sel}(a_i) = 1$$

it is required that: $1 \leq \eta_{max} \leq 2$ and $\eta_{min} = 2 - \eta_{max}$.

The selective pressure can be adjusted by varying η_{max}. As remarked by Baker [2] if $\eta_{max} = 2.0$ then all individuals would be within 10% of the mean and the population is driven to convergence during every generation. To restrain selective pressure, Baker recommended a value of η_{max} =1.1. This value for η_{max} close to 1 leads to $P_{sel}(a_i) \approx 1/m$, almost the case of random selection.

2.3. Tournament Selection

In tournament selection q individuals are randomly chosen from the population and then the best-fitted individual, designated as the winner, is selected for the matting pool. The parameter q is known as the tournament size and usually it is fixed to $q = 2$ (binary tournament). If $q = 1$ then there is no selection at all: each individual has the same probability to be selected. As long as q increases the selective pressure is augmented.

As Blickle [3] affirms, tournament selection can be implemented efficiently having the time complexity $O(m)$ because no sorting of the population is necessary but, as a counterpart, this also leads to a high variance in the expected number of offspring resultant from m independent trials.

As showed by Bäck [1], the selection probability for individual a_i , $(i \in \{1,\ldots,m\})$ for *q-tournament* selection is given by:

$$P_{sel}(a_i) = \frac{1}{m^q}\left((m-i+1)^q - (m-i)^q\right)$$

2.4. Stochastic Universal Sampling (SUS)

The idea, introduced by Baker [2], is to make a single draw from a uniform distribution and use it for determining the exact number of copies from each parent. In this method the individuals are mapped to contiguous segments of a line, such that each individual's segment is equal in size to its fitness exactly as in proportional selection. But here equally spaced pointers are placed over the line as many as individuals have to be selected. If n is the numbers of the individuals to be selected, then the distance between the pointers is $1/n$ and the position of the first pointer is given by a randomly generated number in the range $[0, 1/n]$.

3. EXPERIMENT DESCRIPTIONS

The experiments were developed for a set of selected instances of the JSSP, under permutation representation. EA runs were made for each combination of crossover methods and selection mechanisms. Moreover for each of these combinations, trials were obtained varying n_1 (number of crossovers) and n_2 (number of parents). Finally, 50 EA runs were made for each experiment, and the results were averaged.

To distinguish EAs by selection mechanisms, six different EA´s resulted:

- **PS-EA**: this EA formed the mating pool using proportional selection.
- **RS-EA**: EA working with rank based selection.
- **TS-EA**: EA with tournament selection.
- **TpS-EA**: in this algorithm the tournament selection was modified in the way of selecting the individuals to compete: the q individuals were chosen by the use of proportional selection.
- **SUS-EA**: EA plus SUS.
- **RndS-EA**: this EA chooses randomly the selection mechanism generation by generation. All previous methods were considered.

Elitism to retain the best-valued individual was implemented. The population size was fixed at 100 individuals. USX, OBSX, OBABC and UABC were implemented and for insertion in the next generation the best child was chosen ($n_3 = 1$). Number of crossovers and parents were set to: $1 \leq n_1 \leq 4$ and $3 \leq n_2 \leq 5$, respectively. For mutation, an *interchange* operator was used. The algorithms evolved for a minimum of 500 generations, after that a control on the progress of the population mean fitness began: if this value remained within a determined range for 20 consecutive generations the algorithm stops. Probabilities for crossover and mutation were fixed at 0.7 and 0.2, respectively. These values were determined as the best combination of probabilities after many initial trials. Proportional selection and SUS were applied in the conventional way. In the case of raking selection, η_{max} was set to 1.1. In tournament selection, the size q was fixed to 2. Four instances [12], with known optimal makespan were used (table 1).

Table 1: Selected instances

Instance	Size	Optimum
la01	10 × 5	666
la06	15 × 5	926
la12	20 × 5	1039
la15	20 × 5	1207

The following relevant performance variables were chosen:

- ***Ebest*** = (Abs(*opt_val* - best value)/*opt_val*)100
  It is the percentile error of the best found individual when compared with the known, or estimated, optimum value *opt_val*. It gives us a measure of how far the best individual is from that *opt_val*.
- ***Epop*** = (Abs(*opt_val*- pop mean fitness)/*opt_val*)100
  It is the percentile error of the population mean fitness when compared with *opt_val*. It tell us how far the mean fitness is from that *opt_val*.
- ***Avg_ebest/Avg_epop*** : average of ebest/epop values for a particular approach.

4. RESULTS

In what follows, results on ***la06*** and ***la12*** instances will be discussed as demonstrative ones because similar conclusions can be derived from the other instances selected. In the next tables boldfaced values correspond to the best-found values for each performance variable considered.

Values in table 2 summarise average ebest values for *la06* instance, which were obtained as an average of all ebest values reached for each (n_1, n_2) combination. In this table boldfaced values correspond to the best-found value for each crossover method. When U-Scan or U-ABC are applied, both PS-EA and SUS-EA get a zero ebest value showing that the optimum is reached in all runs. This also happens in RndS-EA under U-Scan. Analysing OB-Scan, PS-EA has the minimum average ebest followed by SUS-EA. An opposite situation is observed in OB-ABC.

Results in table 2 also show that, both U-Scan and U-ABC obtain better ebest values than OB-Scan and OB-ABC respectively, independently of the selection methods. Most EAs employing U-Scan obtained better results than those using U-ABC.

Table 2: average ebest under EA approaches for *la06* instance.

	PS-EA	RS-EA	TpS-EA	TS-EA	SUS-EA	RndS-EA
U-Scan	**0.00000**	0.60295	0.00900	0.02700	**0.00000**	**0.00000**
OB-Scan	**0.31497**	0.50396	0.57595	0.53096	0.43197	0.46796
U-ABC	**0.00000**	0.44096	0.17099	0.08099	**0.00000**	0.14399
OB-ABC	0.44096	0.58495	0.77394	0.68395	**0.39597**	0.80994

Table 3: average ebest under EA approaches for *la12* instance.

	PS-EA	RS-EA	TpS-EA	TS-EA	SUS-EA	RndS-EA
U-Scan	1.17902	3.20821	2.30189	1.78858	**1.06673**	2.21367
OB-Scan	3.10395	3.39269	3.05582	3.48893	**2.84729**	3.44883
U-ABC	**1.53994**	3.00770	2.22169	2.35002	1.65223	2.39012
OB-ABC	3.10395	3.30446	3.20019	3.37664	**3.03978**	3.44883

Table 3 summarises average ebest values for *la12* instance. SUS-EA presents the lowest average ebest for most crossover methods, with the exception of U-ABC where PS-EA reaches an inferior value. According to these average ebest values, the best performer is SUS-EA using U-SCAN (with an error value of 1.07%) followed by PS-EA using the same crossover (1.18%). Moreover, U-SCAN crossover provides better results than the other approaches (four of the six combinations of crossovers and selection mechanisms), independently of the selection adopted. U-ABC follows it with two of the six combinations. U-XX crossover methods have reached better ebest values than OB-XX crossovers, as in the previous instance.

In what follows a deeper analysis of ebest behaviour as a function of (n_1,n_2) combinations is considered. For this study, and guided by the results observed in previous tables, we concentrate in U-Scan crossover, SUS and PS selection.

Table 4 shows ebest values for *la12* instance. In most cases, SUS-EA has lower errors than PS-EA. But the deviation of best values respect to the average for PS-EA is generally less than the corresponding to SUS-EA. With regard to the mean ebest, both algorithms exhibited rather similar values. Now analysing these results according to the (n_1, n_2) combinations (number of crossovers, number of parents), when n_1 is fixed to 2 or 3 lowest errors are reached for both algorithms independently of n_2. When only one crossover is carried out, ebest values are quite big compared with others n_1 values.

Table 5 indicates the population errors for *la12* instance. SUS-EA presents higher epop values than PS-EA in the majority of the cases. When n_1 changes from 2 to 3, a strong difference is observed in the corresponding epop values for both algorithms. Moreover, if the number of crossover applied is 3 or 4, epop and ebest values are similar, indicating that most individuals belonging to the final population are quite similar, in quality, to the best individual found so far.

Figures 1 and 2 show how individuals are spread in final populations when the number of parents is fixed, and the number of crossovers changes (similar picture is presented in remaining final populations). When the number of crossovers is increased, populations including individual

Table 4: ebest values for *la12* instance.

n_2	n_1	PS-EA			SUS-EA		
		ebest	Avg_ebest	dev	Ebest	avg_ebest	dev
3	1	1.73244	3.30703	7.11641	1.63619	3.55727	10.12566
	2	0.76997	2.47546	7.39012	1.44370	2.67565	8.37635
	3	1.15496	2.56015	7.84805	0.67372	2.55630	9.91424
	4	1.73244	2.66410	6.73838	0.67372	2.67180	10.53964
4	1	1.34745	3.43985	10.48188	1.25120	3.31665	10.44951
	2	0.76997	2.42348	9.62923	0.38499	2.66987	7.99952
	3	0.76997	2.70452	9.10315	0.48123	2.76997	10.90458
	4	0.96246	3.02406	9.81895	0.67372	2.81617	10.08497
5	1	1.82868	3.73821	9.95574	2.02117	3.63619	11.04405
	2	0.86622	3.11068	11.20211	0.67372	2.76997	10.56233
	3	0.76997	3.02984	10.34159	1.25120	2.91242	9.41516
	4	1.44370	3.31088	9.79171	1.63619	3.40712	10.19604

Table 5: epop values for *la12* instance.

n_2	n_1	PS-EA			SUS-EA		
		epop	avg_epop	dev	epop	avg_epop	dev
3	1	16.82419	21.27944	32.28795	17.06895	21.84160	29.71571
	2	5.39742	15.86333	32.32524	12.15493	16.65155	22.69353
	3	1.34191	11.05726	59.49887	1.53106	10.76689	53.50390
	4	1.73244	6.43885	45.83055	1.15496	8.02566	52.66961
4	1	13.08437	20.21564	32.34739	15.00076	19.36752	27.80081
	2	9.55332	14.29027	21.08610	10.58736	14.77719	25.46321
	3	0.76997	6.82334	57.21803	1.34745	8.91348	53.79106
	4	0.96246	3.75289	25.79113	0.82803	5.01895	43.71177
5	1	12.29599	18.65617	27.60525	13.44596	18.90383	27.23307
	2	4.79249	14.15772	31.74946	11.08114	14.44205	20.87464
	3	1.16718	6.96309	51.07711	1.42729	8.08333	55.75630
	4	1.44370	3.71299	21.94697	1.63619	3.73865	13.53900

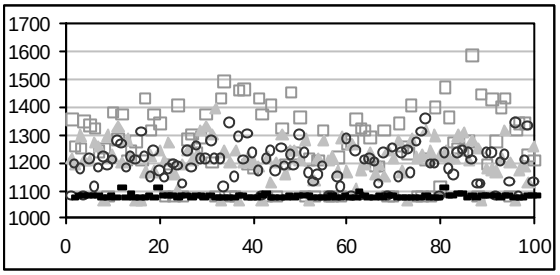
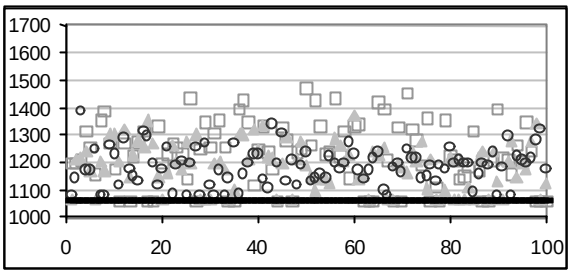


Figure 1: Final populations for PS-EA for the same n_1 and different n_2.



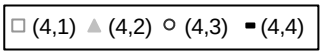


Figure 2: Final populations for SUS-EA for the same n_1 and different n_2

with more similar makespan are observed, independently of the selection method adopted. When n_1 is equal to 4, the makespan of individuals remains between 1050 and 1100 for PS-EA, but for SUS-EA a more concentration around a 1050 value is observed. This fact is confirmed by epop values in the (4,4) combination (table 5). The gene conformation shows that these individuals represent the same schedules. In both algorithms, the greatest dispersion is presented when n_1 is set to one.When two crossovers are applied, the greatest concentration of makespan values are between 1150 and 1300, but also individuals with makespan near to 1050 are observed in the majority of the populations. A similar situation happens when n_1 is set to 3.

Figures 3 to 6 outline the relationship between the population mean fitness (fluctuating curves) and the best individual fitness (bottom curves) through the evolution process for *la12* instance. The horizontal axis plots the generation number while the vertical axis plots the makespan values. Figures 3 and 4 show this relation for a fixed number of parents and variable number of crossovers. Although the curve corresponding to population mean fitness in (4,4) combination presents the same fluctuation as others through generation changes, it exhibits a bigger similarity to the curve corresponding to the best individual found, in both algorithms. Contrasting values from table 4 and 5 for the same combination, this strong similarity can also be perceived. Curves representing (2,4) and (3,4) population mean fitness for PS-EA (figure 3) fluctuate between similar values during the first half of the evolution, but in the last part (3,4) combination is below others. For the same algorithm, (1,4) combination presents the biggest distance between best and average curves. Examining SUS-EA (figure 4), the (1,4) curve representing population mean fitness is above to the (2,4) one until half of the evolution, but the inverse situation is observed in the rest of the evolution. For the (3,4) combination, the population mean fitness curve remains close to that representing the makespan of the best individual.

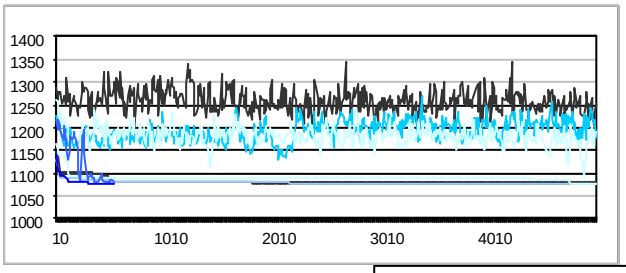
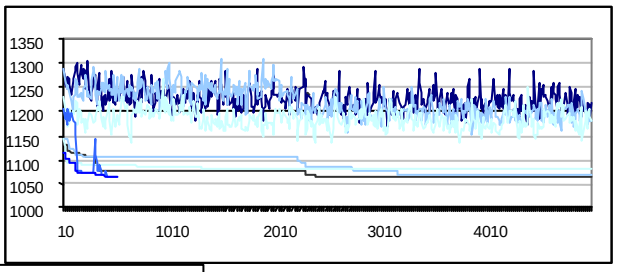


Figure 3: Evolution of the best and average values for the same n_2 = 4 and different n_1 for PS-EA.



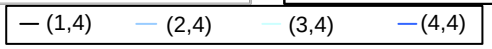


Figure 4 Evolution of the best and average values for the same n_2 = 4 and different n_1 for SUS-EA.

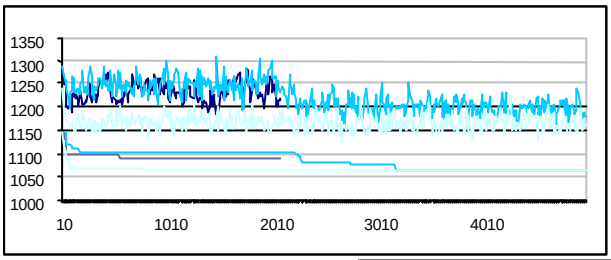
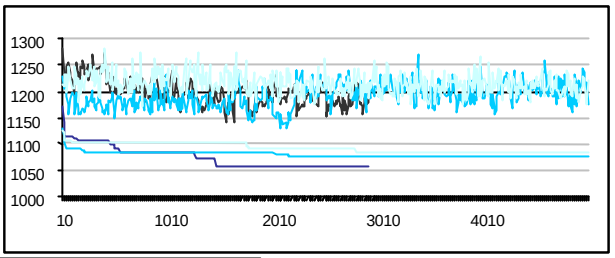


Figure 5: Evolution of the best and average values for the same $n_1 = 2$ and different n_2 for PS-EA.



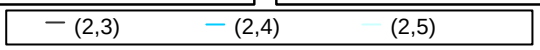


Figure 6 Evolution of the best and average values for the same $n_1 = 2$ and different n_2 for SUS-EA.

An earlier stop of the evolution (near generation 500) can be observed in the (4,4) combination for both algorithms. This means that the population mean fitness value remained within a determined range for 20 consecutive generations, and no further improvement was found.

Figures 5 and 6 introduce the relationship between the number n_1 of crossover and n_2 of parents by fixing n_1 and changing n_2 for the *la12* instance, we opt for n_1=2 regarding best *ebest* value in Table 4. In both algorithms it can be observed a bigger difference in the fluctuation limits of population mean fitness values for the distinct (n_1,n_2) combinations. For both algorithms, the (2,3) association reaches a stagnation of the mean and best makespan in early generations.

Now a genotype analysis follows. Figures 7 and 8 were obtained from the chromosomal conformation of all best individuals found in each run. From there, we want to analyse how the jobs, in function to their length, are positioned in the chromosome for optimum or near optimum solutions to a particular instance, trying to find a pattern of allocation of jobs in the chromosome. Moreover, this gives us an idea about the genotype diversity of the best individuals found. Jobs in these figures are sorted according to their total processing time, which is calculated as the sum of the duration of all operations for a particular job. The X-axis represents chromosome positions, the Y-axis shows the number of occurrences of a job in a given position and the Z-axis represents the different jobs.

The genotype distributions of best individuals, found under both PS-EA and SUS-EA (instance *la06*, figure 7), have a strong similarity. The shortest job (job 2) appears frequently at the end of the chromosome. Short jobs are distributed at the first half positions while average length jobs are generally located in the second half positions. Long jobs appear with more frequency at the beginning of the chromosome.

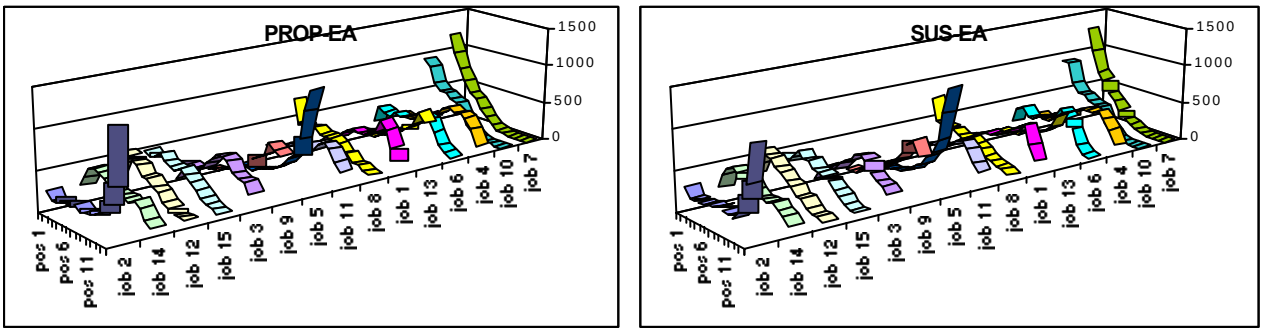


Figure 7: Job distribution for *la06* instance

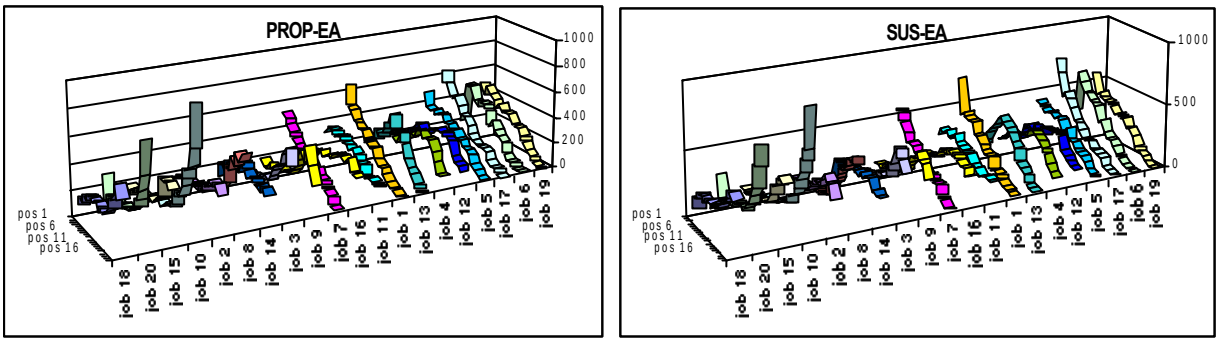


Figure 8: Job distribution for *la12* instance

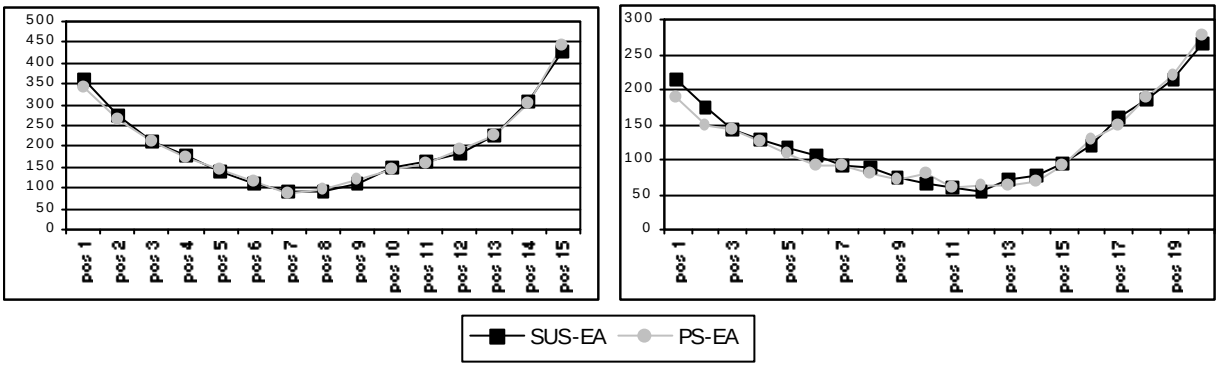


Figure 9: Variety of jobs in each position of the chromosome for *la06* and *la12* instance

For the *la12* instance (figure 8), again a high similitude is observed in the genotype distributions of best individuals under both selection methods. Shortest jobs are more frequently placed in the final positions; while longest jobs are located at the beginning of the chromosome. Each average length jobs present the same particular pattern of distribution under both selection methods.

Figure 9 shows the standard deviation observed for each chromosome position regarding the frequency of jobs positioned in each one. The horizontal axis represents the different positions and the vertical axis, the deviation value. SUS-EA curves are above the corresponding PS-EA curves in both instances, showing a higher variation. But all these curves are following the same pattern: higher variation at the ends of the chromosome and lower variation in the central positions. This indicates that job distributions are independent of the selection method adopted: this is the task of the evolutionary algorithm.

5. CONCLUSIONS

This contribution had presented some aspects of the behaviour of an evolutionary algorithm used to solve the JSSP for a set of instances of diverse complexity. The EA joined a multirecombinative approach (MCMP) with different selection methods. Also, an analysis to determine the most suitable crossover method to be applied to the adopted permutation representation was carried out.

For both crossover methods studied (scanning or adjacency based crossover), there is an indication that for any (n_1,n_2) association, uniform approaches had better performance than those based on occurrence. Particularly, contrasting the results obtained for the uniform approaches, USX shows a better quality of results than UABC.

Regarding (n_1,n_2) combinations, best results are obtained when two or three crossovers are applied, independently of the number of parents. Worst results correspond to the implementation of a single crossover. Finally, when four crossovers are applied the results obtained are better than when a

single crossover is applied; in most cases the average makespan of the final population is equal or rather similar to the makespan of the best individual. As soon as the number of crossover is increased, the individuals belonging to the final population tends to be more similar from the point of view of both makespan value and gene conformation. Then a mechanism to incorporate diversity to the evolution might be considered.

Contrasting the different selection methods, the values achieved showed a better performance for both SUS and Proportional Selection over other methods, without a clear conclusion on which one outperforms the other.

The promising results on these smallest instances encourage us to deep investigation. To improve results in larger instances, further work will include diverse multirecombination schemes to add diversity without losing the algorithm search ability, and the study of the adjustment of selection method according to the evolution. Another important aspect to study is how the jobs are positioned in the chromosome through the evolutionary process; this can complement the formae-based [14, 20] evolutionary search methods.

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 Pampa, and the ANPCYT from which we receive continuous support.

7. REFERENCES

[1] Bäck T: *Evolutionary algorithms in theory and practice*. Oxford University Press, 1996.

[2] Baker J. E.: *Reducing bias and inefficiency in the selection algorithm*. Proc. of the 2nd International Conf. on Genetic Algorithms, pp. 14-21. J.J Grefenstette Ed, 1987.

[3] Blickle T., Thiele L.: *A comparison of selection schemes used in genetic algorithms.* Technical Report 11 TIK, Swiss Federal Institute of Technology, December 1995.

[4] Bruns, R. Scheduling, in Th. Bäck, D. B. Fogel, and Z. Michalewicz, (editors), *Handbook of Evolutionary Computation* , chapter F1.5, pages F1.5:1-F1.5:9. Oxford University Press, New York, and Institute of Physics Publishing, Bristol, 1997.

[5] Eiben A.E., Van Kemenade CHM, Kok J.N.: *Orgy in the computer: multi-parent reproduction in genetic algorithms,* in Moran F., Moreno A., Merlo J.J., and Chacon P., editors, Proc. of the 3rd European Conference on Artificial Life, number 929 in LNAI, pp. 934-945. Springer Verlag, 1995.

[6] Eiben A.E., Bäck T.: *An empirical investigation of multi-parent recombination operators in evolution strategies,* Evolutionary Computation, 5(3), pp.347-365, 1997.

[7] Esquivel S., Leiva A., Gallard R.: *Multiple crossover per couple in genetic algorithms.* Proc. of the 4th IEEE International Conf. on Evolutionary Computation (ICEC'97), pp 103-106, Indianapolis, USA, April 1997.

[8] Esquivel S., Leiva A., Gallard R.: *Couple Fitness Based Selection with Multiple Crossover Per Couple in Genetic Algorithms.* Proc. of the International Symposium on Engineering of Intelligent Systems, University of La Laguna, España, Tenerife, Vol. 1, pp. 235-241, ISBN 3-906454-12-6, Feb. 1998.

[9] Esquivel S., Leiva A., Gallard R.: *Multiple Crossovers between Multiple Parents to Improve Search in Evolutionary Algorithms,* en Proc. of the 1999 Congress on Evolutionary Computation (IEEE), vol. 2, pp. 1589-1594, Washington DC., 1999.

[10] Esquivel S., Leiva A., Gallard R.: *Multiplicity in Genetic Algorithms to Face Multicriteria Optimization*, Proc. of the 1999 Congress on Evolutionary Computation, IEEE Service Center, pp. 85-90, Washington, D.C., 1999.

[11] Holsapple, C., Jacob, V., Pakath, R., y Zaveri, J., A Genetics-Based Hybrid Scheduler for Generating Static Schedules in Flexible Manufacturing Contexts, en *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, pag. 953-971, 1993.

[12] Lawrence S.: *Resource Constrained Project Scheduling: an Experimental Investigation of Heuristic Scheduling Techniques*, (Supplement), Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, Pennsylvania, 1984.

[13] Minetti G., Salto C., Alfonso H., Gallard R.: *A Comparison of two Multirecombinated Evolutionary Algorithms for the Job Shop Scheduling Problem*, Proc. VI Congreso Argentino de Ciencias de la Computación, CACIC 2000 Universidad Nacional de la Patagonia San Juan Bosco, 2000.

[14] Radcliffe, N. J., *Non-Linear Genetic Representations*, in Parallel Problem Solving from Nature 2, (Ed: R. Manner and B. Manderick, Elsevier Science Publishers), 1992.

[15] Salto C., Alfonso H., Gallard R.: *Multiplicity and Incest Prevention in Evolutionary Algorithms to Deal with de Job Shop Problem*, Proc. of the Second ICSC Symposium on Engineering of Intelligent Systems, University of Paisley, Scotland, pp. 451-457, 2000.

[16] Salto C., Minetti G., Alfonso H., Gallard R.: *A Hybrid Evolutionary Algorithm: Multirecombination with Priority Rule Base Representation for the Job Shop Scheduling Problem*, Proc. VI Congreso Argentino de Ciencias de la Computación, CACIC 2000, Universidad Nacional de la Patagonia San Juan Bosco, pp. 1365-1376, 2000.

[17] Salto C., Stack N.., Alfonso H., Gallard R.: *Contrasting Two Mcmp Alternatives In Evolutionary Algorithms To Solve The Job Shop Scheduling Problem*, Proc. V Congreso Argentino de Ciencias de la Computación, CACIC 2001, Universidad Nacional de la Patagonia Austral, pp. 1093-1104, 2001.

[18] Salto C., Alfonso H., Gallard R.: *Breeding in multirecombinated evolutionary algorithms with permutation based representation for the job shop scheduling problem*, Proc. of 4th International ICSC Symposium on soft Computing and Intelligent Systems for Industry (SOCO/ISFI2001), pp. 130, 2001.

[19] Salto C., Alfonso H., Gallard R.: *Performance Evaluation of Evolutionary Algorithms under Different Representations when Solving the JSS problem*, Proc. of 5th World Multi-Conference on Systemics, Cybernetics and Informatics (SCI 2001), pp. 424-428, 2001.

[20] Surry, P.D., Radcliffe, N.J., *Formal Algorithms + Formal Representations = Search Strategies*, Parallel Problem Solving From Nature IV, 1996.

[21] Whitley D.: *The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best*. Proc. of the 3rd International Conf. on Genetic Algorithms, pp. 116-121. Morgan Kaufmann 1989.