INCORPORATING TABU SEARCH FOR LOCAL SEARCH INTO EVOLUTIONARY ALGORITHMS TO SOLVE THE JOB SHOP SCHEDULING PROBLEM

Fernández Natalia, Salto Carolina, Alfonso Hugo
Laboratorio de Investigación en Sistemas Inteligentes (LISI)[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: { fernaty, saltoc, alfonsoh }@ing.unlpam.edu.ar
Phone/Fax: +54 (2302) 422780/422372, Ext. 6302

Gallard Raúl
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

A new issue for combinatorial optimization problems is to incorporate local search into the framework of evolutionary algorithms, leading to hybrid evolutionary algorithms. With the hybrid approach, evolutionary algorithms are used to perform global exploration among population while other heuristic methods are used to perform local exploitation around chromosomes. Due to the complementary properties of evolutionary algorithms and conventional heuristics, the hybrid approach often outperforms either method operating alone. When designing a hybrid evolutionary algorithm (HEA), a fundamental principle is to hybridize where possible.

This paper aims at developing powerful HEA to find high quality sub-optimal solutions for the job shop scheduling problem through tabu search (TS), an advanced local search meta-heuristic.

Experiments of such a hybrid algorithm are carried out on different benchmark. Analysis of the behaviour of the algorithm sheds light on ways to further improvement and are discussed here.

Key Words: evolutionary algorithms, scheduling, hybridization, local search, optimization.

[1] The Research Group is supported by the Universidad Nacional de La Pampa.

[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 (EAs) offer a robust approach to problem solving. EAs are extremely flexible and can be extended by incorporating alternative approaches to favour the search process. One way is to hybridize an evolutionary algorithm with standard local search procedures. Local search employs the idea that a given solution may be improved by making small changes [17].

Some notions need to be introduced concerning local search. Two functions are defined on a set of *feasible solutions* (F). The *cost function* is a mapping $c : F \to R$, which in most cases is closely related to the function that is to be optimized. The *neighbourhood function* is a mapping $n : F \to 2^F$, which defines for each solution $x \in F$ *a neighbourhood* $n(x) \subseteq F$. Each solution in $n(x)$ is called a neighbour of x. The execution of a local search algorithm defines a walk in F such that each solution visited is a neighbour of the previously visited one.

The basic algorithm is called *iterative improvement*. Starting at some initial feasible solution, its neighbourhood is searched for a solution of lower cost. If such a solution is found, the algorithm is continued from there; otherwise, a local minimum has been found.

The quality of the local minimum depends on the initial solution, on the neighbourhood function, and on the method of searching the neighbourhoods. An initial solution may be obtained by generating it randomly or by applying a heuristic rule. The choice of a good neighbourhood is often difficult. There is a clear trade-off between small and large neighbourhoods: if the number of neighbours is larger, the probability of finding a good neighbour may be higher, but looking for it takes more time. There are several alternatives for searching the neighbourhood: one may take the first neighbour found of lower cost (*first improvement*), or take the best neighbour in the entire neighbourhood (*best improvement*), or take the best of a sample of neighbours, provided it is an improving one.

In our case, the evolutionary algorithms is combined with an advanced local search meta-heuristic: tabu search algorithm [7], leading to a class of HEAs.

In this paper, we are interested in tackling with HEAs a hard combinatorial optimization problem: the job shop scheduling problem (JSSP). The JSSP consists of a set of jobs to be allocated to a set of machines [3, 10]. Each job consists of a chain of operations, each of which needs to be processed during an uninterrupted time period of a given length on a given machine. Each machine can process at most one operation at a time. A schedule is an allocation of the operations to time intervals on the machines. The problem is to find a schedule of minimum length. Unfortunately, the problem is very difficult to solve because it belongs to the NP-complete family [2].

In previous works [12, 13, 9, 16] diverse representations of solutions for the JSSP such as operation based representation, job based representation, decoders and priority rule based representation were investigated. Operation based representation (OBR) was the one with best results, although for those more complex instances suboptimal solutions were far from the known optimum. OBR encodes a schedule as a sequence of operations, and each gene stands for one operation. Gen, Tsujimura and Kubota proposed to identify all operations for a job with the same symbol and then interpret them according to the order of occurrence in the sequence for a given chromosome [4, 5]. For an n-job m-machine problem a chromosome contains $n \times m$ genes. Each job appears in the chromosome exactly m times, and each repeated gene does not indicate a concrete operation of a job but refers to an operation which is context-dependent. It is easy to see that any permutation of the chromosome always yields to a feasible schedule. In order that chromosomes represent feasible schedules a specially designed crossover operator called *partial schedule exchange* [3] was used. For mutation a simple *exchange mutation* operator was adopted.

Consequently, in the present work the hybrid EA is contrasted against a conventional EA both using OBR on the set of those instances, which resulted more difficult in previous works.

The paper is organized as follows. We first review the TS algorithm (section 2). Then, we present the different alternatives of hybrization proposed in this work (section 3). Section 4 presents computational results and section 5 analyses the behaviour of the algorithms. The last section gives conclusions.

2. TABU SEARCH

The main idea behind tabu search is very simple [8]. The procedure moves from one schedule to another, with the next schedule being possibly worse than the preceding schedule. A memory forces the search to explore new areas of the search space. Some solutions (schedules) that have been examined recently can be memorized and these become tabu (forbidden) points to be avoided in making decisions about selecting the next solution. Tabu search is basically deterministic, but it is possible to add some probabilistic elements to it [7].

Supposing a JSSP with 3 jobs and 3 machines, a possible chromosome under operation based representation might be $x = [2, 1, 2, 3, 1, 2, 3, 3, 1]$. Now some evaluation function that provides feedback for the search is needed. Let us assume that the previous sequence of operations give a makespan of 30. For each solution or schedule a neighbourhood is defined. In this work, it consists in fixing one position (randomly selected) and interchanging the gene in that position with (λ-1) non-identical genes at randomly selected positions. In these experiments, λ was set to the amount of jobs for a determined instance. In this particular example λ is equal to 3, thus the neighbourhood for a particular solution is formed by two solutions. For example, if the first selected position is 7 (the gene corresponds to the second operation of job 3) and the other two are 5 (second operation of job 1) and 9 (third operation of job 1); after the interchange of values the neighbours obtained are [2, 1, 2, 3, 3, 2, 1, 3, 1] and [2, 1, 2, 3, 1, 2, 1, 3, 3]. These new sequences need to be examined, and the best one is selected. Suppose that interchanging the second operation of job 3 and third operation of job 1 generates the best evaluation, so this new sequence of operations [2, 1, 2, 3, 1, 2, 1, 3, 3] yields the current best solution. In order to keep a record of the actions (moves), some memory structures will be needed for bookkeeping. Due to the representation of solutions adopted in this particular implementation and the neighbourhood configuration adopted, the operations of jobs that were interchanged are remembered, as well as the time when this interchange was made, so that a difference between older and more recent interchanges can be made. For each possible interchange of operations of different jobs a time stamp is needed: the value of the time stamp will provide information on the recency of the interchange of these particular operations. Moreover, the positions inside the chromosome occupied by the interchanged operations are registered, trying to avoid returning to a local minimum that has been visited before. Thus, a matrix M will serve as a memory so-called *recency based memory*. This matrix is initialized to 0. Assuming that any piece of information can stay in a memory for at most, say five iterations (horizon), an interpretation of an entry:

$$M(i_n, j_m) = (k, pos_{source}, pos_{dest}) \; (k \neq 0)$$

might be: "the n-th operation of i job at pos_{source} position inside the chromosome and the m-th operation of j job at pos_{dest} position inside the chromosome were interchanged 5-k iterations ago". Under this interpretation, the contents of the memory structure M after one iteration (after the registration of the new current solution) in our example is given in figure 1.

The values in each entry of the memory are interpreted as the number of iterations for which given operations and their respective positions inside the chromosome are not available for any interchange. This interpretation requires that all nonzero entries of the memory be decreased by one unit at every iteration. This decrement facilitates the process of forgetting after five iterations and reflects the fact that all of the recorded interchanges took place one generation earlier.

2	**2**	**2**	**3**	**3**	**3**	
0	0	0	0	0	0	**1**
	0	0	0	0	0	**1**
		0	0	(5, 9, 7)	0	**1**
			0	0	0	**2**
				0	0	**2**
					0	**2**

Figure 1: the contents of the recency-based memory *M* for the JSSP after one iteration.

2	**2**	**2**	**3**	**3**	**3**	
0	0	0	(5, 3, 6)	0	0	**1**
	0	(2, 2, 5)	0	0	(4, 1, 7)	**1**
		0	0	(1, 9, 7)	0	**1**
			0	0	0	**2**
				(3, 9, 4)	0	**2**
					0	**2**

Figure 2: the contents of the recency-based memory *M* for the JSSP. The horizon is five iterations.

After evaluating each neighbour, their respective merits are known, but tabu search utilises the memory to force the search to explore new areas of the search space. The memorised interchanges that have been made recently are tabu (forbidden) for selecting the next solution, so they are not considered.

After four additional iterations, selecting the best neighbour (which is not necessarily better than the current point) and making appropriate interchanges, the memory has the contents shown in figure 2.

Interchanging the first operation of job 3 with the first operation of job 1 is *tabu* for the next five iterations (if the positions currently occupied by those operations are equal to the ones in the corresponding cell on *M* matrix). Similarly, interchanges of the third operation of job 2 and the second operation of job 1, the third operation of job 3 with second operation of job 1, second operation of job 3 with third operation of job 1, first operation of job 3 with first operation of job 1, and second operation of job 3 with second operation of job 2, are also in the tabu list. Among them, the interchange between the second operation of job 3 and third operation of job 1 is the oldest (i.e., it happened five iterations ago) and this interchange will be removed from the tabu list after the next iteration.

At any stage, there is a current solution being processed which implies a neighbourhood, and from its neighbourhood, tabu solutions are eliminated from possible exploration.

It might happen that one of the tabu neighbours provides an excellent evaluation score better than the score of any solution considered previously. In order to make the search more flexible, tabu search considers solutions from the whole neighbourhood, evaluates them all, and under normal circumstances selects a non-tabu solution with a better evaluation score than the current solution. But in exceptional circumstances, such a superior tabu solution is taken as the next point. This override of the tabu classification occurs when a so-called *aspiration criterion* is met.

There are also other possibilities for increasing the flexibility of the search. Other memory, so-called *frequency-based* memory, which operates over a much longer horizon can be introduced. For the JSS problem, a matrix *H* may serve as a long term-memory. This matrix is initialized to 0 and at any stage of the search the entry $H(i_n, j_m) = p$ is interpreted as "during the last h iterations, the n^{th} operation of job i and the m^{th} operation of job j was interchanged p times". Usually, the value of the horizon h is quite large, at least in comparison with the horizon of recency-based memory.

These frequency counts show the distribution of moves throughout the last h iterations, so this type of memory might be useful to *diversify* the search. For example, the frequency-based memory provides information concerning which interchanges have been under-represented or not represented at all, and we can diversity the search by exploring these possibilities.

The use of long-term memory in tabu search is usually restricted to some special circumstances. For example, we might encounter a situation where all non-tabu moves lead to a worse solution. Thus, to make a meaningful decision, it might be worthwhile to refer to the contents of the long-term memory. There are many possibilities here for incorporating this information into the decision-making process. The most typical approach makes the most frequent moves less attractive. Usually the value of the evaluation score is decreased by some penalty measure that depends on the frequency, and the final score determines the winner. The evaluation formula for a new solution x' used in such circumstances is

$$eval(x') - penalty(x')$$

Where *eval* returns the value of the original evaluation function and

$$penalty(x') = c * H(i_n, j_m)$$

where c serves as a coefficient and $H(i_n, j_m)$ is the value taken from the long-term memory H.

The above option of including frequency values in a penalty measure for evaluating solutions diversifies the search. Of course, many other options might be considered in connection with tabu search. For example, if we have to select a tabu move, we might use an additional rule (so-called *aspiration by default*) to select a move that is the "oldest" of all those considered.

3. THE HYBRID APPROACH

Local improvement procedures have been incorporated into EAs in order to improve their performance. Such EAs are called hybrid-EAs (HEA). There are several ways how to incorporate local improvement procedures into an EA [11, 1]:

- Devise new genetic operators inspired from conventional heuristics.
- Incorporate conventional heuristics into evolutionary algorithms: involves hybridizing conventional heuristics into evolutionary algorithms where possible.

In this work, we have considered the last approach of how to incorporate local search, in particular tabu search, into an EA. Various alternatives for hybridization are considered [14, 15]:

- HEA-IP: TS is applied to some percentage X% of the best (or worst) individuals in the randomly generated initial population, to establish a better starting point for the evolutionary process. In this way, a hybrid evolutionary algorithm with elitism can guarantee to do no worse than the conventional EA does.
- HEA-MP: TS is applied to some percentage X% of the best (or worst) individuals in an intermediate population. In this option the evolutionary process is suspended after certain number α of generations, then local search is applied and when it finishes the evolutionary process is restarted for the subsequent α generations.
- HEA-FP: TS is applied to all individuals in the final population.
- HEA-P: As it is usual with genetic operators, here TS is applied with certain probability to each new created individual. Here TS works together with mutation and crossover operators, to perform quick and localized optimization in order to improve offspring before returning it to be evaluated.

4. EXPERIMENTS

A conventional EA and the above described hybrid variants (HEA-IP, HEA-MP, HEA-FP, HEA-P) were contrasted for a set of selected instances of the JSSP. For each instance a series of ten runs was performed. The evolutionary algorithms used proportional selection for mating and elitism to retain the best-valued individual. The population size was fixed at 50 individuals. The maximum number of generations was fixed at 500, and probabilities for crossover and mutation were fixed at 0.65 and 0.01, respectively. In HEA-IP and HEA-MP the percentage X was set to 50, 75, 90 and 100%. In HEA-MP α was fixed at 150. The probability for applying TS in HEA-P was set to 0.1, 0.2, 0.4 and 0.5. The stop criterion adopted for the tabu search procedure was setting to 100 the number of iterations. Moves are considered tabu for 5 iterations, the information stored in frecuency-based memory is erased after 50 iterations. The *c* coefficient used by the penalty evaluation was set to 0.7. All these values were determined as the best combination of values after many initial trials. Four instances [6], with known optimal makespan were used (table 1).

Instance	Size	Optimum
abz6	10x10	943
abz7	20x15	657
ft10	10x10	930
ft20	20x5	1165

Table1: Instances

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 tells us how far the mean fitness is from that *opt_val*.

5. RESULTS

In what follows we will discuss results on *abz6* and *ft20* as demonstrative instances because similar conclusions were derived from the other instances. Boldfaced values correspond to the best found ones for each performance variable considered.

Table 2 and figure 3 present average *Ebest* values corresponding to *abz6* instance for each hybrid proposed method. All hybrid methods exhibited best results than the ones obtained for a conventional evolutionary algorithm. Regarding quality in the results, HEA-P presents a considerable improvement, followed in order of merit by MEA-MP.

Table 3 and figure 4 show average *Epop* values corresponding to *abz6* instance for each hybrid proposed method. HEA-P achieves good results, reducing *Epop* values to the half of those reached by the conventional evolutionary algorithm. When considering all hybrid methods, HEA-FP presents individuals belonging to the final population closest to the best-found individual; this can be derived from the smaller epop value observed for this hybrid approach.

Analyzing table 4, HEA-P reach the optimum value at least once for all probability values (in all the cases *Ebest* values are 0%). For the same alternative, a gradual reduction in average *Ebest* values can be observed as long as the probability values used in the experiments are increased. In the case of HEA-IP, regarding both the percentages used to select individuals from the whole population and the

Alternative Methods	Average Ebest	
	Min	Mean
EA	2.96925	5.91729
HEA-IP	2.52992	4.01606
HEA-MP	0.69686	2.28753
HEA-FP	2.22694	3.10710
HEA-P	**0.00000**	**0.23860**

Table 2: Average minimum *ebest* for *abz6* instance

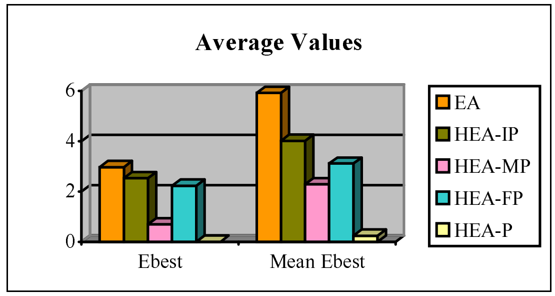

Figure 3: Average minimum *ebest* for *abz6* instance

Alternative Methods	Average Epop	
	Min	Mean
EA	20.23190	24.40701
HEA-IP	21.79335	24.24886
HEA-MP	22.04118	24.31451
HEA-FP	**7.11570**	**7.87847**
HEA-P	12.80191	14.39643

Table 3: Average minimum *epop* for *abz6* instance

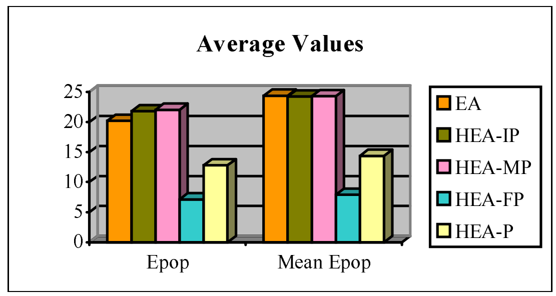

Figure 4: Average minimum *epop* for *abz6* instance

Experiment		Min. Ebest	Av. Ebest	Min. Epop	Av. Epop
EA		2.96925	5.91729	20.23190	24.40701
	100%	3.71156	4.11453	22.03054	23.95621
	Best 90%	2.75716	4.51750	21.75033	24.89052
	Best 75%	3.18134	4.04030	22.45328	25.00830
HEA-IP	Best 50%	1.48462	3.92365	21.19376	23.73925
	Worst 90%	2.22694	3.61612	22.73673	24.55313
	Worst 75%	2.01485	3.93425	21.47141	23.60944
	Worst 50%	2.33298	3.96607	20.91738	23.98520
	100%	0.21209	2.0360551	24.319506	25.839016
	Best 90%	1.16649	2.3753977	20.779657	23.479901
	Best 75%	1.16649	2.5132556	22.171128	24.319506
HEA-MP	Best 50%	0.53022	3.0010604	22.030539	23.970702
	Worst 90%	0.53022	1.5800636	20.642251	23.710381
	Worst 75%	0.53022	2.173913	21.750331	23.667101
	Worst 50%	0.74231	2.3329799	22.594842	25.214947
HEA-FP		2.22694	3.10710	**7.11570**	**7.87847**
	0.1	**0.00000**	0.55143	17.43581	19.29861
HEA-P	**0.2**	**0.00000**	0.24390	13.41662	15.12815
	0.4	**0.00000**	0.11665	10.57825	12.43060
	0.5	**0.00000**	**0.04242**	9.77696	10.72835

Table 4: *abz6* instance

characteristics of them, it is difficult to establish a pattern in the algorithm behaviour. For this alternative, the minimum ebest (1.48%) is reached when the 50% of the best indivuals from the initial population are selected to fine-tuning. The minimum *Ebest* for HEA-MP approach (0.21%) is obtained when all the individuals from intermediate population are submitted to tabu search. Moreover, a decrease of average *Ebest* values are observed as long as the percentage of selected individuals to submit to the local search procedure are increased, independently if they belong to the best or worst group.

When looking at *Epop,* HEA-FP provides both the best minimum and average *Epop* values (7.11 and 7.85%). Other approaches with values around 10 to 24%, give a higher diversity in the final population. These results have further implications. From the application point of view, a final population with individuals of similar fitness is providing a set of alternative good schedules. From the evolutionary point of view a high diversity in the population allows us to continue the search for improving solutions. Depending on other more general objectives in the designer's mind, he (she) could adopt either point of view.

Analizing *Ebest* values for *ft20* instance (table 5 and figure 5), all proposed hybrid methods outperform results obtained under conventional evolutionary algorithms. Among the different alternatives, HEA-P presents the best results for both minimum and mean *Ebest,* and it is followed by HEA-MP. Regarding minimum and average *Epop* values for *ft20* instance (table 6 and figure 6), HEA-FP displays the smallest population error (12.64 %) among the hybrid methods. Also good performance is exhibited by HEA-P.

Alternative Methods	**Average Ebest**	
	Min	**Mean**
EA	10.38627	12.38627
HEA-IP	8.17903	10.16364
HEA-MP	4.73329	6.51625
HEA-FP	5.23605	7.26180
HEA-P	**1.75966**	**2.66524**

Table 5: Average minimum Ebest for *ft20* instance

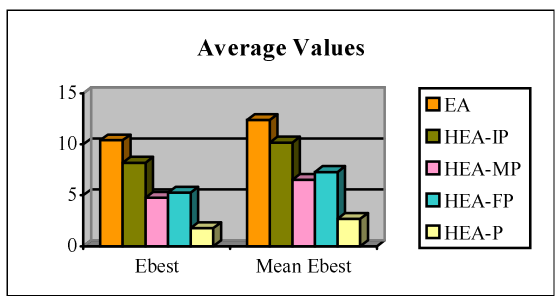

Figure 5: Average minimum Ebest for *ft20* instance

Alternative Methods	**Average Epop**	
	Min	**Mean**
EA	28.88425	31.55082
HEA-IP	28.97398	30.93217
HEA-MP	27.93657	30.29113
HEA-FP	**12.64686**	**13.70633**
HEA-P	16.59247	19.32689

Table 6: Average minimum Epop for *ft20* instance

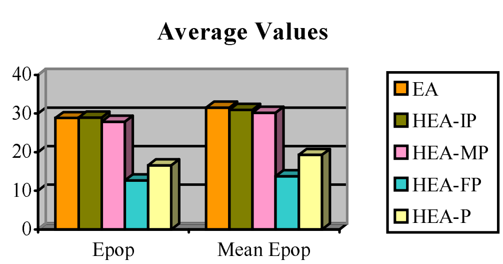

Figure 6: Average minimum Epop for *ft20* instance

Also analizing *ft20* instance (table 7), all HEA versions improve results. Regarding minimum *Ebest*, EA produces an error of 10% while the larger minimum *Ebest* in any of the hybrid algorithms is below 9.7%. Under HEA-IP, best results (6.7%) are obtained when 90% of the best individuals are submitted to TS before evolution takes place. HEA-MP provides the best minimum error (4.12%) when 100% of the individuals and the 90% of the best or worst individuals are fine-tuned in intermediate populations. Highest quality of results is obtained also under HEA-P (1.11%). Regarding average *Ebest* values HEA-P is again the best alternative, and incrementing the probability of the trials implies a continuous reduction in the considered values. This means that in average this version provides better quality results. When looking at *Epop,* HEA-FP again provides both the best minimum and average *Epop* values (12.6% and 13.7%). Other approaches with values between 14.1 and 30.6%, show a higher diversity of the final population.

Experiment		Min. Ebest	Av. Ebest	Min. Epop	Av. Epop
EA		10.38627	12.38627	28.88425	31.55082
HEA-IP	100%	7.29614	9.21888	27.73350	30.68957
	Best 90%	6.69528	9.55651	30.64979	32.14714
	Best 75%	8.75536	10.09061	29.27245	30.34118
	Best 50%	9.69957	11.07296	28.30629	31.36962
	Worst 90%	7.81116	9.55651	28.11479	29.92469
	Worst 75%	8.75536	10.89175	29.66301	31.07095
	Worst 50%	8.24034	10.75823	29.07806	30.98206
HEA-MP	100%	4.12017	6.21459	27.54370	29.58471
	Best 90%	4.12017	6.01717	27.16579	31.02871
	Best 75%	6.09442	6.70386	28.49837	30.18414
	Best 50%	4.54936	6.40343	30.64979	30.64979
	Worst 90%	4.12017	6.4978541	27.5437	30.749292
	Worst 75%	4.89270	6.8154506	27.92386	29.350377
	Worst 50%	5.23605	6.9613734	26.23075	30.490894
HEA-FP		5.23605	7.26180	**12.64686**	**13.70633**
HEA-P	**0.1**	2.83262	3.73391	21.92743	24.67235
	0.2	1.54506	2.41202	16.15279	20.27030
	0.4	**1.11588**	2.27468	14.14483	17.02374
	0.5	1.54506	**2.24034**	14.14483	15.34118

Table 7: *ft20* instance

6. CONCLUSIONS

This work shows how to hybridize an evolutionary algorithm incorporating conventional heuristics at different stages of the evolutionary process, experimenting with the selection of distinct samples of individuals from the population. In this hybridization, a well-known local search method (tabu search) is used. The main issue here was quality of results disregarding computational effort. All these algorithms provide optimal or near optimal solutions for the JSSP using operation based representation as chromosome representation.

When contrasted against a similar conventional evolutionary approach all of the hybrid approaches show a better performance in the quality of solutions provided.

Regarding quality of results, the HEA-P hybrid option, which incorporates the local search heuristic as an add-on to the basic loop of the evolutionary algorithm, provided the optimal or the best near-optimal schedules. Also solutions of good quality, but inferior than those from HEA-P, are obtained when fine-tuning is applied to the whole final population, but requiring less CPU time.

These preliminary experiments used fixed parameter settings and obtained promising results. Consequently to improve these results, future work will include dynamic control of parameters depending on the progress of the search process.

In future works with TS, an interesting issue will be to consider other rules to incorporate penalty measures for evaluating solutions trying to diversify the search.

7. ACKNOWLEDGEMENTS

We acknowledge the co-operation 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.

8. REFERENCES

[1] Davis, L., editor, Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, 1991.

[2] Garey, M.R. and D.S. Hohnson, Computer and intractability, Freeman, San Francisto, 1979.

[3] Gen, M. y Runwei, C., *Genetic Algorithms and Engineering Design*, Wiley-Interscience Publication John Wiley & Sons, Inc, 1997.

[4] Gen, M., Y. Tsujimura, and E. Kubota, "Solving Job-Shop Scheduling Problem Using Genetic Algorithms", *Proceedings of the 16th International Conference on Computers and Industrial Engineering*, 1994, pag. 576-579.

[5] Kubota, A., "Study on Optimal Scheduling for Manufacturing System by Genetic Algorithms", Master Thesis, Ashikaga Institute of Technology, Ashikaga, Japan, 1995.

[6] 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.

[7] Glover F. and M. Laguna. Tabu Search, Kluver, London (1997).

[8] Michalewicz, Z. and D.B. Fogel. How to Solve It: Modern Heuristics, Springer, Germany 2000.

[9] Minetti, G.F., C. Salto, H.A Alfonso, R.H. Gallard, "A comparison of Two Multirecombinated Evolutionary Algorithms for the Job Shop Scheduling Problem", Proceedings del VI Congreso Argentino de Ciencias de la Computación, 2000.

[10] Pinedo, M, *Scheduling: Theory, Algorithms and Systems*, Prentice Hall, 1995.

[11] Renders, J., and H. Bersini, "Hybridizing Genetic Algorithms with Hill-Climbing Methods for Global Optimization: Two Possible Ways", Proc. of the First IEEE Conference on Evolutionary Computation, 1994, pp. 312-317.

[12] Salto, C., H.A. Alfonso, and R.H. Gallard,, "Multiplicity and Incest Prevention in Evolutionary Algorithms to Deal with de Job Shop Problem", Proceedings of the Second ICSC Symposium on Engineering of Intelligent Systems, 2000, pp. 451-457.

[13] Salto, C., G.F. Minetti, H.A Alfonso, and R.H. Gallard,, “A Hybrid Evolutionary Algorithm: Multirecombination with Priority Rule Based Representation for the Job Shop Scheduling Problem”, Proceedings del VI Congreso Argentino de Ciencias de la Computación, 2000.

[14] Salto, C., H.A. Alfonso, and R.H. Gallard, “Evolutionary Algorithms and Local Search to Solve Scheduling Problems”, Proceedings of Fourth International ICSC Symposium on Soft Computing (SOCO 2001), 2001 pp. 131.

[15] Salto, C., H.A. Alfonso, and R.H. Gallard, "A Hybrid EA Approach Used to Solve the Job Shop Problem", submitted to ICS 2001, Orlando, USA.

[16] Salto, C., H.A. Alfonso, and R.H. Gallard, "Performance Evaluation of Evolutionary Algorithms under Different Representations when Solving The JSS Problem", submitted to ICS 2001, Orlando, USA.

[17] Vaessens, R.J.M., E.H.L. Aarts and J.K. Lenstra, Job Shop Scheduling by Local Search, INFORMS J. on Computing, 8:302--117, 1996.