5

Evolutionary Techniques in Multi-Objective Optimization Problems in Non-Standardized Production Processes

Mariano Frutos[1], Ana C. Olivera[2] and Fernando Tohmé[3]
[1]*Department of Engineering,*
[2]*Department of Computer Science & Engineering,*
[3]*Department of Economics,*
Universidad Nacional del Sur and CONICET,
Argentina

1. Introduction

To schedule production in a Job-Shop environment means to allocate adequately the available resources. It requires to rely on efficient optimization procedures. In fact, the Job-Shop Scheduling Problem (JSSP) is a NP-Hard problem (Ullman, 1975), so ad-hoc algorithms have to be applied to its solution (Frutos et al., 2010). This is similar to other combinatorial programming problems (Olivera et al., 2006), (Cortés et al., 2004). Most instances of the Job-Shop Scheduling Problem involve the simultaneous optimization of two usually conflicting goals. This one, like most multi-objective problems, tends to have many solutions. The Pareto frontier reached by an optimization procedure has to contain a uniformly distributed number of solutions close to the ones in the true Pareto frontier. This feature facilitates the task of the expert who interprets the solutions (Kacem et al., 2002). In this paper we present a Genetic Algorithm linked to a Simulated Annealing procedure able to schedule the production in a Job-Shop manufacturing system (Cortés et al., 2004), (Tsai & Lin, 2003), (Wu et al., 2004), (Chao-Hsien & Han-Chiang, 2009).

1.1 JSSP treatments: State of the art

The huge literature on the topic presents a variety of solution strategies that go from simple priority rules to sophisticated parallel branch-and-bound algorithms. A particular variety of scheduling problem is the JSSP. Muth and Thompson's 1964 (Muth & Thompson, 1964) book Industrial Scheduling presented the JSSP, basically in its currently known form. Even before, Jackson in 1956 (Jackson, 1956) generalized the flow-shop algorithm of Johnson (1954) (Johnson, 1954) to yield a job-shop algorithm. In 1955, Akers and Friedman (Akers & Friedman, 1955) gave a Boolean representation of the procedure, which later Roy and Sussman (1964) (Roy & Sussman, 1964) described by means of a disjunctive graph, while Egon Balas, already in 1969 (Balas, 1969), applied an enumerative approach that could be better understood in terms of this graph. Giffler and Thompson (1960) (Giffler & Thomson, 1960) presented an algorithm based on rule priorities to guide the search. For these reasons,

the problem was already part of the folklore in Operations Research years before its official inception. The JSSP generated a huge literature. Its resiliency made it an ideal problem for further study. Besides, its usefulness made it a problem worth to scrutinize. Due to its complexity, several alternative presentations of the problem have been tried (Cheng & Smith, 1997), (Sadeh & Fox, 1995), (Crawford & Baker, 1994), (De Giovanni & Pezzella, 2010), in order to apply particular algorithms like Clonal Selection (Cortés Rivera et al., 2003), Taboo Search (Armentano & Scrich, 2000), Ant Colony Optimization (Merkle & Middendorf, 2001), Genetic Algorithms (Zalzala & Flemming, 1997), Priority Rules (Panwalker & Iskander, 1977), Shifting Bottlenecks (Adams et al., 1998), etc. The performance of these meta-heuristic procedures varies, and some seem fitter than others (Chinyao & Yuling, 2009).

1.2 Multi-objective optimization: Basic concepts

Our goal in this section is to characterize the general framework in which we will state the Job-Shop problem. We assume, without loss of generality, that there are several goals (objectives) to be minimized. Then, we seek to find a vector $\vec{x}^* = [x_1^*,...,x_n^*]^T$ of decision variables, satisfying q inequalities $g_i(\vec{x}) \geq 0,\ i = 1,...,q$ as well as p equations $h_i(\vec{x}) = 0,\ i = 1,...,p$, such that $\vec{f}(\vec{x}) = [f_1(\vec{x}),..., f_k(\vec{x})]^T$, a vector of k functions, each one corresponding to an objective, defined over the decision variables, attains its minimum. The class of the decision vectors satisfying the q inequalities and the p equations is denoted by Ω and each $\vec{x} \in \Omega$ is a feasible alternative. A $\vec{x}^* \in \Omega$ is Pareto optimal if for any $\vec{x} \in \Omega$ and every $i = 1,...,k$, $f_i(\vec{x}^*) \leq f_i(\vec{x})$. That is, if there is no $\vec{x}$ that improves some objectives without worsening the others. To simplify the notation, we say that a vector $\vec{u} = [u_1,...,u_n]^T$ dominates another, $\vec{v} = [v_1,...,v_n]^T$ (denoted $\vec{u} \prec \vec{v}$) if and only if $\forall i \in \{1,...,k\}$, $u_i \leq v_i \wedge \exists i \in \{1,...,k\} : u_i < v_i$. Then, the set of Pareto optima is $P^* = \{\vec{x} \in \Omega \mid \neg\, \exists\, \vec{x}' \in \Omega, \vec{f}(\vec{x}') \prec \vec{f}(\vec{x})\}$ while the corresponding Pareto frontier is $FP^* = \{\vec{f}(\vec{x}), \vec{x} \in P^*\}$. The search of the Pareto frontier is the main goal of Multi-Objective Optimization.

2. Flexible job-shop scheduling problem

The JSSP can be described as that of organizing the execution of n jobs on m machines. We assume a finite number of tasks, $\{J_j\}_{j=1}^n$. These tasks must be processed by a finite number of machines $\{M_k\}_{k=1}^m$. To process a task J_j in a machine M_k is denoted by O_{jk}^i, where i indicates the order in which a class of operations $\{S_j\}_{j=1}^n$ is applied on a task J_j. O_{jk}^i requires the uninterrupted use of a machine M_k for a period τ_{jk}^i (the processing time) at a cost υ_{jk}^i (see Table 1). A particular case is Flexible JSSP, in which the allocation of O_{jk}^i on M_k is undifferentiated, which means that each O_{jk}^i can be processed by any of the machines in $\{M_k\}_{k=1}^m$.

After allocating the operations, we obtain a finite class E of groupings of the $O_{jk}^i s$ on the same machine. We denote each of these groupings as E_k, for $k = 1,...,m$. A key issue here is the scheduling of activities, i.e. the determination of the starting time t_{jk}^i of each O_{jk}^i. The Flexible JSSP demands a procedure to handle its two sub-problems: the allocation of the $O_{jk}^i s$ on the different $M_k s$ and their sequencing, guided by the goals to reach. That is, to find optimal levels of Makespan (Processing Time) (see Eq. 1) and Total Operation Costs (see Eq. 2).

MF01 / Problem 3 × 4 with 8 operations (flexible)									
J_j	O_{jk}^i	M_1		M_2		M_3		M_4	
		τ_{j1}^i	υ_{j1}^i	τ_{j2}^i	υ_{j2}^i	τ_{j3}^i	υ_{j3}^i	τ_{j4}^i	υ_{j4}^i
J_1	O_{1k}^1	1	10	3	8	4	6	1	9
	O_{1k}^2	3	4	8	2	2	10	1	12
	O_{1k}^3	3	8	5	4	4	6	7	3
J_2	O_{2k}^1	4	7	1	16	1	14	4	6
	O_{2k}^2	2	10	3	8	9	3	3	8
	O_{2k}^3	9	3	1	15	2	10	2	13
J_3	O_{3k}^1	8	6	6	8	3	12	5	10
	O_{3k}^2	4	11	5	8	8	6	1	18

Table 1. Flexible Job-Shop Scheduling Problem

$$f1: C_{\max}^j = \sum_{i \in O(j)} \max_{k \in M} (t_{ij}^k + \tau_{ij}^k) \tag{1}$$

$$f2: \sum_j \sum_i \sum_k x_{jk}^i \upsilon_{jk}^i \tag{2}$$

Where $x_{jk}^i = 1$ if $O_{jk}^i \in E_k$ and 0 otherwise. On the other hand $\sum_k x_{jk}^i = 1$. Besides, $t_{jk}^i = max\ (t_{jh}^{(i-1)} + \tau_{jh}^{(i-1)},\ t_{pk}^s + \tau_{pk}^s,\ 0)$ for each pair O_{jh}^{i-1}, $O_{pk}^s \in E_k$ and all machines M_k, M_h and operations S_i, S_s.

3. Hybrid genetic algorithm

Due to its many advantages, evolutionary algorithms have become very popular for solving multi-objective optimization problems (Ztzler et al., 2001), (Coello Coello et al., 2002). Among the evolutionary algorithms used, some of the most interesting are Genetic Algorithms (GA) (Goldberg, 1989). To represent the individuals, we use a variant of (Wu et al., 2004). Since the Flexible JSSP has two subproblems, the Hybrid Genetic Algorithm (HGA) presented here operates over two chromosomes. The first one represents the allocation A_{jk}^i of each O_{jk}^i to every M_k. We denote with values between 0 and (m - 1) the allocation of each M_k, that is, for m = 4, we might have something like 0→M_1, 1→M_2, 2→M_3 and 3→M_4. The second chromosome represents the sequencing of the O_{jk}^i already assigned to each of the $M_k(\forall O_{jk}^i \in E_k)$. We denote with values between 0 and (n! - 1) the sequence of J_j in each M_k. That is, for n = 3, we may have 0→$J_1J_2J_3$, 1→$J_1J_3J_2$, 2→$J_2J_1J_3$, 3→$J_2J_3J_1$, 4→$J_3J_1J_2$ and 5→$J_3J_2J_1$ (see Table 2).

The algorithm NSGAII (Non-Dominated Sorting Genetic Algorithm II) (Deb et al., 2002), creates an initial population, be it random or otherwise. NSGAII uses an elitist strategy joint with an explicit diversity mechanism. Each individual candidate solution i is assumed to

have an associated rank of non-dominance r_i and a distance d_i which indicates the radius of the area in the search space around i not occupied by another solution (see Eq. 3). A solution i is preferred over j if $r_i < r_j$. When i and j have the same rank, i is preferred if $d_i > d_j$. Let Y_i be an ordered class of individuals with same rank as i and f_j^{i+1} the value for objective j for the individual after i, while f_j^{i-1} is the value for the individual before i. $f_j^{\max}$ is the maximal value for j among Y_i while $f_j^{\min}$ is the minimal value among Y_i. The distances consider all the objective functions and attach an infinite value to the extreme solutions in Y_i. Since these yield the best values for one of the objective functions on the frontier, the resulting distance is the sum of the distances for the N objective functions.

MF01 / Problem 3 × 4 with 8 operations (flexible)						
J_j	O_{jk}^i	M_k	M_1	M_2	M_3	M_4
		Chr.	3	3	0	5
J_1	O_{1k}^1	2				
	O_{1k}^2	1				
	O_{1k}^3	0				
J_2	O_{2k}^1	1				
	O_{2k}^2	2				
	O_{2k}^3	3				
J_3	O_{3k}^1	0				
	O_{3k}^2	3				

0→$J_1J_2J_3$, 1→$J_1J_3J_2$, 2→$J_2J_1J_3$, 3→$J_2J_3J_1$, 4→$J_3J_1J_2$ and 5→$J_3J_2J_1$ / 0→MB_{1B}, 1→MB_{2B}, 2→MB_{3B}, 3→MB_4

Table 2. Chromosome encoding process

$$d_i = \sum_{j=1}^{N} (f_j^{i+1} - f_j^{i-1}) \Big/ (f_j^{max} - a_j^{min}) \tag{3}$$

Starting with a population P_t a new population of descendants Q_t obtains. These two populations mix to yield a new one, R_t of size $2N$ (N is the original size of P_t). The individuals in R_t are ranked with respect the frontier and a new population P_{t+1} obtains applying a tournament selection to R_t. After experimenting with several genetic operators we have chosen the uniform crossover for the crossover and two-swap for mutation (Fonseca & Fleming, 1995). After the individuals have been affected by these operators and before allowing them to become part of a new population we apply an improvement operator (Frutos & Tohmé, 2009). This operator has been designed following the guidelines of Simulated Annealing (Dowsland, 1993). This complements the genetic procedure. For the change of structure of both chromosomes we select a gene at random and change its value. This is repeated $M = (1/T) + \omega$, where T corresponds to the actual temperature determined

up from a cooling coefficient (α) while ω is a control parameter ensuring sufficient permutations, particularly when the temperature is high. Summarizing all this, the relevant parameters for this phase of the procedure are the initial temperature (T_i), the final one (T_f), the cooling parameter (α) and the control parameter (ω). The general layout of the whole procedure is depicted in Fig. 1.

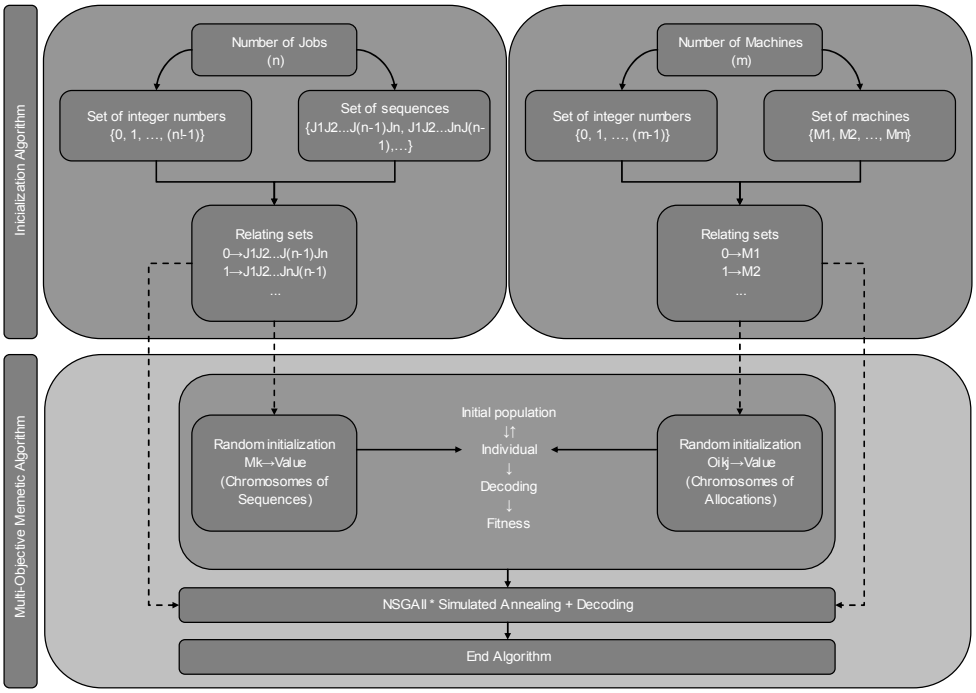


Fig. 1. Lay-out of the Hybrid Genetic Algorithm

4. Practical experiences

The parameters and characteristics of the computing equipment used during these experiments were as follows: size of the population: 200, number of generations: 500, type of crossover: uniform, probability of crossover: 0.90, type of mutation: two-swap, probability of mutation: 0.01, type of local search: simulated annealing (T_i: 850, T_f: 0.01, α: 0.95, ω: 10), probability of local search: 0.01, CPU: 3.00 GHZ and RAM: 1.00 GB. We worked with the PISA tool (A Platform and Programming Language Independent Interface for Search Algorithms) (Bleuler et al., 2003). The results obtained by means of HGA were compared to those yield by Greedy Randomized Adaptive Search Procedures (GRASP) (Binato et al., 2001), Taboo Search (TS) (Armentano & Scrich, 2000) and Ant Colony Optimization (ACO) (Heinonen & Pettersson, 2007). For the problems MF01, MF02, MF03, MF04 and MF05 (Frutos et al., 2010), we show the results for the multi-objective analysis based on Makespan (f_1, (1)) and Total Operation Costs (f_2, (2)). They were obtained by running each algorithm 10 times.

For each algorithm the sets of undominated solutions $P_1, P_2, ..., P_{10}$ were obtained as well as the superpopulation $P_T = P_1 \cup P_2 \cup ... \cup P_{10}$. From each superpopulation a class of undominated solutions was extracted, constituting the Pareto frontier for each algorithm. To obtain an approximation to the true Pareto front (Approximate Pareto Frontier), we take the fronts of each algorithm, from which all the dominated solutions are eliminated. These are detailed in Table 3 (MF01), Table 4 (MF02), Table 5 (MF03), Table 6 (MF04) and Table 7 (MF05), and are shown in Fig. 2 (MF01), Fig. 3 (MF02), Fig. 4 (MF03), Fig. 5 (MF04) and Fig. 6 (MF05).

MF01 / Problem 3 × 4 with 8 operations (flexible)										
	HGA [1]		GRASP [2]		TS [3]		ACO [4]		Approach	
Solutions	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2
3x4_1	6	66	6	70	6	66	6	66	6	66
3x4_2	7	62	7	65	7	62	7	62	7	62
3x4_3	8	55	8	61	8	55	8	57	8	55
3x4_4	9	51	9	57	9	51	9	51	9	51
3x4_5	10	47	10	50	10	48	10	47	10	47
3x4_6	11	43	11	47	11	44	11	43	11	43
3x4_7	13	42	13	43	13	43	13	42	13	42
3x4_8	-	-	-	-	15	41	-	-	15	41
3x4_9	17	40	17	40	-	-	17	40	17	40
3x4_10	-	-	-	-	20	39	-	-	20	39
3x4_11	22	38	22	38	-	-	22	38	22	38
3x4_12	-	-	-	-	25	37	-	-	25	37
3x4_13	27	36	27	37	-	-	27	36	27	36
3x4_14	28	35	28	35	28	35	28	35	28	35
3x4_15	30	34	30	34	30	34	30	34	30	34
3x4_16	31	33	-	-	31	33	31	33	31	33
3x4_17	32	31	32	32	32	32	32	32	32	31
3x4_18	35	29	35	29	35	29	35	29	35	29
Mean Time	5,325 sec.		2,147 sec.		4,673 sec.		3,218 sec.		-	

[1](Frutos et al., 2010), [2](Binato et al., 2001), [3](Armentano & Scrich, 2000) and [4](Heinonen & Pettersson, 2007)

Table 3. Solutions for MF01

MF02 / Problem 4 × 5 with 12 operations (flexible)										
	HGA [1]		GRASP [2]		TS [3]		ACO [4]		Approach	
Solutions	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2
4x5_1	16	148	16	152	16	148	16	148	16	148
4x5_2	17	142	17	146	17	142	17	142	17	142
4x5_3	19	139	19	140	19	139	19	140	19	139
4x5_4	20	135	20	136	20	136	20	135	20	135
4x5_5	22	130	22	132	22	130	22	130	22	130
4x5_6	25	124	25	128	25	124	25	124	25	124
4x5_7	26	122	26	122	26	122	26	122	26	122
4x5_8	28	118	28	118	28	118	28	118	28	118
4x5_9	-	-	-	-	30	117	30	117	30	117
4x5_10	31	115	31	116	31	115	-	-	31	115
4x5_11	34	108	34	110	34	110	34	108	34	108
4x5_12	38	102	38	102	-	-	38	102	38	102
4x5_13	39	99	39	100	39	99	39	99	39	99
4x5_14	42	95	42	97	42	95	42	95	42	95
4x5_15	45	90	45	94	45	90	45	94	45	90
4x5_16	50	81	50	83	50	83	50	81	50	81
4x5_17	52	79	52	79	52	79	53	79	52	79
4x5_18	56	68	56	72	-	-	56	72	56	68
4x5_19	-	-	-	-	57	67	-	-	57	67
4x5_20	58	65	-	-	58	65	58	65	58	65
4x5_21	61	60	61	60	61	60	61	60	61	60
4x5_22	63	57	63	58	63	58	63	58	63	57
4x5_23	-	-	-	-	-	-	65	55	65	55
4x5_24	66	53	66	53	66	53	66	53	66	53
4x5_25	67	50	67	52	67	50	-	-	67	50
4x5_26	-	-	-	-	68	48	-	-	68	48
4x5_27	69	42	69	46	69	42	69	42	69	42
4x5_28	71	36	71	36	71	36	71	36	71	36
Mean Time	15,885 sec.		6,405 sec.		13,940 sec.		9,602 sec.		-	

[1](Frutos et al., 2010), [2](Binato et al., 2001), [3](Armentano & Scrich, 2000) and [4](Heinonen & Pettersson, 2007)

Table 4. Solutions for MF02

MF03 / Problem 10 × 7 with 29 operations (flexible)										
	HGA [1]		GRASP [2]		TS [3]		ACO [4]		Approach	
Solutions	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2
10x7_1	15	393	15	393	15	393	15	393	15	393
10x7_2	16	387	16	391	16	387	16	387	16	387
10x7_3	17	383	17	385	17	383	17	383	17	383
10x7_4	18	379	18	379	18	379	18	379	18	379
10x7_5	19	375	19	375	19	375	19	375	19	375
10x7_6	21	368	21	372	21	368	21	368	21	368
10x7_7	23	360	23	360	23	360	23	360	23	360
10x7_8	24	351	24	355	24	351	24	351	24	351
10x7_9	25	347	-	-	25	347	-	-	25	347
10x7_10	27	342	27	342	27	342	27	342	27	342
10x7_11	33	319	33	325	33	319	33	319	33	319
10x7_12	37	291	37	297	37	295	37	295	37	291
10x7_13	45	260	45	260	45	260	45	260	45	260
10x7_14	50	238	50	241	50	238	50	238	50	238
10x7_15	61	194	61	211	61	202	61	202	61	194
10x7_16	-	-	-	-	72	150	72	158	72	150
10x7_17	78	137	78	142	78	137	78	137	78	137
10x7_18	89	98	89	107	89	104	89	98	89	98
10x7_19	96	82	-	-	-	-	-	-	96	82
10x7_20	109	48	-	-	109	57	109	48	109	48
10x7_21	116	34	116	41	116	34	116	34	116	34
10x7_22	122	29	122	29	122	29	122	29	122	29
Mean Time	21,502 sec.		7,669 sec.		18,869 sec.		15,994 sec.		-	

[1](Frutos et al., 2010), [2](Binato et al., 2001), [3](Armentano & Scrich, 2000) and [4](Heinonen & Pettersson, 2007)

Table 5. Solutions for MF03

MF04 / Problem 10 × 10 with 30 operations (flexible)										
	HGA [1]		GRASP [2]		TS [3]		ACO [4]		Approach	
Solutions	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2
10x10_1	7	282	7	282	7	282	7	282	7	282
10x10_2	8	267	8	274	8	267	8	267	8	267
10x10_3	10	254	10	254	10	254	10	254	10	254
10x10_4	11	241	11	246	11	241	11	241	11	241
10x10_5	13	224	13	224	13	224	13	224	13	224
10x10_6	15	205	15	205	15	205	15	205	15	205
10x10_7	16	198	16	198	16	198	16	198	16	198
10x10_8	-	-	-	-	18	186	-	-	18	186
10x10_9	19	176	19	185	19	176	19	180	19	176
10x10_10	23	148	23	148	23	148	23	148	23	148
10x10_11	25	137	25	137	25	137	25	137	25	137
10x10_12	28	113	-	-	-	-	-	-	28	113
10x10_13	29	107	29	115	29	107	29	111	29	107
10x10_14	31	87	31	96	31	90	31	90	31	87
10x10_15	33	78	33	83	33	78	33	78	33	78
10x10_16	34	73	34	73	34	73	34	73	34	73
10x10_17	36	62	36	67	36	62	36	62	36	62
10x10_18	37	58	37	58	37	58	37	58	37	58
10x10_19	38	57	38	57	38	57	38	57	38	57
10x10_20	41	51	41	54	41	55	41	55	41	51
10x10_21	44	49	44	51	44	49	44	49	44	49
10x10_22	47	43	47	48	-	-	-	-	47	43
10x10_23	50	42	50	42	50	42	50	42	50	42
10x10_24	53	40	53	40	53	40	53	40	53	40
10x10_25	-	-	-	-	-	-	56	37	56	37
10x10_26	57	34	57	34	57	34	57	34	57	34
10x10_27	60	30	60	30	60	30	60	30	60	30
Mean Time	31,439 sec.		11,214 sec.		27,590 sec.		22,999 sec.		-	

[1](Frutos et al., 2010), [2](Binato et al., 2001), [3](Armentano & Scrich, 2000) and [4](Heinonen & Pettersson, 2007)

Table 6. Solutions for MF04

MF05 / Problem 15 × 10 with 56 operations (flexible)										
	HGA [1]		GRASP [2]		TS [3]		ACO [4]		Approach	
Solutions	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2
15x10_1	23	799	23	799	23	799	23	799	23	799
15x10_2	25	749	25	749	25	749	25	749	25	749
15x10_3	26	731	26	731	26	731	26	731	26	731
15x10_4	27	719	27	719	27	719	27	719	27	719
15x10_5	30	678	30	693	30	678	30	687	30	678
15x10_6	32	646	-	-	-	-	32	646	32	646
15x10_7	33	631	33	631	33	631	33	631	33	631
15x10_8	35	609	35	615	35	609	35	609	35	609
15x10_9	38	575	38	587	38	578	38	575	38	575
15x10_10	-	-	-	-	41	561	41	561	41	561
15x10_11	41	519	43	519	43	519	43	519	43	519
15x10_12	44	484	44	484	44	484	44	484	44	484
15x10_13	46	437	46	452	46	448	46	437	46	437
15x10_14	49	411	49	411	49	411	49	411	49	411
15x10_15	52	379	52	379	52	379	52	379	52	379
15x10_16	53	346	53	346	53	346	53	346	53	346
15x10_17	55	314	55	322	55	318	55	318	55	314
15x10_18	57	276	-	-	-	-	-	-	56	276
15x10_19	-	-	-	-	58	266	-	-	58	266
15x10_20	62	220	62	242	62	232	62	242	62	220
15x10_21	67	209	67	212	67	209	67	209	67	209
15x10_22	71	195	71	204	71	195	71	195	71	195
15x10_23	75	178	75	181	75	178	75	178	75	178
15x10_24	88	153	88	157	88	153	88	153	88	153
15x10_25	92	135	92	147	92	140	92	140	92	135
15x10_26	101	122	101	134	101	129	101	122	101	122
15x10_27	112	114	112	125	112	114	112	114	112	114
15x10_28	-	-	-	-	-	-	119	97	119	97
15x10_29	127	86	127	86	127	86	127	86	127	86
15x10_30	135	73	135	73	135	73	135	73	135	73
15x10_31	138	56	138	56	138	56	138	56	138	56
Mean Time	42,288 sec.		15,084 sec.		37,110 sec.		35,552 sec.		-	

[1](Frutos et al., 2010), [2](Binato et al., 2001), [3](Armentano & Scrich, 2000) and [4](Heinonen & Pettersson, 2007)

Table 7. Solutions for MF05

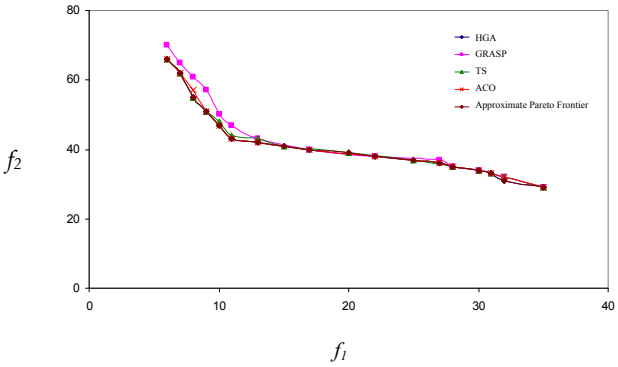


Fig. 2. Makespan *vs.* Total Operation Costs (MF01)

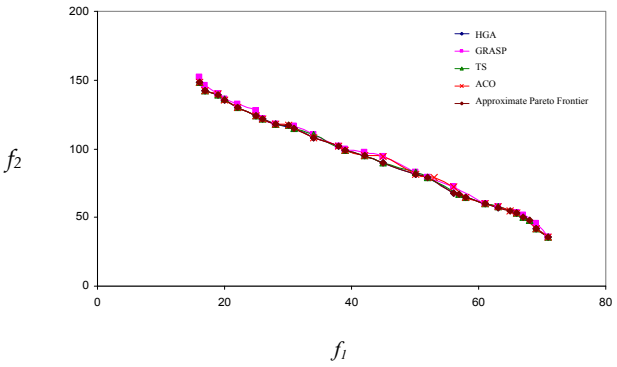


Fig. 3. Makespan *vs.* Total Operation Costs (MF02)

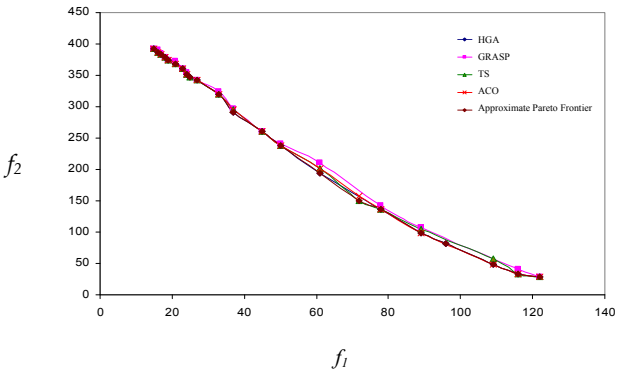


Fig. 4. Makespan *vs.* Total Operation Costs (MF03)

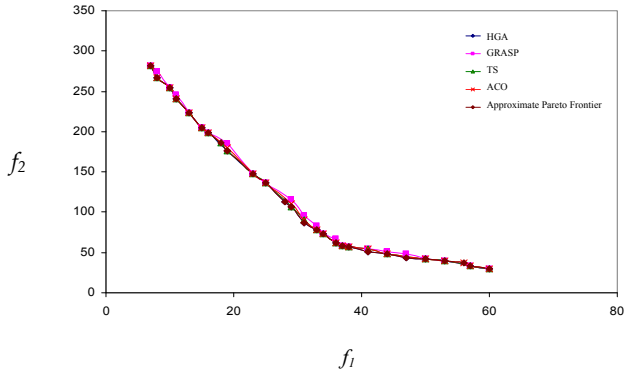


Fig. 5. Makespan *vs.* Total Operation Costs (MF04)

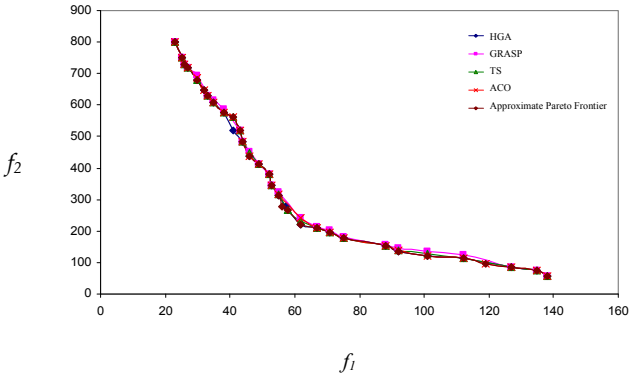


Fig. 6. Makespan *vs.* Total Operation Costs (MF05)

In order to compare the results of the algorithms and establish the better option for the Flexible JSSP, several tests were applied over the solutions. First, we consider a dominance ranking among the different algorithms. One-tailed Mann-Whitney rank sum (Conover, 1999) was run over the results (Ranktest, Table 8 (MF01), Table 9 (MF02), Table 10 (MF03), Table 11 (MF04) and Table 12 (MF05)). The outcomes are summarized in Table 1. None of the results for MF01, MF02, MF03, MF04 and MF05 is statistically significant at an overall significance level α=0.05. This indicates that no algorithm generate approximation sets that are significantly better. Next, we considered unary quality indicators using normalized approximation sets. Then, we applied the unary indicators (unary hypervolume indicator I_H, unary epsilon indicatior I_e^1 and R indicator I_{R2}^1) on the normalized approximation sets as well as on the reference set generated by PISA (I_H, I_e^1 and I_{R2}^1, Table 8 (MF01), Table 9 (MF02), Table 10 (MF03), Table 11 (MF04) and Table 12 (MF05)). Again, no significant differences were found at the 0.05 level.

Test for Problem MF01				
Ranktest				
	HGA	GRASP	TS	ACO
HGA	-	0,4668807	0,5248382	0,5000000
GRASP	0,5331193	-	0,5578024	0,5331193
TS	0,4751618	0,4421976	-	0,4751618
ACO	0,5000000	0,4668807	0,5248382	-
I_H				
	HGA	GRASP	TS	ACO
HGA	-	0,4849375	0,5193377	0,5451365
GRASP	0,5150625	-	0,5537379	0,5782757
TS	0,4806623	0,4462621	-	0,5793756
ACO	0,4548635	0,4217243	0,4206244	-
I_e^1				
	HGA	GRASP	TS	ACO
HGA	-	0,4668807	0,5000000	0,5248382
GRASP	0,5331193	-	0,5331193	0,5567435
TS	0,5000000	0,4668807	-	0,5578024
ACO	0,4751618	0,4421976	0,4751618	-
I_{R2}^1				
	HGA	GRASP	TS	ACO
HGA	-	0,4560385	0,4883887	0,5126501
GRASP	0,5439615	-	0,5207389	0,5438144
TS	0,5116113	0,4792611	-	0,5448488
ACO	0,4873499	0,4561856	0,4551512	-

Table 8. Comparing HGA, GRASP, TS and ACO (MF01)

Test for Problem MF02				
Ranktest				
	HGA	GRASP	TS	ACO
HGA	-	0,4507347	0,4364051	0,5123441
GRASP	0,5492653	-	0,4920168	0,5614884
TS	0,5635949	0,5079832	-	0,5792996
ACO	0,4876559	0,4385116	0,4207004	-
I_H				
	HGA	GRASP	TS	ACO
HGA	-	0,4554712	0,5065161	0,4369711
GRASP	0,5445288	-	0,5705083	0,5110457
TS	0,4934839	0,4294917	-	0,4532833
ACO	0,5630289	0,4889543	0,5467167	-
I_e^1				
	HGA	GRASP	TS	ACO
HGA	-	0,4385116	0,4876559	0,4207004
GRASP	0,5614884	-	0,5492653	0,4920168
TS	0,5123441	0,4507347	-	0,4364051

ACO	0,5792996	0,5079832	0,5635949	-
		$I_{R2}{}^1$		
	HGA	GRASP	TS	ACO
HGA	-	0,4283282	0,4763312	0,4109306
GRASP	0,5716718	-	0,5365099	0,4805909
TS	0,5236688	0,4634901	-	0,4262707
ACO	0,5890694	0,5194091	0,5737293	-

Table 9. Comparing HGA, GRASP, TS and ACO (MF02)

Test for Problem MF03				
Ranktest				
	HGA	GRASP	TS	ACO
HGA	-	0,6430292	0,5250930	0,5052068
GRASP	0,3569708	-	0,3782149	0,3627307
TS	0,4749070	0,6217851	-	0,4791809
ACO	0,4947932	0,6372693	0,5208191	-
		I_H		
	HGA	GRASP	TS	ACO
HGA	-	0,6619159	0,5139296	0,5409620
GRASP	0,3380841	-	0,3707768	0,3928425
TS	0,4860704	0,6292232	-	0,5454012
ACO	0,4590380	0,6071575	0,4545988	-
		$I_e{}^1$		
	HGA	GRASP	TS	ACO
HGA	-	0,6372693	0,4947932	0,5208191
GRASP	0,3627307	-	0,3569708	0,3782149
TS	0,5052068	0,6430292	-	0,5250930
ACO	0,4791809	0,6217851	0,4749070	-
		$I_{R2}{}^1$		
	HGA	GRASP	TS	ACO
HGA	-	0,6224702	0,4833028	0,5087244
GRASP	0,3775298	-	0,3486810	0,3694317
TS	0,5166972	0,6513190	-	0,5128990
ACO	0,4912756	0,6305683	0,4871010	-

Table 10. Comparing HGA, GRASP, TS and ACO (MF03)

Test for Problem MF04				
Ranktest				
	HGA	GRASP	TS	ACO
HGA	-	0,4840368	0,5910066	0,4641226
GRASP	0,5159632	-	0,6017605	0,4794311
TS	0,4089934	0,3982395	-	0,3824045
ACO	0,5358774	0,5205689	0,6175955	-
		I_H		
	HGA	GRASP	TS	ACO

HGA	-	0,5407021	0,5566027	0,6414813
GRASP	0,4592979	-	0,5359183	0,6250339
TS	0,4433973	0,4640817	-	0,6138640
ACO	0,3585187	0,3749661	0,3861360	-
		I_e^1		
	HGA	GRASP	TS	ACO
HGA	-	0,5205689	0,5358774	0,6175955
GRASP	0,4794311	-	0,5159632	0,6017605
TS	0,4641226	0,4840368	-	0,5910066
ACO	0,3824045	0,3982395	0,4089934	-
		I_{R2}^1		
	HGA	GRASP	TS	ACO
HGA	-	0,5084799	0,5234329	0,6032533
GRASP	0,4915201	-	0,5039812	0,5877861
TS	0,4765671	0,4960188	-	0,5772819
ACO	0,3967467	0,4122139	0,4227181	-

Table 11. Comparing HGA, GRASP, TS and ACO (MF04)

Test for Problem MF05				
Ranktest				
	HGA	GRASP	TS	ACO
HGA	-	0,4551815	0,4673350	0,4713557
GRASP	0,5448185	-	0,5207202	0,5201772
TS	0,5326650	0,4792798	-	0,5031851
ACO	0,5286443	0,4798228	0,4968146	-
		I_H		
	HGA	GRASP	TS	ACO
HGA	-	0,4983801	0,5501285	0,5160291
GRASP	0,5016199	-	0,5658896	0,5408593
TS	0,4498715	0,4341104	-	0,4854094
ACO	0,4839709	0,4591407	0,5145906	-
		I_e^1		
	HGA	GRASP	TS	ACO
HGA	-	0,4798228	0,5296443	0,4968146
GRASP	0,5201772	-	0,5448185	0,5207202
TS	0,4713557	0,4551815	-	0,4673350
ACO	0,5031557	0,4792798	0,5326650	-
		I_{R2}^1		
	HGA	GRASP	TS	ACO
HGA	-	0,4686801	0,5173445	0,4852773
GRASP	0,5313199	-	0,5321664	0,5086278
TS	0,4826555	0,4678336	-	0,4564823
ACO	0,5147227	0,4913722	0,5435177	-

Table 12. Comparing HGA, GRASP, TS and ACO (MF05)

Finally, we note that there are no major differences between the Pareto frontiers generated by the four algorithms. Therefore, we calculated the percentage of solutions provided by each algorithm that belong to the Approximate Pareto Frontier (see Table 13).

Percentage of solutions in the Approximate Pareto Frontier				
	HGA [1]	GRASP [2]	TS [3]	ACO [4]
MF01	83,33%	27,78%	61,11%	72,22%
MF02	85,71%	25,00%	75,00%	71,43%
MF03	95,45%	31,82%	77,27%	77,27%
MF04	92,59%	51,85%	81,48%	74,07%
MF05	90,32%	45,16%	70,97%	80,65%

[1] (Frutos et al., 2010), [2](Binato et al., 2001), [3](Armentano & Scrich, 2000) and [4](Heinonen & Pettersson, 2007)

Table 13. Comparing HGA, GRASP, TS and ACO (MF01, MF02, MF03, MF04 and MF05)

5. Conclusions

We presented a Hybrid Genetic Algorithm (HGA) intended to solve the Flexible Job-Shop Scheduling Problem (Flexible JSSP). The application of HGA required the calibration of parameters, in order to yield valid values for the problem at hand, which constitute also a reference for similar problems. We have shown that this HGA yields more solutions in the Approximate Pareto Frontier than other algorithms. As said above, PISA has been used here as a guide for the implementation of our HGA. Nevertheless, PISA itself has features that we tried to overcome, making the understanding and extension of its outcomes a little bit hard. JMetal (Meta-heuristic Algorithms in Java) (Durillo et al., 2006) is already an alternative to PISA implemented on JAVA. We are currently experimenting with other techniques of local search in order to achieve a more aggressive exploration. We are also interested in evaluating the performance of the procedure over other kinds of problems to see whether it saves resources without sacrificing precision in convergence.

6. Acknowledgments

We would like to thank the economic support of the Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET) and the Universidad Nacional del Sur (UNS) for Grant PGI 24/JO56.

7. References

Adams, J.; Balas, E. & Zawack, D. (1998). The Shifting Bottleneck Procedure for job shop scheduling, *Management Science*, Vol. 34 (3), pp 391-401.

Akers, S. B. & Friedman, J. (1955). A Non-Numerical Approach to Production Scheduling Problems, *Operations Research*, Vol. 3 (4), pp 429-442.

Armentano, V. & Scrich, C. (2000). Taboo search for minimizing total tardiness in a job-shop, *International Journal of Production Economics*, Vol. 63, pp 131-140.

Balas, E. (1969). Duality in Discrete Programming: The Quadratic Case, *Management Science*, Vol. 16 (1), pp 14-32.

Binato, S.; Hery, W. J.; Loewenstern, D. M. & Resende, M. G. C. (2001). A grasp for job shop scheduling, *Essays and Surveys in Meta-heuristics*, pp. 59-80.

Bleuler, S.; Laumanns, M.; Thiele, L. & Zitzler, E. (2003). PISA, A Platform and Programming Language Independent Interface for Search Algorithms, *Proceedings of Evolutionary Multi-Criterion Optimization*, pp. 494-508.

Chao-Hsien, J. & Han-Chiang, H. (2009). A hybrid genetic algorithm for no-wait job shop cheduling problems, *Expert Systems with Applications*, Vol. 36 (3), pp 5800-5806.

Cheng, C. C. & Smith, S. F. (1997). Applyng constraint satisfaction techniques to job shop scheduling, *Annals of Operations Research*, Vol. 70, pp 327-357.

Chinyao, L. & Yuling, Y. (2009). Genetic algorithm-based heuristics for an open shop scheduling problem with setup, processing, and removal times separated, *Robotics and Computer-Integrated Manufacturing*, Vol. 25 (2), pp 314-322.

Coello Coello, C. A.; Van Veldhuizen, D. A. & Lamont, G. B. (2002). Evolutionary Algorithms for Solving Multi-Objective Problems, *Kluwer Academic Publishers*, New York.

Conover, W. (1999). Practical Nonparametric Statistics. *John Wiley & Sons*, New York.

Cortés Rivera, D.; Coello Coello, C. A. & Cortés, N. C. (2004). Job shop scheduling using the clonal selection principle, *ACDM'2004*, UK.

Cortés Rivera, D.; Coello Coello, C. A. & Cortés, N. C. (2003). Use of an Artificial Immune System for Job Shop Scheduling, *Proceedings of Second International Conference on Artificial Immune Systems*, Edinburgh, Scotland. Springer-Verlag, Lecture Notes in Computer Science, Vol. 2787, pp 1-10.

Crawford, J. M. & Baker, A. B. (1994). Experimental Results on the Application of Satisfiability Algorithms to Scheduling Problems, *Computational Intelligence Research Laboratory*.

De Giovanni, L. & Pezzella, F. (2010). An Improved Genetic Algorithm for the Distributed and Flexible Jobshop Scheduling problem, *European Journal of Operational Research*, Vol. 200 (2), pp 395-408.

Deb, K.; Pratap, A.; Agarwal, S. & Meyarivan, T. (2002). A Fast and Elitist Multi-objective Genetic Algorithm: NSGAII, *IEEE Transactions on Evolutionary Computation*, Vol. 6 (2), pp 182-197.

Dowsland, K. A. (1993). Simulated Annealing, Modern Heuristic Techniques for Combinatorial Problems, Ed. C. R. Reeves, *Blackwell Scientific Pub*, Oxford.

Durillo, J. J.; Nebro, A. J.; Luna Dorronsoro B. & Alba E. (2006). JMetal: A Java Framework for Developing Multi-Objective Optimization Metaheuristics. Departamento de Lenguajes y Ciencias de la Computación. University of Málaga. *E.T.S.I. Informática*, Campus de Teatinos.

Fonseca, C. M. & Fleming, P. J. (1995). Multi-objective genetic algorithms made easy: Selection, sharing and mating restriction, *GALESIA*, pp 45-52.

Frutos, M.; Olivera, A. C. & Tohmé, F. (2010). A Memetic Algorithm based on a NSGAII Scheme for the Flexible Job-Shop Scheduling Problem, *Annals of Operations Research*, Vol. 181, pp 745-765.

Frutos, M.; & Tohmé, F. (2009). Desarrollo de un procedimiento genético diseñado para programar la producción en un sistema de manufactura tipo job-shop, *Proceedings of VI Congreso Español sobre Meta-heurísticas, Algoritmos Evolutivos y Bioinspirados*, España.

Giffler, B. & Thomson, G. L. (1960). Algorithms for Solving Production Scheduling Problems, *Operations Reseach*, Vol. 8, pp 487-503.

Goldberg, D. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning, Addison Wesley.

Heinonen, J. & Pettersson, F. (2007). Hybrid ant colony optimization and visibility studies applied to a job-shop scheduling problem, *Applied Mathematics and Computation*, pp 989-998.

Jackson J. R. (1956). An Extension of Johnson's Results on Job Lot Scheduling, *Naval Research Logistics Quarterly*, Vol. 2, pp 201-203.

Johnson, S. M. (1954). Optimal two- and three-stage production schedules with setup times included, *Naval Research Logistics Quarterly*, Vol. 1, pp 61-68.

Kacem, I.; Hammadi, S. & Borne, P. (2002). Approach by Localization and Multi-Objective Evolutionary Optimization for Flexible Job-Shop Scheduling Problems, *IEEE Trans. Syst. Man Cybernetics*, Vol. 32.

Merkle, D. & Middendorf, M. (2001). A new approach to solve permutation scheduling problems with ant colony optimization, *Proceedings of Applications of Evolutionary Computing*, EvoWorkshops 2001, Vol. 2037, pp 484-494.

Muth, J. F. & Thompson, G. L. (1964). Industrial Scheduling, Prentice-Hall Inc.

Olivera, A. C.; Frutos, M. & Casal, R. (2006). Métodos para determinar secuencias de producción en un ambiente productivo complejo, *Proceedings of XIII Congreso Latino-Iberoamericano de Investigación Operativa*, Uruguay.

Panwalker, S. & Iskander, W. (1977). A survey of scheduling rules, *Operations Research*, Vol. 25 (1), pp 45-61.

Roy, B. & Sussman, B. (1964). Les problèmes d'ordonnancement avec contraintes disjonctives, *SEMA*.

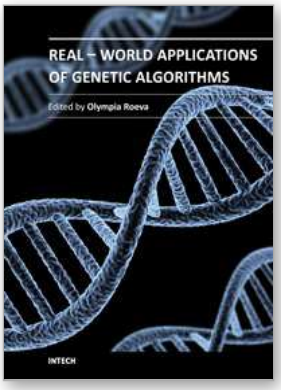
Sadeh, N. M. & Fox, M. S. (1995). Variable and value ordering heuristics for the Job Shop scheduling constraint satisfaction problem, Tecnical report CMU-RI-TR-95-39, *Artificial Intelligence Journal*.

Tsai, C. F. & Lin, F. C. (2003). A new hybrid heuristic technique for solving job-shop scheduling problem, *Intelligent Data Acquisition and Advanced Computing Systems, Second IEEE International Workshop*.

Ullman, J. D. (1975). NP-complete scheduling problems. *Journal of Computer System sciences*, Vol. 10, pp 384-393.

Wu, C. G.; Xing, X. L.; Lee, H. P.; Zhou, C. G. & Liang, Y. C. (2004). Genetic Algorithm Application on the Job Shop Scheduling Problem, *Machine Learning and Cybernetics, International Conference*, Vol. 4, pp 2102- 2106.

Zalzala, A. M. S. & Flemming, P. J. (1997). Genetic Algorithms in engineering systems, *London Institution of Electrical Engineers*.



Real-World Applications of Genetic Algorithms
Edited by Dr. Olympia Roeva

ISBN 978-953-51-0146-8
Hard cover, 376 pages
Publisher InTech
Published online 07, March, 2012
Published in print edition March, 2012

The book addresses some of the most recent issues, with the theoretical and methodological aspects, of evolutionary multi-objective optimization problems and the various design challenges using different hybrid intelligent approaches. Multi-objective optimization has been available for about two decades, and its application in real-world problems is continuously increasing. Furthermore, many applications function more effectively using a hybrid systems approach. The book presents hybrid techniques based on Artificial Neural Network, Fuzzy Sets, Automata Theory, other metaheuristic or classical algorithms, etc. The book examines various examples of algorithms in different real-world application domains as graph growing problem, speech synthesis, traveling salesman problem, scheduling problems, antenna design, genes design, modeling of chemical and biochemical processes etc.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

Mariano Frutos, Ana C. Olivera and Fernando Tohmé (2012). Evolutionary Techniques in Multi-Objective Optimization Problems in Non-Standardized Production Processes, Real-World Applications of Genetic Algorithms, Dr. Olympia Roeva (Ed.), ISBN: 978-953-51-0146-8, InTech, Available from: http://www.intechopen.com/books/real-world-applications-of-genetic-algorithms/evolutionary-techniques-in-multi-objective-optimization-problems-in-non-standardized-production-proc

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821