Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 49-12 (2016) 1272–1277

A NSGA-II and NSGA-III comparison for solving an open shop scheduling problem with resource constraints

Guillermo Campos Ciro * **Frédéric Dugardin** *
Farouk Yalaoui * **Russell Kelly** **

* *ICD-LOSI, Université de Technologie de Troyes, UMR 6281 12 rue Marie Curie, CS42060, 10004 Troyes Cedex, France (e-mail: guillermo.campos_ciro@utt.fr).*
** *Norelem SAS 5 rue des Libellules, 10280 Fontaine-Les-Grès, France (e-mail: russell.kelly@norelem.fr).*

Abstract: This paper presents an open shop scheduling problem based on a real mechanical workshop made of m machines that process n jobs. It deals with different resource constraints related to the tools allocation and the multi-skills staff assignment. Resource skills and their availability are required to execute one process task. In this work, we expose a multi-objective problem where the idea is to minimize three objectives simultaneously. The first one considers the minimization of the total flow time of jobs in the production system, then the workload balancing concerning both, humans and machines is addressed. We propose and compare two multi-objective methods: NSGA-II and NSGA-III. Computational experiments are designed according to the literature, we expose the small and large sized instances to present the general performance of these algorithms using the hyper-volume metric to compare them.

Keywords: multi-objective, scheduling algorithms, operation research, resource allocation, skills

1. INTRODUCTION

The multi-objective optimization has been a interesting topic in several domains for the researchers from 1970. Our economical and social environment evolves very quickly thanks to the technological and the knowledge advances. In this context, all enterprises: small, medium or big must deal with many challenges at the same time that are looking for the minimization or the maximization of one objective. However, next to this principal objective there will be other parallel challenges that follow the same optimization line (they deal with indirect related subjects).

Talking about scheduling problems, Wang et al. (2008) have exposed that the methods to solve the multi-objective problems have been classified in 5 types: scalars, interactives, fuzzys, metaheuristics and decision-making aids methods. The solution method depends on the objective of the study and its context.

Different multi-objective research works have applied the multi-objective evolutionary algorithms (MOEA's) as is presented by Coello et al. (2007a). This kind of methods have shown to be powerful and robust in the multi-objective problem solutions as well as they are more needed in the scheduling subjects. In that way, three categories of solutions have been proposed: the weighted sum of the objective functions, the approaches based on the population and the ones based on the concept of Pareto.

The first category combines all the sub-objectives in an only one objective where its complexity can change from one problem to another (Indraneel and Dennis (1997), Coello et al. (2007b)). So that, a mono-criteria problem can be generated adding the objectives at the same level. In that way, a linear weighted addition can be built as was exposed by Ishibuchi et al. (2008). However, this method could create perturbations in the model conception, distorting the importance of one objective with respect to the others (Collette and Siarry (2002)). The next approach based on the population considers the population evolution without taking into account the Pareto's dominance in the selection process.

On the third place, we find different methods based on the optimality of Pareto, the Non-dominated Sorting Genetic Algorithm (NSGA) proposed by Srinivas and Deb (1994). The multi-objective genetic algorithms (MOGA) by Fonseca and Fleming (1993), the GA with Pareto's niche (NPGA) by Abido (2003). Besides that, Deb et al. (2002) have developed the NSGA-II that is the second version of the NSGA. Others methods as the strength Pareto evolutionary algorithm (SPEA and SPEA2) have been exposed to solve multi-objective problems. Nowadays, the most used method to solve this kind of problems is the notion of the Pareto dominance.

This paper is organized as follows: the next section explains the used comparison criteria between the Pareto fronts, then the features of the problem are introduced, after that the solution methods are described. At the

10.1016/j.ifacol.2016.07.690

end, the results are exposed with their comments and a conclusion is provided in the last section.

2. CRITERIA FOR THE PARETO FRONTS COMPARISON

After dealing with concept of the multi-objective optimization, we will mention the different metrics that can be used to compare the fronts of the non-dominated solutions found by the different algorithms. This process is so meaningful in order to determine the quality of the solutions and the advantages of one algorithm. These criteria are divided in two types: the metrics related to one front and the metrics that compare different fronts.

Concerning the metrics that deal with only one front, we can find: the number of solutions in the optimal front n_f, the hyper-surface exposed by Collette and Siarry (2002), the hyper-volume described by While et al. (2006), the spacing presented and the metric HRS described by Collette and Siarry (2002).

On the other hand, the metrics that consider two fronts: the progression metric expose by Collette and Siarry (2002), the Laumanns et al. (2000) metric.

Next, we describe the hyper-volume metric in a deeper way. This metric will let us to determine the dominance of one algorithm over the others according to the obtained results.

2.1 The hyper-volume

The hyper-volume (HV) is a metric that has been used by many authors Huband et al. (2003), Purshouse (2003).

This metric represents the volume of the objective space that is covered by the individuals of a non-dominated solutions set (solutions that belong to a Pareto front). The volume is delimited by two points: one point that is called the anti-optimal point (A) that is defined as the worst solution inside the objective space, and a second optimal point (pseudo-optimal) that is calculated by the proposed solution method. A normalization of the objective function values is required before calculating the hyper-volume. The hyper-volume is the surface (volume), depending on the number of objective functions, that is inside of the individuals D_l (l = 1,...,Υ) of a non-dominated solutions set (ND) and the point A, as it is showed in the equation (1):

$$HV(ND, A) = \wedge(\{\bigcup h(D_l) | D_l \in ND, l = 1, ..., \Upsilon\}) \quad (1)$$

Where $h(D_l) = |D_{l1} - A_1| \times |D_{l2} - A_2| \times ... \times |D_{lC} - A_C|$, Υ is the number of solutions that are included in the Pareto front and C is the number of objective functions.

In figure 1, the surface is obtained by the union of the different rectangles created by each solution of the front (y_1, y_2 and y_3) and the reference point (A). Then, figure 2 shows the calculation of the hyper-volume with three objective functions where there are five solutions in the front (y_1, y_2, y_3, y_4 and y_5) and one reference point.

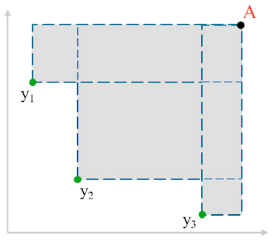

Fig. 1. Hyper-volume for a two objective functions problem

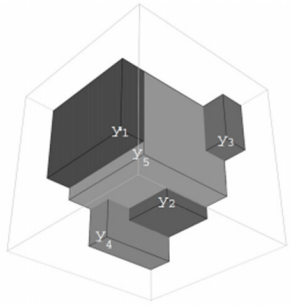

Fig. 2. Hyper-volume for a three objective functions problem

3. PROBLEM DESCRIPTION

In other related works, we have already studied an open shop scheduling problem with multi-skills human resource assignment constraints (workers) (Campos-Ciro et al. (2015b)), and also the addition of tool's availability constraints to execute one operation in Campos-Ciro et al. (2015a). In those works, we have dealt with a mono-objective problem: the minimization of the total flow time (the time that a product stays in the production system from its release date until its completion time).

In this section, we describe an open shop that has multi-skill resource constraints, tool limitations and deals with a multi-objective optimization, so we consider the minimization of three objective functions at the moment of find a schedule for our problem. It is composed of N jobs (i = 1, 2, 3, ..., N) and a set of M machines (m = 1, 2, 3, ..., M), where each job i needs at most M operations and multiple options of processing order can find according to the product job type. Each operation O_{im} must be carried out on the selected machine with a known processing time p_{im}, considering a setup time SE_{im} and taking into account some availability resource constraints (the required skills per operation and the mastered skills per worker before his assignment to execute one operation, the required tools per operation and the their availability before being assigned).

The preemption is not allowed. The setup and the processing times are considered separately (setup task can be started as soon as the machine is free). Each machine can process one job at a time and one job can only be carried out by one machine at any time. The jobs are available according to their release dates and all the machines are always available during the schedule.

Concerning resources assignment, one operation can require several types of tools to be executed but their allocation will be made only if they are available at the moment. The human resource or worker should be allocated to execute a required skill that he masters, he can only be assigned to carry out one task at a time and any worker that has been already assigned must be on the machine during the entire operation's processing time.

As mentioned above, we will optimize three objective functions. On the one hand, it is interesting to keep the minimization of the total flow time ($Min \sum F_i$) as the first objective function.

On the other hand, the study of the resource workload balancing (humans and materials) can provide an interesting solutions for our problem. In our case, the human resources assignment is made according to the worker skills, so we can find the case where some workers will be assigned to execute longer operations than the others (for example in the longest or the most polyvalent operations). Knowing that, we have chosen a second objective to minimize, the worker's workload balancing in order to distribute the different activities among the set of available workers.

Inspired by the study proposed by Ouazene et al. (2014), we have formulated our second objective function in two steps. First of all, we have to determine the value of the completion time of the last operation executed by each worker w ($E_w = \underset{i\in J, m\in L}{Max} C_{wim}$). Then, we minimize the difference between the maximal and minimal value of the obtained completion times (equation 2) to balance the worker's load in the proposed scheduling configuration.

$$Min\left(\frac{1}{N_w}\sum_{w=1}^{N_w} E_w\right) \Leftrightarrow Min\left(\underset{w=1,\ldots,N_w}{Max} E_w - \underset{w=1,\ldots,N_w}{Min} E_w\right) = Min\left(\Delta E_{RH}\right) \tag{2}$$

Figure 3 shows the operations allocated by each worker w (in our example, we present $N_w = 4$ - the same schedule of the first objective function as well as the graphical representation of ΔE_{RH}). For this reason, the second objective function is the minimization of ΔE_{RH} ($Min\Delta E_{RH}$).

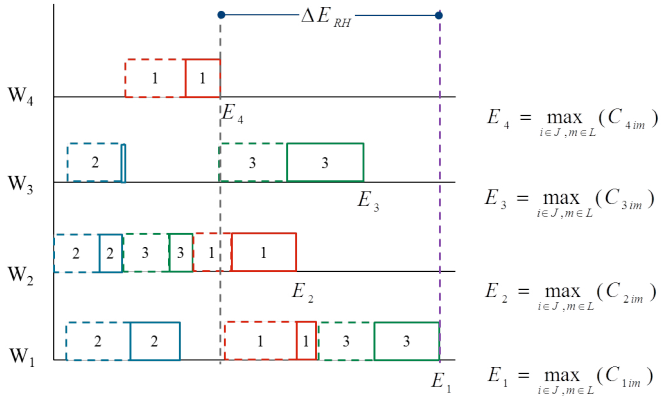

Fig. 3. OF2: Worker's workload balancing

Finally, we cope with the machine's workload as the third objective. We apply the same process, so in the first step we must calculate the completion time value of the last operation executed on machine m ($E_m = \underset{w\in W, i\in J}{Max} C_{wim}$). Next, we find the minimization of the difference between the maximal and minimal obtained completion times (equation 3).

$$Min\left(\frac{1}{M}\sum_{m=1}^{m} E_m\right) \Leftrightarrow Min\left(\underset{m=1,\ldots,M}{Max} E_m - \underset{m=1,\ldots,M}{Min} E_m\right) = Min\left(\Delta E_{RM}\right) \tag{3}$$

Figure 4 exposes the activities that have been assigned to the different machines (M = 3 in the example) and the representation of ΔE_{RM}. We define our third and final objective as the minimization of ΔE_{RM} ($Min(\Delta E_{RM})$).

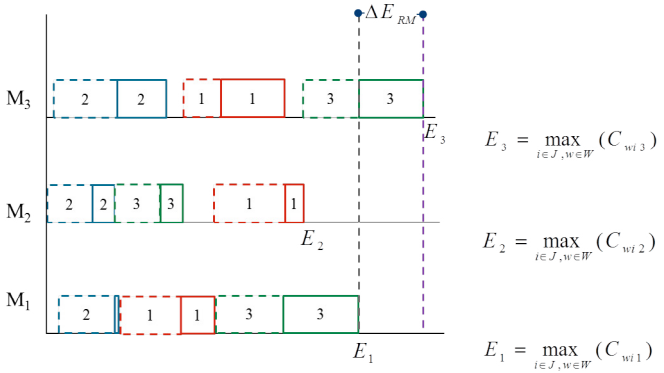

Fig. 4. OF3: Machine's workload balancing

4. SOLUTION METHODS

4.1 NSGA - II

Many multi-objective evolutionary algorithms have been exposed in the last years. The NSGA - II is composed of two principal parts: a fast non-dominated sorting solution part and the preservation of the solution's diversity. Based on these criterion, we are exposing the used process in our method.

Initial population The initial population is generated at random according to the number of defined chromosomes (POP). Then, the resource assignment (workers and tools) is made randomly taking into account the required skills to execute one activity and the mastered skills by each worker as well as the required tools par each operation and their availability.

Population sorting: Non-dominated solutions This step follows the same procedure used by Deb et al. (2002), the crowding distance is also applied.

Selection The selection is made by the tournament method, where the confrontation of two individuals of the initial population (chose at random) is made to stock the individuals with the best Pareto's front. If the two confronted members are of the same front, the decision is making according to the crowding distance criteria.

Crossover The crossover is described by algorithm 1.

Mutation We generate two cut-points selected at random in one parent, so the chromosome is divided into three sub-vectors named sv#1, sv#2 and sv#3. Then, we exchange sv#1 and sv#3.

Population update After applying the genetic operators and the population sorting criterion (fronts and crowding distance), we must determine the individuals that will be kept for the next generation. We begin with an initial population (P_t), we create the children (Q_t) in order to have a final population (R_t) that will be classified in terms

Algorithm 1 Crossover

1: **Step 1 :** Apply the mutation to every selected parent according to a mutation probability (before applying the crossover, some perturbations will generate in the parents to increase the diversity of the created individuals).
2: **Step 2 :** one cut point is selecting at random and two subdivisions for each parent are generated (it is composed by three vectors: the operation's sequence, the tool's allocation and the worker's allocation).
3: **Step 3 :** the subdivisions of the two parents are exchanged (we made the exchange of positions keeping the worker's and tool's allocation that have been already done).
4: **Step 4 :** the chromosome's correction process to avoid the repetition of any element is done.

of fronts and crowding distance to keep a constant number of individuals (POP: population size).

4.2 NSGA - III

We expose the NSGA - III that is designed to face up with many objectives at the same time (more than two). This algorithm was proposed by Deb and Jain (2014) changing some selection mechanisms. The NSGA - III is based on the steps described in algorithm 2:

Algorithm 2 NSGA - III

1: Calculate the number of reference points (H) to place on the hyper-plan
2: Generate the initial population at random taking into account the resources assignment constraints (POP chromosomes)
3: Realize the non-dominated population sorting
4: **for** $i = 1$ *Stopping criteria* **do**
5: Select two parents $P1$ and $P2$ using the tournament method
6: Apply the crossover between $P1$ and $P2$ with a probability P_c
7: Realize the non-dominated population sorting
8: Normalize the population members
9: Associate the population member with the reference points
10: Apply the niche preservation (counter)
11: Keep the niche obtained solutions for the next generation
12: **end for**

Determination of reference points on a hyper-plane We must define a set of reference points to ensure the diversity of the obtained solutions. Different points are placed on a normalized hyper-plan that have the same orientation in the all the axis. The number of reference points (H) is defined by:

$$H = \binom{C+g-1}{g} \tag{4}$$

Where C is the number of objective function, g is the number of divisions to consider on every objective axis (for 3 objectives and 4 divisions, we will have 15 reference points). The reference points are place on the hyper-plan and the solutions will be described by a Pareto front, then the solutions will be associated with the created reference points.

Normalization of the population members We must determine an ideal point of the currently population, so we must identify the minimal value of each objective function $(OF_i^{min}, i = 1, 2, ..., C)$. Then, each objective function will be moved subtracting $z_i^{min} = (OF_1^{min}, OF_2^{min}, ..., OF_C^{min})$ to the objective f_i. We continue with the steps proposed by Deb and Jain (2014) in order to generate a hyper-plan (solving the Pareto fronts where the objective solutions are different scale).

Association among reference points and solutions After normalizing each objective function, it is necessary to associate each population member with a reference. We define a reference line for each point joining the reference point with the origin point (5). Then, we determine the perpendicular distance among each population member and each reference line. Finally, the reference point that has the closest reference line from a population individual is associated to this population member.

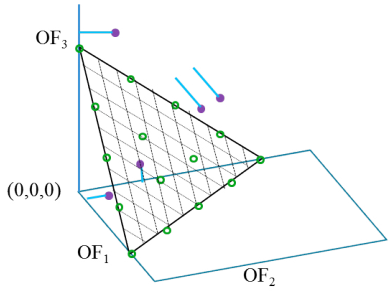

Fig. 5. Association of solutions with reference points

Niche preservation operation A reference point can be associated to one or more solution members, but we must keep the solution that is closer of the point (perpendicular distance from the reference line, Deb and Jain (2014))

Genetic operators The children generation has been made applying the genetic operators used in the NSGA-II algorithm. According to the suggestions of Deb and Jain (2014), we have fixed a population size (POP) close to the number of reference points (H) to give the same importance to all the population members.

5. COMPUTATIONAL EXPERIMENTATION

In this section, we expose the results of the numerical application of the described multi-objective methods. The tests were run using C language in codeblocks 12.11, a personal computer with CPU CORE i3 2.4 GHz, 4GB RAM memory and the operating system windows 7.

5.1 Generating numerical instances

Different manufacturing environments were randomly generated using the same criterion presented in Campos-Ciro et al. (2015a).

5.2 Testing set of problems

We present the results obtained after comparing the two proposed methods, running the small sized and medium-large instances.

Small sized problems Every scenario follows the next notation: $N \times M\_\omega - \lambda - N_W - N_S - N_K$. In addition, nine test problems were generated randomly for each scenario (ran 10 times). We have 117 scenarios (1053 tested instances - 9 problems of each scenario type and every scenatio has 9 possible parameter's combinations of ω and λ values). We present 13 set of instances that arrange the values of the parameters ω et λ after the different combinations $(N \times M - N_W - N_S - N_K)$.

The parameters of the algorithms were defined: crossover probability to 0.8, mutation probability to 0.2. Both methods were run for a maximum evaluation of 23000 objective functions (number of generation to 250 and the population size to 92 as was suggested by Deb and Jain (2014)).

Talking about the NSGA-III, H is not a parameter of the algorithm because the population size depends on H, so $POP \simeq H$. Following the suggestions of the MOEA/D methods developers, for a three objectives problem $H = 91$, so $g = 12$ (number of divisions over each objective).

Table 1 exposes the comparison of both algorithms, the best hyper-volume (H_{max}), the average value ($\overline{H}$) and the worst value (H_{min}) obtained in the different scenarios. The NSGA-III shows the best results. However, the average values between one algorithm and the other are close for the tested instances. Figures 6, 7 and 8 represent a graphical example where there are a reference point (the worst solution in the objective space), the best front of the NSGA-II (16 solutions - asterisks) and the best front of the NSGA-III (18 solutions - crosses).

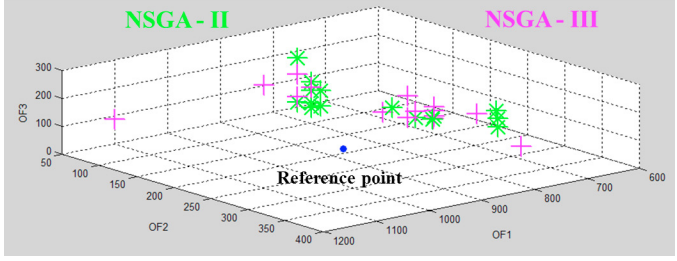

Fig. 6. Scenario graphical representation $3 \times 3 - 3 - 3 - 3$

Medium-large sized problems In the same way, nine test problems were generated randomly for each manufacturing scenario (ran 10 times). The parameters remain as they were described for the small instances.

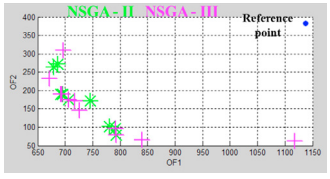

Fig. 7. Scenario graphical representation $3 \times 3 - 3 - 3 - 3$

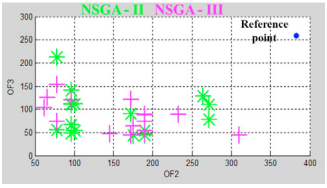

Fig. 8. Scenario graphical representation $3 \times 3 - 3 - 3 - 3$

Table 2 presents the comparisons of both algorithms, where the NSGA-III presents the best results, it means that this algorithm shows an important progression of the region that is cover by the solutions that are included in the best Pareto front, departing from a reference point (anti-optimal solution). Besides that, we can verify that for the instances with less of 150 operations ($N \times M$), the distance between the average hyper-volume values of both algorithms is small, but every time that we increase the number of operations, the NSGA-III performs better than the NSGA-II for solving our open shop scheduling problem.

6. CONCLUSION

An open shop scheduling with resource constraints (multi-skill assignment) and the minimization of three objective functions has been studied $(O_m|r_i|\sum F_i, \Delta E_{RH}, \Delta E_{RM})$. The proposed problem exposes the minimization of the total flow time and the resources workload balancing (humans and machines) in a simultaneous way. Two multi-objective evolutionary algorithms were adapted and described to solve this problem: the NSGA-II and the NSGA-III. Different test were done and the results have shown that for the small-sized instances, both algorithms have a similar performance but the NSGA-III presents a slight higher hyper-volume average than the NSGA-II.

The large-sized instances results showed that the NSGA-III has a better performance than the NSGA-II if we increase the size of the instance problem. According to the results, the NSGA-III is better adapted to solve problems with more than 2 objectives due to the process of the definition of a hyper-plan and some reference points to classify the solutions and select the best qualified for the next population.

These two methods were compared and analyzed using the hyper-volume metric that calculates the progression of the optimal Pareto front from a reference point (anti-optimal). As future works, we would like to propose a local search or a dynamic parameter tunning using fuzzy logic controls to improve the results and also propose a hybridization of the NSGA-II and the NSGA-III with the Lorenz dominance criterion.

REFERENCES

Abido, M.A. (2003). A niched pareto genetic algorihtm for multiobjective environmental/economic dispatch. *Electrical Power and Energy Systems*, 25, 97 – 105.

Campos-Ciro, G., Dugardin, F., Yalaoui, F., and Kelly, R. (2015a). A fuzzy ant colony optimization approach for solving an open shop scheduling problem with multi-skills resource and tool constraints. In *Metaheuristics International Conference (MIC), Agadir, Morocco.*

Campos-Ciro, G., Dugardin, F., Yalaoui, F., and Kelly, R. (2015b). A fuzzy ant colony optimization to solve an open shop scheduling problem with multi-skills resource constraints. In *Information Control Problems in Manufacturing (INCOM), IFAC-PapersOnLine, Volume 48, Issue 3, 2015, Pages 715-720.*

Coello, C.A.C., Aguirre, A., and Zitzler, E. (2007a). Evolutionary multi-objective optimization. *European Journal of Operational Research*, 181 (16), 1617 – 1619.

Table 1. The best, the average and the worst hyper-volume values obtained by the NSGA-II and the NSGA-III - small size

Problem	NSGA - II			NSGA - III		
	H_{max}	$\overline{H}$	H_{min}	H_{max}	$\overline{H}$	H_{min}
$3 \times 3 - 3 - 3 - 3$	0,8715	0,8473	0,8390	**0,8890**	**0,8562**	0,8318
$3 \times 3 - 3 - 5 - 3$	0,8510	**0,8383**	0,8201	**0,8601**	0,8361	0,8294
$4 \times 2 - 2 - 5 - 5$	0,8405	0,8207	0,8060	**0,8526**	**0,8299**	0,8010
$4 \times 3 - 3 - 3 - 8$	**0,8497**	0,8234	0,8001	0,8412	**0,8287**	0,8090
$4 \times 3 - 3 - 5 - 7$	0,8599	0,8364	0,8124	**0,8680**	**0,8402**	0,8099
$4 \times 4 - 4 - 3 - 10$	**0,8419**	0,8161	0,7835	0,8402	**0,8197**	0,7869
$4 \times 4 - 4 - 5 - 12$	0,8314	0,8031	0,7721	**0,8406**	**0,8088**	0,7775
$5 \times 4 - 4 - 3 - 18$	0,8411	0,8125	0,7699	**0,8608**	**0,8275**	0,7768
$5 \times 4 - 4 - 5 - 15$	**0,8407**	0,8064	0,7703	0,8387	**0,8084**	0,7761
$5 \times 4 - 4 - 7 - 11$	0,8496	0,8199	0,7715	**0,8537**	**0,8206**	0,7788
$7 \times 3 - 3 - 3 - 9$	0,8290	0,8015	0,7819	**0,8318**	**0,8119**	0,7836
$7 \times 3 - 3 - 5 - 17$	0,8315	0,8118	0,7807	**0,8402**	**0,8192**	0,7835
$7 \times 3 - 3 - 7 - 21$	0,8297	0,8009	0,7814	**0,8324**	**0,8083**	0,7890

Table 2. The best, the average and the worst hyper-volume values obtained by the NSGA-II and the NSGA-III - medium-large size

Problem	NSGA - II			NSGA - III		
	H_{max}	$\overline{H}$	H_{min}	H_{max}	$\overline{H}$	H_{min}
$10 \times 5\_50 - 0.1 - 5 - 3 - 15$	0,7798	0,7602	0,7518	**0,7901**	**0,7790**	0,7598
$10 \times 5\_75 - 0.3 - 5 - 5 - 35$	0,7715	**0,7528**	0,7362	**0,7732**	0,7499	0,7297
$10 \times 5\_100 - 0.5 - 5 - 3 - 42$	**0,7854**	0,7631	0,7401	0,7807	**0,7687**	0,7482
$10 \times 10\_50 - 0.1 - 10 - 5 - 25$	0,7801	0,7523	0,7388	**0,7922**	**0,7618**	0,7435
$10 \times 10\_75 - 0.5 - 10 - 3 - 70$	**0,7810**	**0,7684**	0,7515	0,7790	0,7596	0,7468
$10 \times 10\_100 - 0.1 - 10 - 7 - 55$	0,7935	**0,7732**	0,7586	**0,7986**	0,7681	0,7422
$30 \times 5\_50 - 0.1 - 5 - 3 - 100$	0,7627	0,7438	0,7209	**0,7893**	**0,7691**	0,7438
$30 \times 5\_75 - 0.5 - 5 - 5 - 123$	0,7835	0,7511	0,7291	**0,7950**	**0,7780**	0,7507
$30 \times 5\_100 - 0.1 - 5 - 7 - 19$	0,7854	0,7609	0,7492	**0,8132**	**0,7834**	0,7480
$30 \times 15\_50 - 0.5 - 15 - 3 - 62$	0,7215	0,6967	0,6699	**0,8001**	**0,7810**	0,7690
$30 \times 15\_75 - 0.1 - 15 - 5 - 99$	0,7402	0,7115	0,6898	**0,7865**	**0,7790**	0,7566
$30 \times 15\_100 - 0.3 - 15 - 3 - 160$	0,7317	0,7018	0,6797	**0,7821**	**0,7609**	0,7410
$50 \times 5\_50 - 0.1 - 5 - 3 - 182$	0,7932	0,7568	0,7225	**0,8199**	**0,7810**	0,7583
$50 \times 5\_75 - 0.5 - 5 - 5 - 45$	0,7819	0,7458	0,7167	**0,8102**	**0,7794**	0,7439
$50 \times 5\_100 - 0.1 - 5 - 7 - 138$	0,7928	0,7628	0,7428	**0,8367**	**0,7836**	0,7418
$50 \times 10\_50 - 0.1 - 10 - 3 - 200$	0,7198	0,6874	0,6508	**0,7987**	**0,7225**	0,6815
$50 \times 10\_75 - 0.5 - 10 - 7 - 120$	0,7030	0,6691	0,6131	**0,7624**	**0,7137**	0,6718
$50 \times 10\_100 - 0.1 - 10 - 5 - 317$	0,7152	0,6598	0,6027	**0,8049**	**0,7312**	0,6816

Coello, C.A.C., Lamont, G.B., and Veldhuizen, D.A.V. (2007b). *Evolutionary algorithms for solving multi-objective problems*. Springer.

Collette, Y. and Siarry, P. (2002). *Optimisation multi-objectif*. EYROLLES.

Deb, K. and Jain, H. (2014). An evolutionary many-objective optimiszation algorithm using reference-point based non-dominated sorting approach, part I: solving problems with box constraints. *Evolutionary Computation, IEEE Transactions on*, 18 (4), 577 – 601.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast elitist non-dominated sorting genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6 (2), 182 – 197.

Fonseca, C.M. and Fleming, P.J. (1993). Genetic algorithm for multiobjective optimization: formulation, discussion and generalization. In *Proceedings of the fifth International Conference on Genetic Algorithms. San Mateo: Morgan Kauffman Publishers. Pages 416 - 423.*

Huband, S., Hingston, P., While, L., and Balone, L. (2003). An evolution strategy with probabilistic mutation for multi-objective optimisation. In *The Congress on Evolutionary Computation, 2003. CEC '03*, 2284–2291 Vol.4.

Indraneel, D. and Dennis, J. (1997). A closer look at drawbacks of minimizing weighted sums of oobjective for pareto set generation in multicriteria optimization problems. *Structural Optimization*, 14, 63 – 69.

Ishibuchi, H., Narukawa, K., Tsukamoto, N., and Nojima, Y. (2008). An empirical study on similarity-based mating for evolutionary multi-objective combinatorial optimization. *European Journal of Operational Research*, 188, 57 – 75.

Laumanns, M., Zitzler, E., and Thiele, L. (2000). A unified model for multi-objectiv evolutionary algorithms with elitism. In *Congress on evolutionary Computationn Oiscataway, New Jersey, pages 46 - 53.*

Ouazene, Y., Yalaoui, F., Chehade, H., and Yalaoui, A. (2014). Analysis of different criteria for work balancing on identical parallel machines. In *Proceedings of the International Conference on Industrial Engineering and Operations Management, Bali, Indonesia.*

Purshouse, R. (2003). *On the evolutionary optimization of many objectives*. Ph.D. thesis, Univ. Sheffield, Sheffield, U.K.

Srinivas, N. and Deb, K. (1994). Multiobjective function optimization using non dominated sorting genetic algorithms. *Evolutionary Computation*, 2 (3), 221 – 248.

Wang, X.J., Zhang, C.Y., Gao, L., and Li, P.G. (2008). A survey and future trend of study on multi-objective scheduling. In *Natural Computation, 2008. ICNC '08. Fourth International Conference on*, volume 6, 382–391.

While, L., Hingston, P., Barone, L., and Huband, S. (2006). A faster algorithm for calculating hypervolume. *Evolutionary Computation, IEEE Transactions on*, 10, 29–38.