A General Local Search Pareto Approach with Regular Criteria for Solving the Job-Shop Scheduling Problem Multi-resource Resource Flexibility with Linear Routes

Andrés Alberto García-León(✉)
and William Fernando Torres Tapia

Facultad de Ingeniería Programa de Ingeniería Industrial, Universidad de Ibagué, Ibagué, Colombia
andres.garcia@unibague.edu.co,
williamindustrial2014@gmail.com

Abstract. The Job-shop Scheduling problem Multi-resource Resource flexibility with linear routes is an extension of the classical Job-shop Scheduling problem "JSMRFLR" where an operation needs several resources (machines) simultaneously to be processed and each machine is selected from a given set. Linear route indicates that an operation is exclusively performed for a job. Publications related to this problem are really scarce and they are dedicated to minimize makespan criterion, which aims to minimize the use of the machines. In the modern scenery of the Operations Management, the exclusive minimization of the makespan does not allow to analyze aspects of the customer service to ensure high levels of competitiveness such as the consideration of due-date and the importance between jobs. In this paper, we propose a general Pareto approach to solve the JSMRFLR with regular criteria, which operates by an efficient local search at using a fast estimation function for the criteria considering the conjunctive graph. During the search, the set of non-dominated solutions is updated. The efficiency of our approach is illustrated on instances of literature at performing three sets of criteria. The first set considers makespan and maximum tardiness. In the second total flow time is added and in the third the total tardiness. As a product of our approach, a reference of results is proposed by future research.

Keywords: Scheduling theory · Extension of the Job-shop scheduling problem · Regular criteria · Local search · Pareto optimization

1 Introduction

For the Operations Management, the success of the production programming depends on the scheduling decisions. These decisions lead to determine the sequence of activities and the assignment of resources to optimize an objective function. In the literature of scheduling, the highest rigor of the decisions have been focused on configurations derived of the classical flow-shop and Job-shop problems such as parallel machines, sequence dependent set up times, maintenance activities and others. Concerning to the minimization of criteria, makespan, that is the maximum completion time of jobs has been the most studied. Minimizing makespan does not consider relevant aspects for the customer service like the due-date of jobs that represents the delivery date agreed with customers and the importance between them. Therefore, minimizing regular criteria different than the makespan leads to identify key aspects to improve the customer service.

One of the extensions of the Job-shop problem is the Multi-resource and Resource flexibility with linear routes. This extension was proposed by [1]. The Multi-resource considers that an operation needs several resources simultaneously to be performed and resource flexibility determines that a resource (machine) is chosen from a given set. Linear route means that each operation is performed exclusively for a job. To best of our knowledge, it is scarce the level of publications at solving this problem even considering makespan.

This paper is oriented to improve the level of customer service and the productivity in a more realistic industrial environment such as the Job-shop Scheduling Problem Multi-resource Resource Flexibility with linear routes "JSMRFLR". To reach the goal, a fast local search algorithm that uses the properties of a conjunctive graph adapted to regular criteria is designed to solve the JSMRFLR in Pareto manner at minimizing a combination of regular criteria. The solutions that belong to the Pareto front are gotten iteratively. In each iteration, a random criterion is selected and the move is determined by an estimation function that considering the evaluation simultaneous of reversing a critical arc (x, y) that belongs to the critical path of a job that affects a criterion and the reassignment of x and y at maintaining the level of operations into the conjunctive graph (maximum number of arcs from the start dummy node) between operations j and k.

During the search, the set of non-dominated solutions is updated at removing the dominated solutions and adding of a new one if it is necessary. The proposed approach is validated in instances of the literature proposed by [1] at solving three different sets of criteria. In the first set, makespan and maximum tardiness are solved. The second set adds the total flow time, and the third adds the total tardiness. As a contribution of our experiments a benchmark of results is proposed for future research. The paper is detailed as follows. Section 2 describes and models the problem with the considerations of the front of Pareto. In Sect. 3, the algorithm that solves the problem is described. Finally, Sect. 4 illustrates some computational experiments.

2 The Problem and Pareto Optimization

The objective of this Section is to illustrate the problem with its components and the guidelines to obtain the Pareto front at minimizing a set of regular criteria.

2.1 Problem Description and Modeling

To describe the problem, it is necessary to consider the Job-shop Scheduling Problem "JSP". In the JSP, a set of n jobs $J = \{J_1, \ldots .J_n\}$ are processed on a set $M =$

$\{M_1, \ldots .M_m\}$ of m machines that are available. Each machine can only process one job at a time. A job J_i is obtained at performing sequentially n_i operations in a fix route. Preempting operations is not allowed. It means that an operation cannot be interrupted once started. Each job J_i has a due-date d_i. The JSP Multi-resource Resource Flexibility with linear routes "JSMRFLR" considers that an operation needs several resources simultaneously to be performed and resource flexibility indicates that a resource may be selected in a given set.

A model of the disjunctive graph for the Multi-resource shop scheduling with resource flexibility considering linear and non-linear routes is proposed in [1]. This model is used for minimizing makespan. To minimize regular criteria in the JSMRFLR, we have added the properties of linear routes from the JSP, which were validated in [2] and adapted for the flexible Job-shop scheduling problem in [3]. In the conjunctive graph for minimizing regular criteria for the JSMRFLR, the n nodes ϕ_i that represents the finalization of the jobs, the concept of maximum number of arcs between node 0 and an operation x (l_x) and the tail from a node x to the dummy node $\phi_i(q_x^i)$ have been added.

In the JSMRFLR, the disjunctive graph is noted $G = (V, A, E)$, where V is the set of nodes, A is the set of conjunctive arcs and E is the set of disjunctive arcs. The nodes in E represent operations of jobs (set O), plus a dummy node 0 that corresponds to the start of each job, and n dummy nodes ϕ_i. A contains conjunctive arcs which connect two consecutive operations on the routing of jobs, the node 0 to every first operation of each job, and the last operation of J_i to a dummy node ϕ_i. The set E contains disjunctive arcs between every pair of operations assigned to the same resource. Let E_k be the set of disjunctive arcs between pairs of operations that must be processed on k, $E = \cup E_k$.

The set of operations O has to be processed on a set of machines (resources) M. To be processed an operation $x \in O$ requires $R(x)$ resources simultaneously and M_x^k is the resource subset in which the k^{th} resource $(1 \leq k \leq R(x))$ must be selected. The M_x^k subsets are not necessarily disjoint, for example, a resource could belong to several subsets. To obtain a feasible schedule in the JSMRFLR, assignment and sequencing decisions have to be made to solve the conflict in E. In each operation, the assignment decision consists on determining which machine or resource must be selected from each subset to perform it. However, it is mandatory to ensure of not assigning a resource twice or more to the same operation, additionally the processing time of an operation x (p_x) is determined by the maximum time of the resource where it is assigned. The sequencing decision deals with determining a sequence of operations on each selected resource k to minimize any regular criterion. It is important to highlight that the sequence of operations on resources does not lead to create cycles in the conjunctive graph. As soon as a solution is obtained, it is possible to extract information to identify the properties to create moves which lead to improve any criterion. The arc from 0 to the first operation of J_i has a length equal to the release date r_i of j_i. Any remaining arc has a length equal to the processing time of the operation from which it starts. The starting time of x, $h_x = L(0, x)$ called head, that corresponds to the length of the longest path from 0 to x. The tail from a node x to the dummy node $\phi_i(q_x^i)$ is equal to $[L(x, \phi_i) - p_x]$ if a path exists from x to ϕ_i and $-\infty$ otherwise. A path from 0 to ϕ_i is called critical if its length is equal to $\mathrm{C_i}$, and every node x belonging to this critical path is critical according to job $\mathrm{J_i}$. A critical node x for job $\mathrm{J_i}$ satisfies $h_x + p_x + q_x^i = \mathrm{C_i}$. The level l_x of a node x in G denotes the maximum number of arcs in a path from 0 to x. After obtaining the heads of the nodes of the graph, the criterion of a feasible schedule represented by the selection can be determined in $O(n)$ from the starting times of the dummy nodes. For instance, the makespan is obtained using the formula $C_{max} = \max C_i$. Additionally the tardiness (T_i), $T_i = \max(0, C_i - d_i)$. Then, the maximum tardiness is $T_{max} = \max T_i$, total tardiness (T) is the sum of the tardiness of jobs $(\sum_1^n T_i)$ and the total flow time is the sum of the completion time of jobs $(\sum_1^n C_i)$.

2.2 Pareto Optimization

Pareto optimization aims to find a set of non-dominated solutions at optimizing a set of criteria (often conflicting). In our case, for optimizing regular criteria, all objectives lead to be minimized. The set of non-dominated solutions Q represents the Pareto front and its comprehension requires to check the dominance between solutions, and the conditions that a solution s must satisfy to be included in Q. Concerning to the dominance between solutions, solutions A and B are considered. To determine if A dominates to B or B is dominated by A, the following conditions must be true: (1) A is not worse than B for all objectives, and (2) A is strictly better than B for at least one objective. To know if a solution s must be included in Q, two conditions must be ensured: (1) Any two solutions of Q must be non-dominated with respect to each other and, (2) Any solution not in Q is dominated by at least one solution in Q.

The Fig. 1 helps to illustrate two non-dominated solutions of an instance with three jobs, seven operations and six machines or resources (M_1, M_2, M_3, M_4, M_5 and M_6) at minimizing makespan and maximum tardiness simultaneously. Besides, the due dates for jobs are determined. They are $d_1 = 6$, $d_2 = 12$ and $d_3 = 6$ (see column d_i). The solution in Fig. 1(a) aims to minimize makespan, and in the Fig. 1(b) the maximum tardiness. The solution in the Fig. 1(a) can be used to explain some properties of the conjunctive graph and some remarks can be extracted. The first job has two operations (see the first route for a job). The first operation of the first job requires two resources simultaneously (see dashed rectangle). The first resource is selected between M_1 and M_4. If M_1 is selected 3 is the processing time, otherwise 2. The second resource is selected between M_2 and M_4. In this example, M_4 and M_2 are assigned respectively (they are highlighted in yellow and blue color) and the processing time is 2. Note that a resource was not assigned twice. Considering the same operation, it starts at time 0 and finishing at time 2 (see 0/2), since the processing time is 2 (see 0 + 2 = 2). Considering that one operation cannot start before finishing all its predecessors, the completion time of all jobs (see column C_i) and the tardiness of all jobs are calculated (see column T_i).

The solutions illustrated in the Fig. 1 are non-dominated solutions. If for example makespan and maximum tardiness are considered, in the Fig. 1(a) the values for the criteria are 12 and 4 respectively, and in the Fig. 1(b) these values are 14 and 2. It means that for a decision maker without considering importance between criteria both solutions are equivalent and they satisfy the conditions of non-dominance between solutions.

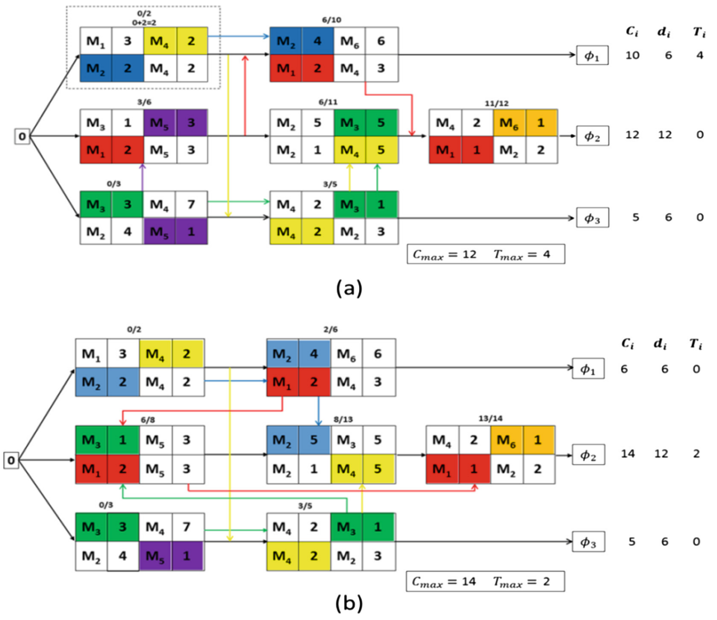
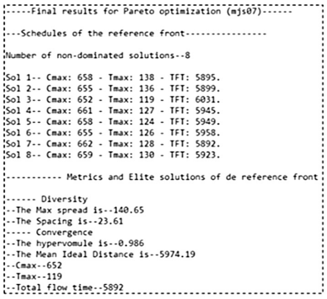


Fig. 1. Examples of two non-dominated solutions with makespan and maximum tardiness

2.3 Quality of the Pareto Front

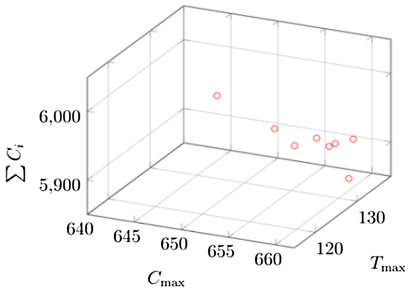
There is not any consensus to establish the quality of the Pareto Front. Different approaches have been proposed (see for example [4] and [5]). However, two aspects are mandatory to consider: convergence and diversity. Convergence implies that the set of non-dominated solutions must be located closest to the origin of coordinates. Diversity is related to the solutions in sparsely space to ensure that the decision maker has several and representative trade-off solutions among conflicting criteria. To determine the quality of the front, the set of metrics employed in [5] are analyzed. In this case, for the convergence the hypervolume (*HV*), elite solutions and the Mean Ideal Distance (*MID*) are measured. *HV* is the volume covered by the solutions of the front and respect to the worst solution, since regular criteria are minimized. Elite solutions are the best values of the criteria. *MID* is the average Euclidean distance obtained between each non-dominated solution and the origin. Concerning to the diversity, the maximum spread (*D*) and spacing (*SP*) are analyzed. *D* is the longest diagonal of the hyper box formed by the extreme values of the criteria, and *SP* is the average distance between consecutive solutions.

The Fig. 2 is an image of the results generated by our algorithm, which will be explained in the next section at minimizing in Pareto manner makespan, maximum tardiness and total flow time (TFT) at performing the instance *mjs07* during five minutes from [1]. This Figure is divided in two parts. The part a represents a txt file with the results. For example, the algorithm gets eight non-dominated solutions. Each solution shows the value of the criteria. Additionally, the metrics are separated by diversity and convergence. In the part b, the eight non-dominated solutions are plotted on a three dimensional plane.



```
-----Final results for Pareto optimization (mjs07)------

---Schedules of the reference front---------------

Number of non-dominated solutions--8

Sol 1-- Cmax: 658 - Tmax: 138 - TFT: 5895.
Sol 2-- Cmax: 655 - Tmax: 136 - TFT: 5899.
Sol 3-- Cmax: 652 - Tmax: 119 - TFT: 6031.
Sol 4-- Cmax: 661 - Tmax: 127 - TFT: 5945.
Sol 5-- Cmax: 658 - Tmax: 124 - TFT: 5949.
Sol 6-- Cmax: 655 - Tmax: 126 - TFT: 5958.
Sol 7-- Cmax: 662 - Tmax: 128 - TFT: 5892.
Sol 8-- Cmax: 659 - Tmax: 130 - TFT: 5923.

---------- Metrics and Elite solutions of de reference front

------ Diversity
--The Max spread is--140.65
--The Spacing is--23.61
----- Convergence
--The hypervomule is--0.986
--The Mean Ideal Distance is--5974.19
--Cmax--652
--Tmax--119
--Total flow time--5892
```

(a)



(b)

Fig. 2. Results for the instance *mjs07* at minimizing C_{max}, T_{max} and $\sum C_i$

3 Description of the Algorithm

In Sect. 2 were defined the components of the problem JSMRFLR and the guidelines of Pareto optimization, which are used to describe the proposed algorithm. Basically, our algorithm is a local search process that uses the properties of the conjunctive graph, a test of feasibility conditions, an estimation function and a procedure to update the set of non-dominated solutions.

3.1 Feasibility Conditions and Estimation Functions

Feasibility conditions aim to check if a critical operation could be moved without performing any transformation in the conjunctive graph. For this problem, two types of moves are considered: reversing critical arc (x, y) and the reassignment of a critical operation performed on resource L to resource L' between j and k. Respect to reverse critical arcs, the conditions validated in [2] for minimizing regular criteria have been extended to this problem. However, it is important to clarify that if several arcs connect x and y, all arcs must be reversed, since they are critical, and x and y must belong to different jobs.

In the reassignment, moving an operation ε ($\varepsilon = x$ or $\varepsilon = y$) between operations j and k from resource L to L', it is necessary to check that there are no paths from $b(\forall b \in B)$ to j, and simultaneously from k to $c(\forall c \in C)$. Here, B (resp. C) is the set formed for all immediate successors (resp. predecessors) of ε without considering the successor (resp. predecessor) linked to the resource L. To ensure it, two expressions considering the number of arcs must be satisfied: (1) $l_j \leq \min_{b \in B}\{l_b\}$ and (2) $l_k \geq \max_{c \in C}\{l_c\}$. These expressions are validated, since there is not possible to get a path from a node u to v if $l_v \leq l_u$. To estimate the value of any regular criterion, the completion time of a job must be estimated. It means to determine the completion time of each node ϕ_i. To determine it, two sets of functions based on the type of move.

Concerning to swap a critical arc, the three expressions to L_1, L_2 and L_3 from [2] have been extended to the JSMRFLR problem. Those expressions aim to maintain the same values for the heads or start time for operations and the tails to nodes ϕ_i at reversing a critical arc. At reassigning an operation ε between j and k, we want to maintain the same conditions (heads and tails) to reach a better precision of the estimation. In this case the condition for C_1, C_2 from [1] and L_3 from [6] have been adapted at considering regular criteria.

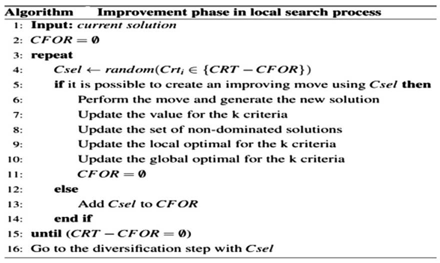
3.2 Steps of the Search

The search starts with an initial solution, which is the first solution added to the set of non-dominated solutions. Then, iteratively two steps (improvement and diversification) according to the conditions of the graph are executed. Simultaneously the set of non-dominated solutions is updated. In the initial solution, the jobs are sorted in increasing way according to their due-date. Then, for the assignment and sequence of operations on machines (resources) is solved at considering the operation in route of the order jobs. It means that, if there exist and ordering of jobs, the assignment and sequence decisions solve the first operation of all them, then, the second and so on. The objective of the assignment and sequence decisions aim to obtain the lowest completion time per operation at evaluating the capacity of machines. The initial solution is general and it does not make any distinction between criteria.

To transform the conjunctive graph, all critical arcs that belong to the critical paths of jobs that affect the criterion are referenced to create a move at selecting the best neighbor. The algorithm aims to optimize a Pareto approach and iteratively a random criterion is selected to create a move. Selecting the best neighbor implies determine the feasibility of all moves, and then on each feasible move the evaluation of the estimation function which were described in Sect. 3.1. To clarify the functioning of the search, let $\boldsymbol{CRT}$ be a set of $\boldsymbol{k}$ criteria to minimize, $Crt_i \in \boldsymbol{CRT}$ a criterion $\boldsymbol{i}$ to minimize; besides, let $\boldsymbol{C_{Sel}}$ be a random chosen criterion to optimize from $\boldsymbol{CRT}$. Let $CFOR \subset CRT$ be a subset of $\boldsymbol{CRT}$ that contains the forbidden criteria, which means that in case of a random selection of a criterion, a forbidden criterion cannot be selected. A criterion becomes forbidden when, in the improvement step, it is selected to create a move, and it cannot be generated.

The Fig. 3 illustrates the algorithm for the improvement step. In each iteration, a random criterion $\boldsymbol{C_{Sel}}$ is selected to create a move. If it is not possible to create a move using $\boldsymbol{C_{Sel}}$, $\boldsymbol{C_{Sel}}$ becomes forbidden and it is added to $\boldsymbol{CFOR}$. The search selects a new criterion that belongs to CRT-CFOR. However, when a move is performed, all criteria are authorized to be selected or the set $\boldsymbol{CFOR}$ is emptied. If it is not possible to create a move, the search goes to the diversification step considering the neighborhood of the last forbidden criterion. In this step, when a move is performed the criteria with its local and global optimal are updated.

When, it is not possible to create a move considering the random criterion, a new criterion is selected until the move can be performed. If an improvement move cannot be created, the diversification step starts. It consists on performing iteratively b random moves (b $\in$ [2, 5]) considering the last selected criterion. After that, the search returns to the improvement phase. The set of non-dominated solutions is updated as follows.



```
Algorithm   Improvement phase in local search process
Input: current solution
$CFOR = \emptyset$
repeat
    $Csel \leftarrow random(Crt_i \in \{CRT - CFOR\})$
    if it is possible to create an improving move using $Csel$ then
        Perform the move and generate the new solution
        Update the value for the k criteria
        Update the set of non-dominated solutions
        Update the local optimal for the k criteria
        Update the global optimal for the k criteria
        $CFOR = \emptyset$
    else
        Add $Csel$ to $CFOR$
    end if
until ($CRT - CFOR = \emptyset$)
Go to the diversification step with $Csel$
```

Fig. 3. Algorithm for the improvement step

When a solution s is gotten, it is checked to determine its addition in Q at applying the conditions described in Sect. 2.2. If s were added in Q, s could dominate other solutions, which must be removed from Q. The set of non-dominated solutions is updated at validating simultaneously the addition of the new solution and the elimination of the dominated solutions.

4 Computational Experiments

To validate and evaluate the efficiency of our approach, the instances with linear routing studied in [1] have been considered. Our algorithm was developed in Java language. The experiments were conducted on a PC with 3.40 GHz and 8 GB RAM for each set of criteria on each instance during ten minutes. To calculate the maximum tardiness and the total tardiness, it is necessary to determine the due-date d_i of the jobs. They were generated by introducing a parameter f equals to 1.1, which is inspired from [7]. d_i is determined by multiplying the sum of the average processing times of operations belong to J_i by f. To determine the average processing time of an operation, it is calculated at considering the highest processing time per subset of resources. It is important to mention that d_i was rounded to the next integer number.

The results are presented in three steps. In the first step the efficiency of makespan criterion is evaluated at considering the best known values determined in [1]. In the second step, the elite solutions are analyzed to determine the best combination of criteria to optimize and finally, with the best combination of criteria the results for the metrics of the front of Pareto are calculated.

4.1 Evaluating Makespan

Makespan criterion is the only benchmark to determine the quality of our results at evaluating its convergence. It means the nearest distance respect to a known value. The Table 1 illustrates the best known values (see column BKV) and the best values at considering the three sets of criteria. Additionally, column $DP(\%)$ determines the percentage difference between the best result and the best known value. Besides, if a better known value is gotten, it is underlined and an asterisk is written in column $DP(\%)$. The best makespan of our experiment are written in **bold**.

Table 1. Results for the best makespan in the three sets

Instance	BKV	*Set A*	*Set B*	*Set C*	*DP*(%)	*Instance*	BKV	*Set A*	*Set B*	*Set C*	*DP*(%)
mjs01	361	**361**	362	362	0.0	*mjs35*	265	**266**	269	269	0.4
mjs02	384	321	376	376	*	*mjs36*	225	**246**	269	257	9.3
mjs03	378	382	**379**	**379**	0.3	*mjs37*	207	**229**	261	242	10.6
mjs04	394	404	**399**	**399**	1.3	*mjs38*	241	**254**	271	262	5.4
mjs05	643	**680**	682	682	5.8	*mjs39*	210	**231**	262	246	10.0
mjs06	585	556	573	573	*	*mjs40*	241	**241**	264	242	0.0
mjs07	644	**648**	673	659	0.6	*mjs41*	218	248	276	**237**	8.7
mjs08	575	**582**	597	597	1.2	*mjs42*	250	**250**	282	257	0.0
mjs09	568	**572**	578	578	0.7	*mjs43*	219	**228**	267	242	4.1
mjs10	928	**932**	965	965	0.4	*mjs44*	258	**268**	324	280	3.9
mjs11	1057	1047	**1040**	**1040**	*	*mjs45*	296	**333**	464	456	12.5
mjs12	859	843	862	862	*	*mjs46*	300	**384**	491	413	28.0
mjs13	827	**884**	902	909	6.9	*mjs47*	333	**377**	490	481	13.2
mjs14	946	1026	**1020**	**1020**	7.8	*mjs48*	327	**368**	469	436	12.5
mjs15	1469	1467	1457	1447	*	*mjs49*	356	**390**	470	466	9.6
mjs16	1312	1380	**1352**	**1352**	3.0	*mjs50*	327	**430**	516	472	31.5
mjs17	1572	1567	1582	1582	*	*mjs51*	373	**452**	536	509	21.2
mjs18	1544	**1571**	1628	1601	1.7	*mjs52*	317	**387**	434	415	22.1
mjs19	1572	1553	1598	1566	*	*mjs53*	353	**419**	546	495	18.7
mjs20	1033	1075	**1068**	**1068**	3.4	*mjs54*	311	**407**	467	454	30.9
mjs21	916	**918**	922	922	0.2	*mjs55*	493	**797**	819	819	61.7
mjs22	924	964	**956**	**956**	3.5	*mjs56*	508	789	873	**715**	40.7
mjs23	957	**969**	990	990	1.3	*mjs57*	500	**788**	973	948	57.6
mjs24	918	893	890	890	*	*mjs58*	530	**802**	884	814	51.3
mjs25	1513	**1541**	1568	1548	1.9	*mjs59*	490	916	1024	**881**	79.8
mjs26	1481	1448	1474	1414	*	*mjs60*	268	**299**	421	395	11.6
mjs27	1566	1649	1693	**1602**	2.3	*mjs61*	303	**332**	452	374	9.6
mjs28	1395	**1530**	1631	1545	9.7	*mjs62*	284	**306**	426	357	7.7
mjs29	1336	**1444**	1502	1455	8.1	*mjs63*	289	**317**	398	344	9.7
mjs30	218	239	247	**231**	6.0	*mjs64*	240	**277**	346	306	15.4
mjs31	218	**250**	281	281	14.7	*mjs65*	381	**508**	622	622	33.3
mjs32	219	**250**	266	255	14.2	*mjs66*	423	**558**	652	652	31.9
mjs33	224	**241**	254	246	7.6	*mjs67*	408	**558**	639	639	36.8
mjs34	213	**229**	238	232	7.5	*mjs68*	400	**554**	754	667	38.5

At observing the Table 1, it is possible to infer that our approach evidences a significant performance at minimizing the makespan combined with other criteria. For example, our approach leads to a better makespan than the best known value in nine of 68 instances (*mjs02, 06, 11, 12, 15, 17, 19, 24* and *26*). Besides, the best known value is reached in three instances: *mjs01*, *mjs40* and *mjs42,* since, the percentage difference is 0.0 (see column *DP*). Additionally, in 16 instances the *DP* is less or equal than 5.0% and in six of them is less than 1.0%. Respect to each set, *SetA* Performs better than *SetB* and *SetC* in 52 instances. *SetC* performs better in 16 instances and *SetB* in eight without a superiority level. Since, in the eight instances *SetB* gets the same performance than *SetC*. As conclusion, at evaluating the convergence of the makespan in the consolidation of the set of non-dominated solutions *SetB* is not efficient and the better combination is the maximum tardiness.

4.2 Evaluating Maximum Tardiness and Total Flow Time

The Table 2 illustrates the elite solutions for both criteria. Again, the best value for the criteria is written in bold. Analyzing the results for maximum tardiness, in 32 instances, optimal solution ($T_{\max} = 0$) is gotten, and in 25 of them is gotten in the three sets for the three sets. Concerning to the performance of the sets, Set A presents the best performance in 29 instances, Set B in 5 and Set C in 15. Respect to the total flow time, only Sets B and C are analyzed. At observing the results, Set C leads to better results, since, it is evidenced in 58 instances. At evaluating the performance of the three sets, it is evident the superiority of the *SetB* when a bi-criteria analysis (Makespan and Tmax) is performed. However, if some criteria are added, the best results are gotten in *SetC*.

Table 2. Results for the best T_{max} and $\sum C_i$ in the three sets

Instance	T_{max}			$\sum C_i$	
	Set A	*Set B*	*Set C*	*Set B*	*Set C*
mjs01	**19**	23	**19**	3132	**3125**
mjs02	57	**53**	54	**3295**	3329
mjs03	**14**	19	17	**3331**	**3331**
mjs04	67	**65**	**65**	3328	**3306**
mjs05	**112**	116	119	6062	**5795**
mjs06	**58**	70	72	5367	**5267**
mjs07	123	136	128	6043	5921
mjs08	**88**	89	89	**5135**	5226
mjs09	**76**	80	80	5065	**4952**
mjs10	**601**	608	**601**	8613	**8437**
mjs11	695	706	**690**	9165	**8936**
mjs12	**482**	498	502	7964	**7674**
mjs13	**530**	550	545	**8144**	8198
mjs14	653	647	**629**	9490	**8939**
mjs15	928	916	**885**	**13447**	13561
mjs16	874	**855**	873	12625	**12825**
mjs17	1082	**1064**	1094	**15017**	15331
mjs18	**1037**	1079	1041	15370	**15091**
mjs19	**1040**	1094	1047	15281	**14461**
mjs20	**702**	705	711	**9351**	9400
mjs21	**547**	**547**	557	**8036**	8121
mjs22	621	614	**609**	8723	**8694**
mjs23	**592**	641	619	8892	**8638**
mjs24	**545**	549	548	8216	**7923**
mjs25	**997**	1033	1015	14776	**14597**
mjs26	940	962	**902**	14485	**13003**
mjs27	1113	1157	**1048**	15887	**14675**
mjs28	998	1103	**958**	14936	**13670**
mjs29	911	969	**902**	14264	**13545**
mjs30	0	0	0	2228	**1980**
mjs31	0	0	0	2473	**2148**
mjs32	0	0	0	2492	**2180**
mjs33	0	0	0	2323	**2173**
mjs34	0	0	0	2087	**1995**
mjs35	0	0	0	2382	**2211**
mjs36	0	0	0	2368	**2146**
mjs37	0	0	0	2291	**1948**
mjs38	0	0	0	2458	**2192**
mjs39	0	0	0	2273	**1984**
mjs40	0	0	0	2194	**1935**
mjs41	0	0	0	2460	**2067**
mjs42	0	0	0	2206	**2009**
mjs43	0	0	0	2011	**1863**
mjs44	0	0	0	2758	**2149**
mjs45	0	0	0	4331	**3765**
mjs46	0	0	0	4606	**3941**
mjs47	0	0	0	4491	**3854**
mjs48	0	0	0	4522	**3508**
mjs49	0	0	0	4274	**3785**
mjs50	**0**	74	26	4975	**4221**
mjs51	**2**	95	50	5032	**4524**
mjs52	**0**	2	**0**	4034	**3539**
mjs53	**0**	103	28	5067	**4518**
mjs54	0	0	0	4304	**3806**
mjs55	**159**	166	181	**7999**	8036
mjs56	116	220	**59**	8494	**6888**
mjs57	**88**	308	242	9391	**8895**
mjs58	**108**	174	121	8294	**7633**
mjs59	240	359	**163**	9872	**7926**
mjs60	0	0	0	3954	**3655**
mjs61	**0**	10	**0**	4000	**3383**
mjs62	0	0	0	3834	**3226**
mjs63	0	0	0	3757	**3207**
mjs64	0	0	0	3208	**2823**
mjs65	0	0	0	6075	**5933**
mjs66	**0**	18	**0**	6358	**5949**
mjs67	**0**	**0**	69	**6124**	6840
mjs68	**0**	60	**0**	7177	**6539**

4.3 Results for Set A and Set C

The main conclusion of the last sub Sections is that the best results are reached for *Sets A* and *C*. In this Section the results of the metrics are calculated and written in Tables 3 and 4, where column Sol means the number of non-dominated solutions generated by the search. Additionally, in Table 4 the total tardiness criterion is recorded. The results of these Tables can be used as benchmark to determine the quality of a Pareto front. However, different combination of criteria must be performed to determine the performance mainly of the makespan at trying to reduce the distance from the origin.

Table 3. Quality metrics for *Set A*

Instance	*Sol*	*HV*	*D*	*SP*	*MID*
mjs01	5	0.996	53.2	14.7	380.5
mjs02	7	0.995	50.7	4.4	408.8
mjs03	3	0.996	15.3	4.4	391.7
mjs04	5	0.995	62.2	2.2	450.5
mjs05	2	0.992	12.2	2.7	704.0
mjs06	4	0.993	47.9	11.1	603.4
mjs07	2	0.992	32.0	4.1	686.2
mjs08	1	0.993	0.0	0.0	616.4
mjs09	1	0.993	0.0	0.0	578.4
mjs10	1	0.983	0.0	0.0	1256.7
mjs11	1	0.983	0.0	0.0	1258.1
mjs12	3	0.985	33.1	10.4	1061.5
mjs13	1	0.986	0.0	0.0	1057.8
mjs14	4	0.983	39.0	11.0	1253.0
mjs15	1	0.975	0.0	0.0	1808.9
mjs16	1	0.976	0.0	0.0	1741.5
mjs17	1	0.97	0.0	0.0	2167.5
mjs18	1	0.97	0.0	0.0	1995.2
mjs19	1	0.97	0.0	0.0	2070.7
mjs20	1	0.981	0.0	0.0	1374.6
mjs21	1	0.985	0.0	0.0	1086.3
mjs22	2	0.984	26.0	3.7	1185.2
mjs23	1	0.984	0.0	0.0	1151.5
mjs24	1	0.985	0.0	0.0	1097.1
mjs25	1	0.974	0.0	0.0	1886.0
mjs26	1	0.974	0.0	0.0	1829.4
mjs27	1	0.971	0.0	0.0	2077.0
mjs28	1	0.973	0.0	0.0	1970.6
mjs29	1	0.975	0.0	0.0	1801.4
mjs30	1	0.997	0.0	0.0	252.0
mjs31	1	0.997	0.0	0.0	255.0
mjs32	1	0.997	0.0	0.0	258.0
mjs33	1	0.997	0.0	0.0	253.0
mjs34	1	0.997	0.0	0.0	241.0
mjs35	1	0.997	0.0	0.0	266.0
mjs36	1	0.997	0.0	0.0	246.0
mjs37	1	0.998	0.0	0.0	239.0
mjs38	1	0.997	0.0	0.0	271.0
mjs39	1	0.998	0.0	0.0	236.0
mjs40	1	0.998	0.0	0.0	241.0
mjs41	1	0.997	0.0	0.0	257.0
mjs42	1	0.997	0.0	0.0	250.0
mjs43	1	0.997	0.0	0.0	238.0
mjs44	1	0.997	0.0	0.0	270.0
mjs45	1	0.996	0.0	0.0	372.0
mjs46	1	0.996	0.0	0.0	429.0
mjs47	1	0.997	0.0	0.0	383.0
mjs48	1	0.996	0.0	0.0	373.0
mjs49	1	0.996	0.0	0.0	430.0
mjs50	1	0.995	0.0	0.0	467.2
mjs51	2	0.994	12.0	2.6	517.9
mjs52	1	0.996	0.0	0.0	421.0
mjs53	3	0.995	17.0	2.4	466.6
mjs54	1	0.996	0.0	0.0	407.0
mjs55	2	0.990	2.8	1.4	879.9
mjs56	3	0.990	29.1	2.4	864.6
mjs57	1	0.990	0.0	0.0	904.9
mjs58	2	0.990	19.7	3.7	875.6
mjs59	1	0.990	0.0	0.0	1043.6
mjs60	1	0.997	0.0	0.0	332.0
mjs61	1	0.996	0.0	0.0	359.0
mjs62	1	0.997	0.0	0.0	342.0
mjs63	1	0.997	0.0	0.0	341.0
mjs64	1	0.997	0.0	0.0	297.0
mjs65	1	0.994	0.0	0.0	569.0
mjs66	1	0.994	0.0	0.0	591.0
mjs67	1	0.994	0.0	0.0	628.0
mjs68	1	0.994	0.0	0.0	626.0

Table 4. Quality metrics for *Set C*

Instance	*Sol*	$\sum T_i$	*HV*	*D*	*SP*	*MID*
mjs01	31	61	0.992	216.7	15.7	3329.3
mjs02	18	179	0.992	145.2	22.1	3454.3
mjs03	12	42	0.992	141.7	10.4	3425.1
mjs04	20	241	0.990	184.3	13.7	3479.5
mjs05	9	1022	0.984	251.4	17.3	6445.7
mjs06	6	434	0.987	95.6	6.9	5541.1
mjs07	3	882	0.985	84.9	50.0	6145.8
mjs08	2	534	0.987	23.8	4.4	5462.3
mjs09	13	399	0.987	292.3	30.7	5359.3
mjs10	5	5315	0.970	432.8	136.0	10433.6
mjs11	2	5634	0.968	35.3	5.6	10652.4
mjs12	9	4358	0.973	50.3	52.4	9362.4
mjs13	3	4887	0.972	107.0	18.9	9800.6
mjs14	5	5523	0.968	348.6	27.6	10798.3
mjs15	2	8557	0.955	97.7	9.1	16230.3
mjs16	1	8114	0.957	0.0	0.0	15560.4
mjs17	6	10904	0.950	457.8	107.4	19429.9
mjs18	4	9972	0.950	166.5	72.7	18246.6
mjs19	3	9492	0.950	355.9	197.5	17542.5
mjs20	3	5851	0.970	287.6	25.5	11308.6
mjs21	9	4980	0.971	322.9	51.2	9925.7
mjs22	5	5376	0.970	612.0	77.8	10573.6
mjs23	2	5856	0.968	65.6	7.4	11045.3
mjs24	1	4993	0.970	0.0	0.0	9763.5
mjs25	7	9466	0.950	290.6	36.8	17645.6
mjs26	1	8306	0.956	0.0	0.0	15682.4
mjs27	4	10180	0.949	74.4	15.0	18405.6
mjs28	1	9537	0.951	0.0	0.0	17405.0
mjs29	6	8535	0.954	489.2	75.3	16364.6
mjs30	1	0	0.996	0.0	0.0	2018.3
mjs31	1	0	0.994	0.0	0.0	3123.1
mjs32	7	0	0.995	56.8	7.3	2324.8
mjs33	2	0	0.995	7.2	2.2	2190.1
mjs34	3	0	0.996	21.2	8.5	2029.2
mjs35	5	0	0.995	95.6	13.3	2284.3
mjs36	1	0	0.995	0.0	0.0	2201.1
mjs37	5	0	0.995	112.8	24.4	2115.3
mjs38	4	0	0.995	74.2	2.5	2293.9
mjs39	3	0	0.995	110.3	42.4	2221.5
mjs40	2	0	0.996	34.8	4.7	1994.9
mjs41	3	0	0.996	34.4	5.2	2174.0
mjs42	2	0	0.995	20.1	3.3	2035.4
mjs43	1	0	0.996	0.0	0.0	1924.3
mjs44	3	0	0.995	101.5	2.4	2223.4
mjs45	1	0	0.991	0.0	0.0	4127.3
mjs46	1	0	0.992	0.0	0.0	3962.6
mjs47	3	0	0.991	48.4	19.8	4191.7
mjs48	1	0	0.992	0.0	0.0	4005.8
mjs49	3	0	0.991	24.4	11.8	4120.5
mjs50	4	49	0.990	247.7	88.1	4419.4
mjs51	2	334	0.990	51.1	6.7	4764.9
mjs52	2	0	0.992	74.6	6.7	3824.0
mjs53	8	113	0.990	132.5	14.4	4590.2
mjs54	2	18	0.990	22,0	4,4	4262.1
mjs55	1	1428	0.980	0.0	0.0	8303.3
mjs56	2	251	0.985	21.3	4.1	6939.3
mjs57	3	2188	0.977	109.1	23.1	9261.0
mjs58	1	847	0.982	0.0	0.0	7723.8
mjs59	1	1779	0.979	0.0	0.0	8686.5
mjs60	1	0	0.992	0.0	0.0	3676.3
mjs61	2	0	0.993	6.4	2.1	3445.6
mjs62	2	0	0.993	34.0	4.9	3259.4
mjs63	2	0	0.993	30.7	4.4	3273.7
mjs64	1	0	0.994	0.0	0.0	2839.5
mjs65	1	0	0.988	0.0	0.0	6119.4
mjs66	3	483	0.984	173.6	121.6	7116.2
mjs67	2	618	0.983	246.4	13.9	7346.4
mjs68	3	0	0.987	7.8	1.4	6576.2

5 Conclusion

This paper illustrates a local search algorithm to solve the job shop scheduling Multi resource Resource flexibility with linear routes at minimizing different combinations of regular criteria in Pareto manner. We have presented results for three sets of criteria. To best of our knowledge, there is not any reference about of this extension of the job-shop scheduling problem to calculate regular criteria different than the makespan. The algorithm is supported in the adaptation of estimation functions as extensions of the classical problem and the proposition of a conjunctive graph. In its formulation, the properties of heads, tails and levels of operations have been considered. Computational results show the efficiency of our algorithm at getting some best known values and better values in some instances for the makespan. Additionally optimal solutions are gotten by the maximum tardiness and the total tardiness.

As perspective of the proposed approach, we will study two extensions. The first is the formulation of new estimation function for non-linear route problem (for example, in assembly process), and the second is the solution considering weighted objective functions for improving the making decision process. This research is supported by the project “Formulation and validation of heuristics for optimizing the customer service in flexible configurations in the Tolima region”, identified with the code 17-465-INT, which is financed by the Universidad de Ibagué (Colombia).

References

1. Dauzère-Pérès, S., Roux, W., Lasserre, J.: Multi-resource shop scheduling with resource flexibility. Eur. J. Oper. Res. **107**(2), 289–305 (1998)
2. Mati, Y., Dauzère-Pérès, S., Lahlou, C.: A general approach for optimizing regular criteria in the job-shop scheduling problem. Eur. J. Oper. Res. **212**(1), 33–42 (2011)
3. García-León, A., Dauzère-Pérès, S., Mati, Y.: Minimizing regular criteria in the flexible job-shop scheduling problem. In: 7th Multidisciplinary International Scheduling Conference: Theory and Applications, pp. 443–456 (2015)
4. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., da Fonseca, V.G.: Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans. Evol. Comput. **7**, 117–132 (2003)
5. García-León, A., Dauzère-Pérès, S., Mati, Y.: An efficient Pareto approach for solving the multi-objective flexible job-shop scheduling problem with regular criteria. Comput. Oper. Res. **108**, 187–200 (2019)
6. Dauzère-Pérès, S., Paulli, J.: An integrated approach for modeling and solving the general multiprocessor job-shop scheduling problem using tabu search. Ann. Oper. Res. **70**, 281–306 (1997)
7. Singer, M., Pinedo, M.: A computational study of branch and bound techniques for minimizing the total weighted tardiness in job shops. IIE Trans. **30**(2), 109–118 (1998)