37 ITERATIVE HEURISTICS FOR PERMUTATION FLOW SHOPS WITH TOTAL FLOWTIME MINIMIZATION

Xiaoping Li, Qian Wang
School of Computer Science &Engineering, Southeast University,
Nanjing, P.R .China, 210096
xpli@seu.edu.cn, qwang@seu.edu.cn

In this paper, flow shop scheduling problems with total flowtime minimization is considered. IRZ (iterative RZ, presented by Rajendran and Ziegler, EJOR, 1997) is found to be effective to improve solutions and LR (developed by Liu & Reeves, EJOR, 2001) is suitable for initial solution developing. By integrating FPE (forward pair wise exchange) and FPE-R(forward pair wise exchange-restart) with IRZ, two efficient composite heuristics, ECH1 and ECH2, are proposed. Computational results show that the proposed three outperform three best existing ones in performance and ECH1 is best. IRZ is the fastest heuristic. ECH2 is a trade-off between IRZ and ECH1 both in effectiveness and efficiency.

1. INTRODUCTION

Flow shop scheduling is an important manufacturing system widely existing in industrial environments. A flow shop can be described as n jobs being processed on m machines and each job having the same machine-order [2]. Total flowtime (or equivalently mean flowtime if all machines are available at time zero) is an important performance measure in flow shop scheduling which can lead to stable or even utilization of resources, rapid turn-around of jobs and minimizing in-process inventory [6][15].

A flow shop with respect to total flowtime minimization is NP-complete [7]. For decades, heuristics have been developed for the problems considered. Heuristics presented by Gupta[8], Rajendran & Chaudhuri[12], Rajendran[16], Ho[9] and Wang et al[18] were efficient algorithms for these problems before NEH [11] seems to be the best heuristic for makespan minimization, while it is not the best for total/mean flowtime minimization [9,12,16]. NEH is based on job-insertion, in which an unscheduled job of the seed generated by some rule is respectively inserted into every possible slot of the current solution (a schedule/partial schedule) and the best generated one is selected as the new current solution. For total/mean flowtime minimization flow shop scheduling, it seems that FL (proposed by Framinan and Leisten [5]), WY (presented by Woo and Yim [19]) and RZ (developed by

Rajendran and Ziegler [13]) are efficient constructive heuristics. FL is similar to NEH but every job-insertion is followed by a pair-wise exchange to improve the current solution. A pair-wise exchange is to generate solutions by exchanging positions of every pair of jobs in the current solution. If the current solution is worse than the best of the generated, it is replaced with the best. WY is also derived from NEH but no seed is predetermined and all unscheduled jobs should perform job-insertion. RZ is based on a different job-insertion from NEH, in which a seed is resulted by sorting jobs with weighted processing times and it is set as the current solution. Every job of the seed performs job-insertion to the subsequence of the current solution without the inserted job. From the comparisons of WY against RZ in [19][13], it is found that RZ outperforms WY for small instances but the relative performance of WY improves with the number of jobs and finally WY outperforms RZ. Computational results of Framinan and Leisten [5] show that FL outperforms both WY and RZ for majority of the randomly generated instances. The temporal complexities of FL and WY are $O(n^4m)$ while that of RZ is $O(n^3m)$.

Recently, many composite heuristics are proposed, such as IH1~IH7 (described by Allahverdi and Aldowaisan [1]), IH7-FL (given by Framinan and Leisten [5]), FLR1 and FLR2 (presented by Framinan, Leisten and Ruiz-Usano [4]). Of these heuristics, FLR2, IH7_FL and FLR1 seem to be the most efficient ones, which adopt FL, the most efficient constructive heuristic, to construct a solution or improve the current solution. Most existing heuristics improve their solutions only by some one-pass method. However, most solutions can be greatly improved by an iterative way, which will be shown by IRZ, ECH1 and ECH2 proposed in this paper.

The paper is organized as follows. Iterative methods and IRZ are described in section 2. In section 3, initial solution development is introduced. ECH1 and ECH2 are proposed in section 4. Computational results are described in section 5, followed by conclusions in section 6.

2. ITERATIVE METHODS

Framinan, Leisten and Ruiz-Usano [4] gave a clear framework to divide a heuristic into three phases: index development, solution construction and solution improvement. Each heuristic can include one or more of these phases. A heuristic is regarded as composite if it employs another heuristic for one or more of the three above-mentioned phases. Consequently, a heuristic is regarded as simple if it does not contain another heuristic within any of the three phases.

Procedure of most heuristics, such as constructive ones FL[5], WY[19] and RZ [13], is actually searching an optimal solution among the neighbor solutions generated from an initial solution (which may be produced by combination of several algorithms) by some rule. For example, sequences generated during RZ can be regarded as neighbor solutions from the ASC(w sum(pt)) produced seed by RZ-insertion rule. The best is selected from the neighbor solutions as the final solution of RZ. The solution of FL is also selected from neighbor solutions of the seed generated by DESC(sum(pt)), which are constructed by NEH-insertion and PE (pair-wise exchange) rule.

However, such heuristics may be considerably improved by iterating the rule, i.e. if the solution of a heuristic is better than its initial solution then it is set as the

new initial solution and performs the same search procedure again. This process repeats until the solution of some iteration (the best of the generated neighbor ones) is not better than its initial solution, which can be illustrated in Figure 1.

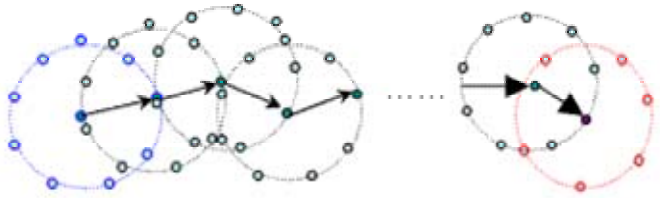

Figure 1 – Procedure of an Iterative Heuristic

Different heuristic has different improvement by iterative method. As for RZ, WY and FL, the corresponding iterative ones are denoted by IRZ (iterative RZ), IWY (iterative RZ) and IFL (iterative RZ) respectively. The six heuristics are tested by 110 benchmark instances generated by Tailard [17]. To compare effectiveness of the six heuristics, ARPD (Average Relative Percentage Deviation) is used which is similar to mean error (used by Allahverdi & Aldowaisan [1], Woo& Yim [19]) and Relative percentage increase in total flowtime (adopted by Rajendran &Ziegler [14]). In this paper, ARPD is slightly different from the one defined by Framinan &Leisten [5], Framinan, Leisten and Ruiz-Usano [4], which is defined as follows.

$$ARPD = \sum_{j=1}^{N} \left(F_i(H)/Best\_known_i - 1\right)/N \times 100 \qquad (1)$$

in which $F_i(H)$ is the total flowtime of instance i obtained by some heuristic H, $Best\_known_i$ is the best total flowtime among the compared heuristics for instance i and N is the number of instances for the same size of combination of jobs and machines. So N is 10 for Tailard's benchmark instances. ARPD of the six heuristics is shown in Table 1.

Table 1 – ARPD comparison of the six heuristics

Prob.	n	m	RZ	IRZ	WY	IWY	FL	IFL
T01	20	5	1.702	0.381	2.066	2.066	1.524	0.642
T02	20	10	1.231	0.164	2.121	2.106	1.762	0.981
T03	20	20	1.055	0.233	2.044	2.030	1.403	0.872
T04	50	5	1.559	0.364	2.418	2.211	0.920	0.302
T05	50	10	1.628	0.358	1.888	1.743	0.778	0.573
T06	50	20	1.394	0.197	1.678	1.661	0.739	0.438
T07	100	5	1.548	0.414	1.400	1.363	0.311	0.089
T08	100	10	1.792	0.180	1.382	1.190	0.939	0.156
T09	100	20	1.691	0.000	2.102	1.761	1.019	0.686
T10	200	10	1.685	0.388	1.484	1.264	0.399	0.075
T11	200	20	1.754	0.036	1.700	1.608	1.149	0.668
Average			1.549	0.247	1.844	1.728	0.995	0.498
Improved times				6.27		1.07		1.99

Note: Improved times is a ratio of ARPD of some heuristic to that of its iterative one.

Table 1 shows that iterative method increases RZ about 5.27 times, FL nearly 1 times but WY very little. Though RZ is far worse than FL in ARPD on average, IRZ (with ARPD 0.247) is even much better than IFL (with ARPD 0.498). WY and FL are based on NEH-insertion which is different from RZ-insertion. The above results indicate that iterative method can improve RZ-insertion much more than NEH-insertion.

Average iterations of RZ, FL and WY are about 8.4, 3.2 and 2.4 respectively. Also, because the temporal complexities of FL and WY are $O(n^4m)$ while that of RZ is $O(n^3m)$, CPU-time of IRZ should be much less that that of IWY or IFL for limited loops. For the above experiment, IRZ needs only 37.63s while IWY requires 194.18 and IFL spends 481.53 on average for class T11.

From above results, it can be seen that the iterative method is desirable for RZ-insertion based heuristics in both effectiveness and efficiency.

3. INITIAL SOLUTION DEVELOPMENT

There are several rules or algorithms (or their combination) for initial solution development, such as ASC(w sum(pt)), DESC(sum(pt)), ASC(sum(pt)), DESC(ABS(dif pt)) and LR(x) (developed by Liu & Reeves [10]). For IRZ, seven initial solution developing methods are performed on the instances in section 4.1. Experimental results indicate that LR can always obtain good results. On average, the best two are LR(n) and LR(n/m) while RANDOM is the worst. LR(1) and FLR1 can make IRZ obtain similar performance, which are only better than RANDOM. Though FLR1 (which is the combination of LR(1) and FL) outperforms FL, it always deteriorates performance of IRZ when it is used to generates initial solution instead of FL. In other words, good initial solution cannot ensure good result for IRZ. As for LR(x), performance of IRZ increases with x on average. However, the larger x is, the more CPU-time needs. For T11, CPU-times of IRZ with LR(n), LR(n/m) and LR(2) generating initial solution are 516.69s, 54.17s and 43.87s respectively.

From above, we can see that LR(n/m) is similar to LR(n) in performance while it only needs little more CPU-time than LR(2) does. Therefore, LR(n/m) is reasonably selected to develop initial solution.

4. NEW COMPOSITE HEURISTICS

In this section, three composite heuristics, IRZ, ECH1 and ECH2, are presented for flowtime minimization flow shop scheduling problems. For convenience, we combine index development and solution construction into one phase, initial solution development. So a heuristic consists of initial solution development and solution improvement phases. For the problem considered in this paper, iterative RZ-insertion is rather efficient for solution improvement and LR(n/m) is desirable for initial solution development, which can be illustrated in the following.

According to literature [4], pair-wise exchange strategies, such as FPE, FPE-R, BPE and BPE-R, can always efficiently improve solutions. In this subsection, FPE-

R and FPE are respectively integrated with RZ-insertion during IRZ procedure and ECH1 and ECH2 are proposed. In other words, ECH1 improves sequence of LR(n/m) by repeating the combination of RZ-insertion with FPE-R until no improvement can be made. ECH1 conducts in a similar way.

As well, the maximum iteration number of IRZ in the experiment of section 4.1 is 14 and the average is 8.4, which means that the improvement is very slight when the iteration-number exceeds some constant. So we choose the stop criterion as either improvement can be made or iteration-times is greater than a constant (20 in this paper), which is also applied to IRZ in the following experiments. Because of the similarity of ECH1 and ECH2, we just give the formal description and the execution procedure of ECH2 as follows, in which GFC is adopted.

1. Call LR(n/m) to generate seed π^s.
2. $k \leftarrow 1, \pi^c \leftarrow \pi^s$.
3. Repeat
   3.1 $\pi^s \leftarrow \pi^c$, $F(\pi^s) \leftarrow F(\pi^c)$.
   3.2 Call RZ-insertion method to obtain current solution π^c.
   3.3 Call FPE and keep the best solution among the procedure in π^b.
   3.4 If $F(\pi^b) < F(\pi^c)$, then $\pi^c \leftarrow \pi^b$ and $F(\pi^c) \leftarrow F(\pi^b)$.
   3.5 $k \leftarrow k+1$.
4. Until $F(\pi^s) \le F(\pi^c)$ or $k \ge 20$.
5. Stop. π^s is the final solution.

The time complexity in step 1 is $O(n^2)$. The number of loops in step 3 cannot exceed 20, which means that its time complexity depends on that of RZ and that of FPE. Time complexity of RZ is $O(mn^3)$ and it is obvious that time complexity of FPE is also $O(mn^3)$. Hence, time complexity of both IRZ and ECH2 is $O(mn^3)$. However, the worst computation effort of FPE-R is about cmn^4 in which c is a constant, i.e. the worst time complexity of FPE-R is $O(mn^4)$. So the time complexity of ECH1 is $O(mn^4)$.

To illustrate the computational procedure of ECH2, an 8-jobs 6-machines problem as shown in Table 2 is considered.

Table 2 – The 8-jobs 6-machines problem

	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8
M_1	22	36	9	89	16	23	66	82
M_2	57	61	87	34	80	78	95	54
M_3	14	23	52	39	23	28	63	50
M_4	82	1	15	12	72	36	28	42
M_5	93	55	43	4	20	72	26	75
M_6	64	80	4	62	74	67	19	14

The seed sequence obtained by LR(n/m) in step 1 is (2,4,1,5,8,3,6,7) with total flowtime 4171. Six iterations are performed in step 3 and 4, i.e. $k = 6$ when the heuristic stops, the corresponding results are shown in Table 3.

Table 3 – Solutions in ECH2 for the instance

No.	Operation	Result sequence	Flowtime
$k = 1$	RZ	(5,2,1,3,4,8,7,6)	4079
	FPE	(5,2,8,3,4,1,7,6)	4022
$k = 2$	RZ	(5,3,4,2,8,7,6,1)	3979
	FPE	(5,3,4,2,8,7,6,1)	3979
$k = 3$	RZ	(3,4,2,5,8,7,6,1)	3887
	FPE	(3,4,2,1,8,7,6,5)	3870
$k = 4$	RZ	(3,4,2,1,8,7,5,6)	3864
	FPE	(3,4,2,1,8,7,5,6)	3864
$k = 5$	RZ	(3,4,2,1,8,5,6,7)	3854
	FPE	(3,4,2,1,8,5,6,7)	3854
$k = 6$	RZ	(3,4,2,1,8,5,6,7)	3854
	FPE	(3,4,2,1,8,5,6,7)	3854

ECH2 stops when neither RZ nor FPE can improve the current solution and (3,4,2,1,8,5,6,7) is the final solution with total flowtime 3854.

5. EXPERIMENTAL RESULTS

To compare the proposed three heuristics with the best existing composite ones (FLR1 and FLR2, which have been proposed by Framinan, Leisten and Ruiz-Usano [4] and IH7_FL [5]) both in effectiveness and efficiency, all the 120 benchmark instances generated by Tailard [17] are tested. ARPD defined in subsection 4.1 is also adopted to evaluate effectiveness for each group, in which every current best solution $Best\_known_i$ is the minimum total flowtime among Rajendran & Ziegler [14], Liu & Reeves [10] and the six heuristics considered in this section. Efficiency is measured by average CPU-time (in seconds) spent on 10 instances of each combination of m and n.

All the heuristics are implemented by Visual Basic 6.0 and performed on IBM PC 2.0GHz with 256M RAM. Experimental results are given in Table 4 and Table 5.

Table 4 shows that the average performance of the proposed three is better than that of the existing three (IH7-FL, FLR1 and FLR2). ARPD of ECH1 is the least among the six composite heuristics, i.e. ECH1 is the most effective heuristic except two cases (IRZ is the best for T03 and ECH2 is the best for T04). ECH2 is worse than ECH1 but it outperforms the other four for majority cases. IRZ outperforms the existing three except that it is outperformed by FLR2 for T01, T02 and T03. As for the existing three, though ARPD of IH7-FL is less than that FLR2 for three cases (T04, T05, T10), FLR2 is better than IH7-FL on average. FLR1 is the worst for all cases among the compared heuristics.

Table 4 – Effectiveness comparison of the six heuristic

Prob	IH7-FL	FLR1	FLR2	IRZ	ECH1	ECH2
T01	1.621	1.702	1.011	1.016	0.680	0.711
T02	1.702	1.989	1.163	1.392	1.161	1.394
T03	1.406	2.439	1.383	1.398	1.414	1.448
T04	1.176	1.820	1.254	0.692	0.586	0.565
T05	1.600	2.740	1.813	1.142	0.861	1.095
T06	1.807	2.656	1.749	1.282	0.937	1.128
T07	0.856	0.864	0.504	0.353	0.165	0.193
T08	1.420	1.987	1.201	0.623	0.408	0.535
T09	1.724	2.420	1.645	1.311	1.148	1.244
T10	0.901	1.302	0.952	0.289	0.096	0.151
T11	1.112	1.406	0.910	0.172	0.121	0.139
T12	0.889	1.061	0.729	0.108	0.040	0.121
Aver.	1.351	1.866	1.193	0.815	0.635	0.727

Table 5 – Efficiency comparison of the six heuristics

Prob	IH7-FL	FLR1	FLR2	IRZ	ECH1	ECH2
T01	0.02	0.01	0.02	0.02	0.03	0.02
T02	0.02	0.02	0.03	0.02	0.03	0.03
T03	0.04	0.03	0.05	0.03	0.04	0.04
T04	0.28	0.24	0.32	0.32	0.41	0.41
T05	0.47	0.43	0.60	0.45	0.65	0.66
T06	0.89	0.80	1.08	0.59	1.14	0.88
T07	4.03	3.22	4.90	3.00	4.29	4.56
T08	7.65	5.85	9.35	3.92	9.02	6.04
T09	15.61	11.14	18.03	5.69	13.98	8.73
T10	112.14	103.57	146.94	43.08	122.00	71.48
T11	221.07	213.16	298.38	58.94	201.06	95.62
T12	6058.22	5804.27	8808.07	1521.61	8297.74	1920.65

Table 5 shows that all the compared composite heuristics have a similar time increasing tendency which is in accordance with their same time complexity. Though time complexity of ECH1 and the existing three is $O(mn^4)$, FLR2 and ECH1 are more time-consuming than the other two (FRL1 and IH7-FL) and they needs more than 8000s for T12 (500×20 instances) on average. IRZ and ECH2, with time complexity $O(mn^3)$, need much less CPU-time than the other four do (only 1521.61s and 1920.65s respectively for T12).

So among the six heuristics considered in this paper, ECH1 is the best in performance while it requires CPU-time nearly as much as FLR2 does. IRZ is the fastest, its overall performance is much better than FLR1, IH7-FL and FLR2 but worse than ECH1 and ECH2. ECH2 is a trade-off between IRZ and ECH1 in performance with CPU-time consuming similar to IRZ

6. CONCLUSIONS

In this paper, flow shop scheduling problems with flowtime minimization was considered. Based on iterative method and LR(n/m) initial solution development,

three composite heuristics, IRZ, ECH1 and ECH2, were proposed and compared with the best existing composite ones, FLR1, FLR2 and IH7_FL. Computational results showed that the proposed three outperform the best existing three. ECH1 is the best among the six heuristics in effectiveness while nearly needs as much CPU-time as FLR2 does. IRZ is the fastest but its overall performance is worse than ECH1 and ECH2. ECH2 is a trade-off between IRZ and ECH1 both in effectiveness and efficiency.

7. ACKNOWLEDGEMENT

This work is supported by National Natural Science Foundation of China under Grants (No. 60504029 and No.90412014).

8. REFERENCES

1. Allahverdi A, Aldowaisan T. New heuristics to minimize total completion time in m-machine flowshops. International Journal of Production Economics 2002; 77(1): 71-83.
2. Baker KR. Introduction to sequencing and scheduling. New York: Wiley, 1974.
3. Framinan JM, Leisten R, Rajendran C. Different initial sequences for the heuristic of Nawaz, Enscore and Ham to minimize makespan, idletime or flowtime in the static permutation flowshop sequencing problem. International Journal of Production Research 2003; 41(1): 121-148.
4. Framinan JM, Leisten R, Ruiz-Usano R. Comparison of heuristics for flowtime minimisation in permutation flowshops. Computers & Operations Research 2005; 32(5): 1237-1254.
5. Framinan JM, Leisten R. An efficient constructive heuristic for flowtime minimisation in permutation flow shops. OMEGA 2003; 31(4): 311-317.
6. French S. Sequencing and scheduling: an introduction to the mathematics of the job-shop. Chichester: Ellis Horwood, 1982.
7. Gray MR, Johnson DS, Sethi R. The complexity of flowshop and jobshop scheduling. Mathematics of Operations Research 1976; 1(2): 117-129.
8. Gupta JND. Heuristic algorithms for multistage flow shop problem. AIIE Transaction 1972; 4(1): 11-18.
9. Ho JC. Flowshop sequencing with mean flowtime objective. European Journal of Operational Research 1995; 81(3): 571-578.
10. Liu J, Reeves CR. Constructive and composite heuristic solutions to the P// Σ Ci scheduling problem. European Journal of Operational Research 2001; 132(2): 439-452.
11. Nawaz M, Enscore EE, Ham I. A heuristic algorithm for the m-machine n-job flow-shop sequencing problem. OMEGA 1983; 11(1): 91-95.
12 Rajendran C, Chaudhuri D. An efficient heuristic approach to the scheduling of jobs in a flowshop. European Journal of Operational Research 1991; 61(2): 318-325.
13. Rajendran C, Ziegler H. An efficient heuristic for scheduling in a flowshop to minimize total weighted flowtime of jobs. European Journal of Operational Research 1997; 103(1): 129-138.
14. Rajendran C, Ziegler H. Ant-colony algorithms for permutation flowshop scheduling to minimize makespan total flowtime of jobs. European Journal of Operational Research 2004; 155(2): 426-438.
15. Rajendran C. A heuristic for scheduling in flowshop and flowline-based manufacturing cell with multi-criteria. International Journal of Production Research 1994; 32(11): 2541-2558.
16. Rajendran C. Heuristic algorithm for scheduling in a flowshop to minimise total flowtime. International Journal of Production Economics 1993; 29(1): 65-73.
17. Tailard E. Benchmarks for basic scheduling problems. European Journal of Operational Research 1993; 64(1): 278-285.
18. Wang C, Chu C, Proth JM. Heuristic approaches for $n/m/F/\sum C_i$ scheduling problems. European Journal of Operational Research 1997; 96(3): 636-644.
19. Woo DS, Yim HS. A heuristic algorithm for mean flowtime objective in flowshop scheduling. Computers & Operations Research 1998; 25(3): 175-182.