International Journal of Industrial Engineering Computations 7 (2016) 353–366

Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Revisiting the NEH algorithm- the power of job insertion technique for optimizing the makespan in permutation flow shop scheduling

A. Baskar*

Principal, Apollo Engineering College, Mevaloorkuppam, Chennai-602 105, India

CHRONICLE

Article history:
Received July 21 2015
Received in Revised Format August 16 2015
Accepted September 1 2015
Available online September 4 2015

Keywords:
NEH Algorithm
Job Insertion Technique
Flow Shop Scheduling: Makespan

ABSTRACT

Permutation flow shop scheduling problems have been an interesting area of research for over six decades. Out of the several parameters, minimization of makespan has been studied much over the years. The problems are widely regarded as NP-Complete if the number of machines is more than three. As the computation time grows exponentially with respect to the problem size, heuristics and meta-heuristics have been proposed by many authors that give reasonably accurate and acceptable results. The NEH algorithm proposed in 1983 is still considered as one of the best simple, constructive heuristics for the minimization of makespan. This paper analyses the powerful job insertion technique used by NEH algorithm and proposes seven new variants, the complexity level remains same. 120 numbers of problem instances proposed by Taillard have been used for the purpose of validating the algorithms. Out of the seven, three produce better results than the original NEH algorithm.

1. Introduction

A flow shop scheduling problem (FSP) with *n* jobs and *m* machines has been a topic of research since 1950. Hejazi and Saghafian (2005) define the scheduling problem as "an effort to specify the order and timing of the processing of the jobs on machines, with an objective or objectives". Pinedo (2012) provides a vast literature about the scheduling theory, definitions, problems and tutorials. Among the many parameters that are being optimized, the makespan optimization is the choice of a good number of researchers in recent years. In permutation flow shop scheduling (PFSP), the total number of possibilities is *n*!, where *n* is the number of jobs to be scheduled. The problem is NP-Complete for more than 3 machines (Garey et al., 1976). In earlier days, Gantt charts (Gantt, 1919) were popular in scheduling which are still used by shop floor personnel for smaller problems. Many exact solutions have been proposed and analyzed by the researchers over the past decades. Most of them are based on branch and bound algorithm (example: Land & Doig, 1960) or mathematical/ computer programming (example: Yin

* Corresponding author.
E-mail: a.baaskar@gmail.com (A Baskar)

doi: 10.5267/j.ijiec.2015.9.001

et al., 2010). The computation time grows exponentially with the size of the problem and hence, heuristics that yield reasonably accurate and acceptable results are more popular in this area.

Johnson's (1954) algorithm reported optimal makespan for two machines, *n* number of jobs in a permutation flow shop problem. Since then, so many simple heuristics have been proposed over the years. The CDS algorithm proposed by Campbell et al. (1970) is the extension of Johnson's algorithm which selects the minimum makespan out of the (*m*-1) enumerations. The NEH algorithm proposed by Nawaz et al. (1983) uses the powerful job insertion technique after arranging the jobs in the descending order of their total processing times. It selects the first two jobs as the initial partial sequence and other jobs are inserted one by one from the third job to obtain a final optimal makespan and its corresponding sequence. It has been generally agreed that the NEH algorithm is regarded as one of the best available simple, constructive heuristics even today. The complexity level and the quality of a few earlier simple heuristics were studied in detail by Taillard (1990) and it was concluded that the NEH algorithm is a better one for different sizes of problems varying from 9 to 50 jobs. Framinan et al. (2003) considered twenty two different approaches and eight different sorting criteria, totaling 176 approaches for every objective function. Additionally, for every objective function, the RANDOM choice of a sequence was considered. The processing times were drawn between 1 to 99 time units. It was concluded that for the makespan optimization, the NEH-insertion approach which selects the first two jobs as the initial partial initial sequence is a clear winner. A study conducted by Kalczynski and Kamburowski (2007) also reaffirms this conclusion.

The SPIRIT algorithm of Widmer and Hertz (1989) is also a constructive heuristic based on the well-known traveling salesman problem. A few other constructive heuristic algorithms such as Rajendran (1993), Gajpal and Rajendran (2006) also use the powerful insertion technique of NEH to obtain the optimal sequence of jobs. Improvements have been performed on the NEH algorithm over the years to minimize the makespan as well as the computing time. The NEH heuristic algorithm itself has been an interesting area of research for many researchers. Many variants and improvements have been suggested over the last three decades. A few claims have been proved to be false also.

Sarin and Lefoka (1993) showed that their proposed algorithm was more effective when the number of machines is fewer than 100; but, NEH is better for larger number of machines. The performance of NEH algorithm was improved by improving the sorting step by Xiao-ping et al. (2004). Dong et al. (2008) ordered the jobs by the sums of the average processing time and the standard deviations of processing times to achieve the improvement. The Min-Max algorithm developed by Ronconi (2004) addresses the makespan minimization PFSP with no buffers. He found that the heuristic yields better results than the NEH algorithm for larger problems. Chakraborty and Laha (2007) modified the NEH algorithm using population-based technique and compared the results with NEH and HFC heuristic (Koulamas, 1998). They claimed that their algorithm outperforms NEH and HFC for the tested problem instances. The performance of the HFC algorithm is reported to outperform the NEH for non-permutation schedules.

Based on the concept of Johnson's algorithm, Kalczynski and Kamburowski (2008) proposed a new priority order combined with a simple tie-breaking method that leads to a heuristic that outperformed the NEH algorithm for all the problem sizes. Liu et al. (2012) proposed two new techniques to improve the NEH algorithm. The running time is reduced by introducing block properties and tie break rules to obtain solutions with smaller makespan. Singhal et al. (2012) considered the first four jobs from the sorted list and based on a parameter *k*, better partial sequences are considered for further job insertion. They claimed that the results are better than the NEH algorithm. However, the claim is not supported by rigorous validation except one numerical example and it is similar to the algorithm proposed by Chakraborty and Laha. Semančo and Modrák (2012) proposed one MOD approach for solving the PFSP. It uses the difference between the sums of processing times for each machine as a pair-splitting strategy to make two groups of the matrix of *n*-job and *m*-machine. Johnson's rule was used in the last step to get the minimum makespan. MOD's average relative percentage deviation for all the 120 Taillard's problems

was reported as 6.88%, which is inferior to that of the NEH. Ancău (2012) proposed two variants of heuristic algorithms to solve the permutation flow shop scheduling problem. The first algorithm is a constructive greedy heuristic which builds the optimal schedule of jobs on the basis of a selective-greedy process. The second algorithm is a modified version of the first algorithm, based on the iterative stochastic start.

Baskar and Xavior (2013) considered the first and last jobs, middle two jobs and the last two jobs as the initial partial sequences and analyzed the performance using Taillard (1993) instances. Initial sequencing of jobs is based on non-increasing order of their total processing times as proposed by NEH. Fernandez-Viagas and Framinan (2014) presented a new tie-breaking mechanism based on an estimation of the idle times of the different subsequences in order to pick the one with the lowest value of the estimation. The computational experiments carried out by them showed that this mechanism outperforms the existing ones, both for the NEH and the iterated greedy algorithms for different CPU times.

Ribas and Mateo (2010) proposed two tools to improve the performance of NEH based heuristics. The first tool uses the reversibility property and the second one is a new tie-breaking strategy to use with the insertion phase of the NEH heuristic. They analyzed five initial solution procedures for both the problems. The algorithm recently proposed by Gupta and Chauhan (2015) performs better than many simple heuristics like Palmer, CDS and RA algorithms; but, could not perform better than the NEH algorithm. The solutions have been further improved by meta-heuristics. The results obtained from simple heuristics like NEH are mostly used as the candidate solution by these meta-heuristics to refine the solution further. Many meta-heuristics based on Simulated Annealing can be found in Osman and Potts (1989), Ogbu and Smith (1990) and Ishibuchi et al. (1995). Meta-heuristics based on Tabu Search techniques (Example: Nowicki & Smutnicki, 1996) and Evolutionary algorithms (Example: Reeves, 1995; Caraffa et al., 2001; Onwubolu & Davendra, 2006; Sayadi et al., 2010; Erdogmus, 2010) are also available in the literature.

2. NEH Heuristic Algorithm

NEH algorithm proposed by Nawaz et al. (1983) is widely accepted as one of the best simple heuristics for makespan minimization in permutation flow shop scheduling problems. The algorithm essentially consists of three steps. (i) All the jobs are scheduled in non-increasing order of their total processing times (ii) First two jobs are considered as an initial partial sequence (iii) Remaining jobs are inserted one by one starting from the third job at the place, which minimizes the partial makespan among the possible options. Total number of sequences to be enumerated is [n (n+1) /2 – 1] and hence, the complexity level is $\partial (n^3m)$. All the three components contribute to the efficient performance of the algorithm. Many initial sequences have been considered by other researchers in step (i) of NEH algorithm and the performances have been analyzed. (Example: Nagano & Moccellin, 2002; Pour, 2001).

Ruiz and Maroto (2005) carried out an exhaustive review of eighteen constructive and improvement heuristics that include the Johnson (1954), Palmer (1965), CDS (1970), Gupta (1971), RA (1977) and NEH (1983) heuristics. They concluded that the NEH algorithm is one of the best simple algorithms available up to 2005 outperforming many meta-heuristics. For validating the algorithms, they have used Taillard (1993) benchmark problems. The reported average % increase for the NEH algorithm over the best known solution for the 120 problem instances was 3.33 %. The Palmer, Gupta and Hundal and Rajgopal (1988) algorithms make use of a slope index assigned to each job. These heuristic algorithms sort the list of jobs using that weight as a sort key to create a feasible schedule.

3. Proposed Variants

Initial arrangement of the jobs may be based on any parameter and many researchers have tried in different ways to have a better initial sequencing. Also, the choice of initial partial sequence is numerous.

The NEH algorithm selects the first two jobs after arranging them in non-increasing order of their total processing times. Out of the three steps in NEH algorithm, only the third step, that is, the job insertion technique is considered and the following variants listed in Table 1 are analyzed in this paper. Only two cases of initial arrangement of jobs are considered, (i) arranging them in non-increasing order of their total processing times and (ii) taking the data as it is without any sorting. The reason being, the purpose is to analyze the power of the job insertion technique. The initial sequence has been selected in many ways and the final sequence is constructed using the job insertion technique. Let,

OSEQ – The sequence of the jobs in original data
PSEQ – The sequence of the jobs after arranging them in non-increasing order of total processing times
n – Number of jobs
m – Number of machines

Table 1
Variants of NEH algorithm

S.No	Algorithm	Initial Sequence	Further Job Insertion
1	NEH	First two jobs from PSEQ	Other jobs one by one from PSEQ
2	NEHAB1	Job nos. [7, 14] if n=20; [17, 34] if n=50; [34, 68] if n=100; [67, 134] if n=200 and [167, 334] if n=500 from OSEQ	Other jobs one by one from PSEQ
3	NEHAB2	Job nos. [1, 11] if n=20; [1, 26] if n=50; [1, 51] if n=100; [1, 101] if n=200 and [1, 251] if n=500 from OSEQ	Other jobs one by one from PSEQ
4	NEHAB3	First two jobs from OSEQ	Other jobs one by one from OSEQ
5	NEHAB4	Job nos. [1, 11] if n=20; [1, 26] if n=50; [1, 51] if n=100; [1, 101] if n=200 and [1, 251] if n=500 from PSEQ	Other jobs one by one from PSEQ
6	NEHAB5	Job nos. [10, 11] if n=20; [25, 26] if n=50; [50, 51] if n=100; [100, 101] if n=200 and [250, 251] if n=500 from OSEQ	Other jobs one by one from PSEQ
7	NEHAB6	Job nos. [7, 14] if n=20; [17, 34] if n=50; [34, 68] if n=100; [67, 134] if n=200 and [167, 334] if n=500 from PSEQ	Other jobs one by one from PSEQ
8	NEHAB7	Job nos. [7, 14] if n=20; [17, 34] if n=50; [34, 68] if n=100; [67, 134] if n=200 and [167, 334] if n=500 from OSEQ	Other jobs one by one from OSEQ

4. Results and Discussion

The algorithms are coded in MATLAB R2008a and run in an i5 PC with 4 GB RAM. For validating the algorithms, the 120 numbers of Taillard (1993) benchmark instances have been used. They are grouped into 12 sets of 10 problems each with varying sizes having 20, 50, 100, 200 and 500 jobs and 5, 10 and 20 machines. The lower bounds are available in the Taillard paper itself and the known upper bounds as of April 2005 (www.mistic.heig-vd.ch/taillard/problems.dir/ordonnancement,dir/flowshop.dir/best_lb_up.txt) were taken from the website. The obtained makespan are reproduced in Table 2. The performance measure being considered for the analysis purpose is:

% increase in Makespan from the known upper bound (Relative Deviation, RD),
= (Makespan obtained- Known Upper Bound) x 100) / Known Upper Bound.

The average % increase in Makespan from the known upper bound (Relative Deviation, RD), are computed and presented Table 3.

Table 2
Makespan for different Algorithms using Taillard Problems

m/c × Job	UB	NEH	NEHAB1	NEHAB2	NEHAB3	NEHAB4	NEHAB5	NEHAB6	NEHAB7
	1278	1286	1286	1299	1310	1299	1300	1297	1297
	1359	1365	1367	1365	1383	1383	1377	1383	1383
	1081	1159	1139	1140	1132	1150	1123	1139	1141
	1293	1325	1337	1314	1355	1364	1314	1315	1354
5×20	1235	1305	1244	1250	1277	1244	1305	1305	1283
	1195	1228	1212	1229	1224	1231	1234	1236	1237
	1234	1278	1280	1251	1276	1269	1266	1251	1253
	1206	1223	1224	1278	1248	1223	1221	1257	1239
	1230	1291	1270	1291	1263	1278	1253	1245	1258
	1108	1151	1153	1151	1131	1151	1161	1153	1153
	1582	1680	1655	1631	1665	1680	1631	1623	1658
	1659	1729	1729	1735	1753	1776	1729	1729	1767
	1496	1557	1541	1550	1602	1566	1574	1570	1583
	1377	1439	1439	1441	1465	1424	1443	1452	1437
10×20	1419	1502	1515	1493	1503	1502	1505	1498	1511
	1397	1453	1473	1447	1515	1434	1487	1474	1520
	1484	1562	1550	1581	1529	1527	1541	1523	1557
	1538	1609	1585	1663	1635	1605	1599	1583	1619
	1593	1647	1644	1652	1645	1647	1646	1678	1668
	1591	1653	1683	1639	1670	1634	1639	1635	1662
	2297	2410	2397	2422	2479	2402	2402	2394	2446
	2099	2150	2160	2126	2167	2184	2159	2163	2175
	2326	2411	2382	2407	2512	2405	2403	2398	2449
	2223	2262	2281	2240	2278	2280	2240	2264	2283
20×20	2291	2397	2397	2367	2341	2367	2363	2383	2435
	2226	2349	2281	2337	2354	2328	2357	2343	2279
	2273	2362	2348	2356	2386	2338	2356	2393	2350
	2200	2249	2249	2257	2314	2273	2265	2282	2283
	2237	2320	2299	2320	2387	2330	2320	2349	2384
	2178	2277	2235	2296	2289	2277	2276	2300	2297
	2724	2733	2733	2733	2729	2733	2733	2733	2744
	2834	2843	2882	2843	2933	2843	2843	2882	2925
	2621	2640	2630	2638	2709	2640	2647	2630	2676
	2751	2782	2782	2832	2778	2804	2812	2811	2805
5×50	2863	2868	2868	2868	2891	2868	2868	2897	2891
	2829	2850	2840	2863	2873	2841	2850	2841	2882
	2725	2758	2745	2763	2751	2753	2753	2767	2736
	2683	2721	2698	2688	2725	2707	2710	2713	2739
	2552	2576	2575	2574	2613	2574	2574	2574	2596
	2782	2790	2786	2793	2794	2789	2786	2798	2791
	2991	3135	3151	3156	3194	3176	3178	3117	3140
	2867	3032	3012	3038	3046	3024	3013	3097	3028
	2839	2986	2987	2987	3077	2973	2995	3009	3156
	3063	3198	3136	3139	3202	3177	3147	3203	3218
10×50	2976	3160	3097	3129	3142	3153	3144	3174	3149
	3006	3178	3184	3172	3154	3165	3165	3177	3139
	3093	3277	3291	3289	3239	3277	3310	3265	3296
	3037	3123	3196	3185	3170	3153	3168	3171	3167
	2897	3002	3062	3078	3062	3079	3035	3029	3032
	3065	3257	3229	3276	3263	3209	3294	3232	3285
	3850	4082	4078	4044	4099	4044	4069	4122	4078
	3704	3921	4009	4011	4029	4021	3869	3865	4047
	3640	3927	3882	3890	3882	3852	3894	3887	3907
	3723	3969	3936	3942	4037	3934	3991	3966	3991
20×50	3611	3835	3879	3860	3878	3865	3899	3861	3859
	3681	3914	3901	3892	3968	3902	3917	3903	3996
	3704	3952	3897	3958	3986	3949	3936	3952	3939
	3691	3938	3925	3903	3976	3932	3950	3926	3982
	3743	3952	3922	3970	3972	3997	3996	4004	3971
	3756	4079	3991	3953	3958	3991	4069	4028	3964

Table 2

Makespan for different Algorithms using Taillard Problems (Contd…)

m/c × Job	UB	NEH	NEHAB1	NEHAB2	NEHAB3	NEHAB4	NEHAB5	NEHAB6	NEHAB7
	5493	5519	5519	5582	5527	5560	5552	5552	5533
	5268	5348	5348	5341	5326	5348	5348	5329	5350
	5175	5219	5219	5220	5262	5253	5208	5201	5247
	5014	5023	5023	5021	5028	5023	5049	5023	5022
	5250	5266	5266	5261	5298	5261	5267	5267	5263
5×100	5135	5139	5141	5139	5192	5139	5139	5139	5192
	5246	5259	5259	5266	5340	5289	5292	5270	5304
	5094	5120	5108	5113	5137	5105	5131	5126	5151
	5448	5489	5489	5489	5514	5489	5500	5489	5504
	5322	5341	5349	5349	5357	5342	5346	5346	5382
	5770	5846	5881	5866	5920	5864	5893	5894	5882
	5349	5453	5421	5478	5550	5431	5466	5431	5533
	5676	5824	5752	5756	5841	5775	5756	5834	5810
	5781	5929	5986	5982	6112	5958	5988	5929	6086
10×100	5467	5679	5621	5639	5672	5717	5693	5687	5628
	5303	5375	5365	5366	5447	5347	5376	5369	5468
	5595	5704	5718	5719	5712	5713	5744	5678	5739
	5617	5760	5740	5761	5829	5789	5762	5729	5808
	5871	6032	6013	6040	6037	6018	6014	6009	6063
	5845	5918	5903	5903	5925	5920	5909	5928	6041
	6202	6541	6582	6574	6631	6586	6505	6603	6581
	6183	6523	6524	6494	6502	6516	6571	6622	6534
	6271	6639	6628	6632	6583	6656	6664	6603	6609
	6269	6557	6590	6624	6687	6572	6609	6608	6572
20×100	6314	6695	6645	6721	6720	6640	6612	6658	6692
	6364	6664	6741	6670	6772	6696	6741	6742	6716
	6268	6632	6567	6657	6605	6628	6623	6564	6657
	6401	6739	6774	6744	6869	6778	6791	6723	6846
	6275	6677	6563	6655	6636	6560	6614	6617	6737
	6434	6677	6762	6710	6741	6771	6727	6734	6785
	10862	10912	10942	10942	11033	10952	11032	10954	10988
	10480	10716	10735	10722	10728	10660	10640	10702	10770
	10922	11025	11028	11048	11172	11062	11098	11027	11175
	10889	11057	11057	11057	11057	11057	11057	11057	10954
10×200	10524	10645	10623	10617	10689	10636	10658	10654	10693
	10329	10458	10458	10460	10559	10530	10439	11449	10626
	10854	10989	10991	11039	11053	10967	10991	11005	11003
	10730	10829	10856	10887	10913	10851	10867	10846	10980
	10438	10574	10625	10574	10639	10578	10580	10635	10599
	10675	10807	10870	10811	10946	10773	10822	10823	10927
	11195	11625	11663	11600	11734	11648	11668	11597	11630
	11203	11675	11798	11741	11815	11710	11771	11736	11847
	11281	11852	11858	11806	11848	11766	11810	11848	11831
	11275	11803	11729	11728	11767	11709	11780	11800	11892
20×200	11259	11685	11691	11719	11763	11685	11692	11649	11803
	11176	11629	11679	11657	11699	11759	11725	11661	11709
	11360	11833	11803	11758	11878	11796	11852	11815	11942
	11334	11913	11797	11783	11844	11896	11851	11870	11847
	11192	11673	11646	11619	11767	11688	11651	11625	11733
	11288	11869	11820	11754	11875	11784	11703	11755	11854
	26059	26670	26703	26686	26802	26731	26843	26733	26843
	26520	27232	27238	27258	27255	27144	27276	27312	27292
	26371	26848	26917	26887	27108	26982	26861	26977	27182
	26456	27055	26993	26913	27043	26946	26943	26934	26990
20×500	26334	26727	26784	26887	26953	26813	26957	26830	26900
	26477	26992	26941	27157	27079	27083	27030	27074	27104
	26389	26797	26746	26702	26932	26838	26834	26783	27050
	26560	27138	27237	27073	27153	27153	27165	27205	27270
	26005	26631	26618	26652	26924	26584	26586	26602	26734
	26457	26984	26955	26977	27044	27080	26932	27021	27080

Table 3
Average % increase in Makespan from the known upper bound (Relative Deviation)

Size	NEH (Ref.)	NEHAB1	NEHAB2	NEHAB3	NEHAB4	NEHAB5	NEHAB6	NEHAB7
5×20	3.300288	2.466798	2.94145	3.12553	3.105381	2.797727	3.0305926	3.153593
10×20	4.601116	4.503733	4.611437	5.632414	4.318363	4.397786	4.1938635	5.613124
20×20	3.730891	3.028943	3.464978	5.152144	3.73614	3.534852	4.109836	4.595636
5×50	0.727204	0.639589	0.845576	1.589029	0.693401	0.7814	1.0256895	1.539656
10×50	5.072897	5.067393	5.418086	5.770844	5.208919	5.408071	5.511376	5.974275
20×50	6.648051	6.251361	6.261627	7.234613	6.431331	6.707959	6.4955621	7.099907
5×100	0.527212	0.522582	0.631669	1.020972	0.687648	0.733877	0.5604858	0.957314
10×100	2.21498	1.994005	2.194672	3.151677	2.23288	2.359485	2.1542496	3.16718
20×100	5.344636	5.390678	5.561187	5.977305	5.434273	5.520767	5.5521205	5.95049
10×200	1.230268	1.392741	1.363868	1.957031	1.280906	1.386078	2.3294067	1.892757
20×200	4.435269	4.37247	4.088959	4.822075	4.334303	4.389163	4.2574951	4.908013
20×500	2.066128	2.088155	2.11122	2.530346	2.17239	2.200622	2.2164883	2.586686
Mean	3.324912	3.143204	3.291227	3.996998	3.302995	3.351482	3.4530972	3.953219
Rank	4	1	2	8	3	5	6	7

In case of the NEH Algorithm, Ruiz reported an average increase of 3.33 % over the best known solution for the 120 problem instances. Whereas, the average obtained by the author is 3.3249 %. The CDS, Palmer, Gupta and RA heuristics are not considered as the deviations are very high when compared to the other heuristics. The average increase is below 4% of all the cases analyzed. Even the NEHAB3 algorithm that selects the first two jobs as the initial partial sequence from the original data without sequencing the jobs initially, reports a deviation less than 4%. Also, the algorithm NEHAB7 that selects different jobs initially from the original data reports a deviation of 3.953219 %, which is slightly better than NEHAB3. Only, the job insertion technique is used to construct the final schedule from the original data. The performances of these two are better than most of the simple heuristics, including that of Suliman (2000) in which the reported increase was 6.21 %.

The heuristics analyzed in this paper perform better than a few meta-heuristics also:
GA Sprit - 7.15 % (Widmer & Hertz, 1989);
One GA – 4.75 % (Chen et al., 1995);
One Hybrid GA – 8.92 % (Murata et al., 1996) and
Another GA – 12.53 % (Ponnambalam et al., 2001).

The individual performances are also analyzed and reported in Table 4 with respect to the number of problem instances they report better makespan when compared with the NEH (reference). NEHAB1 reports better results in case of 57 instances, whereas, NEH reports in 45 problems only. In 18 cases, the reported makespan are the same. NEHAB1 reports a mean deviation of 3.143204 %, which is the least among the heuristics including the NEH.

Table 4
Individual Performance based on the Number of Instances

S.No.	Algorithm	Self	NEH (Ref.)	Same Makespan	Rank
1	NEHAB1	57	45	18	1
2	NEHAB2	49	59	12	5
3	NEHAB3	26	93	01	8
4	NEHAB4	53	50	17	2
5	NEHAB5	47	63	10	6
6	NEHAB6	52	59	09	4
7	NEHAB7	28	92	00	7

The rank of NEH is taken as 3 in Table 4. This ranking is different from the one based on the mean deviation from the known upper bounds. NEH holds the position of 4 there in Table 3. In both the cases, NEHAB1 that selects the initial partial sequence as the job numbers [7, 14] if n=20; [17, 34] if n=50; [34, 68] if n=100; [67, 134] if n=200 and [167, 334] if n=500 from the original data performs extremely

better. Other jobs are inserted one by one from the sequence of the jobs after arranging them in non-increasing order of total processing times. The mean deviations are presented graphically in Fig. 1 to Fig. 3.

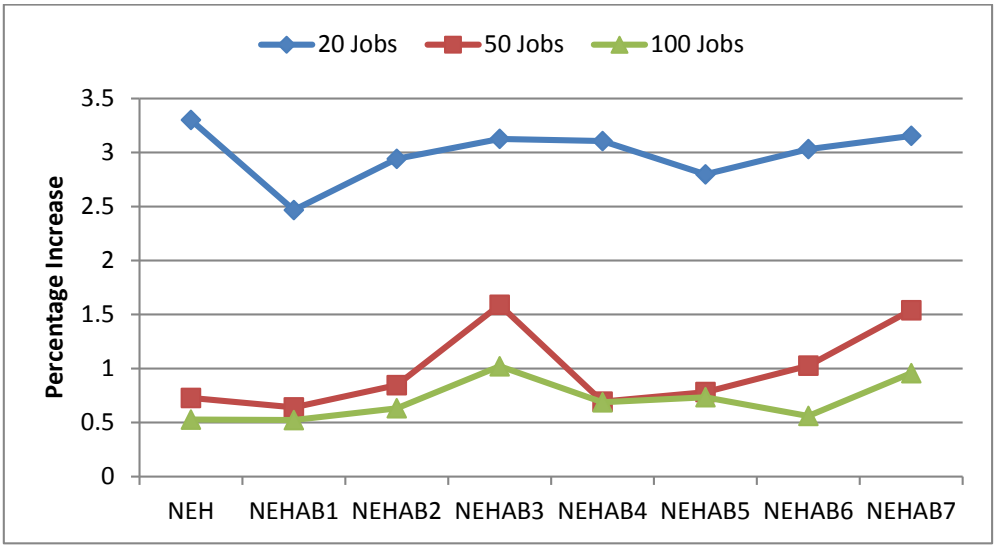

Fig. 1. Percent Increase Plot for 5 Machines

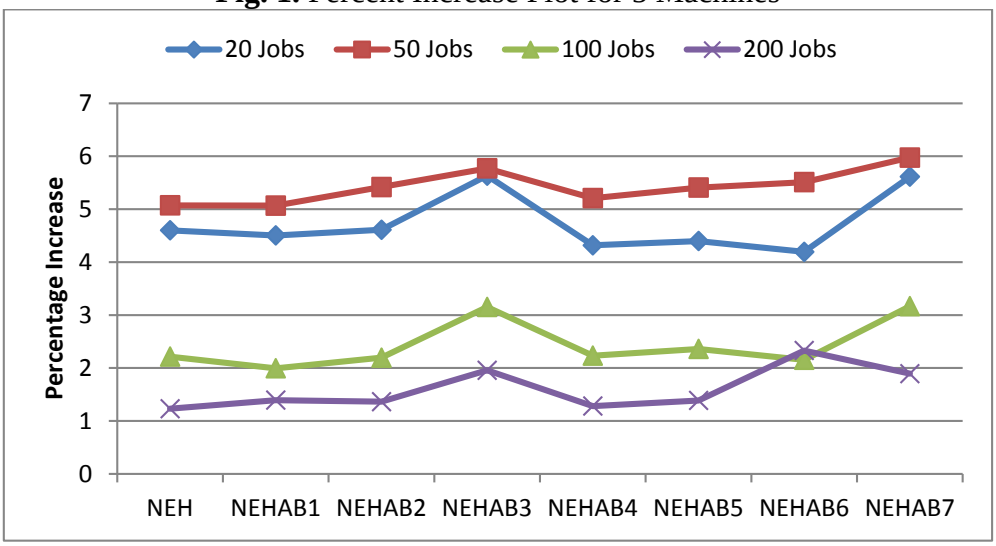

Fig. 2. Percent Increase Plot for 10 Machines

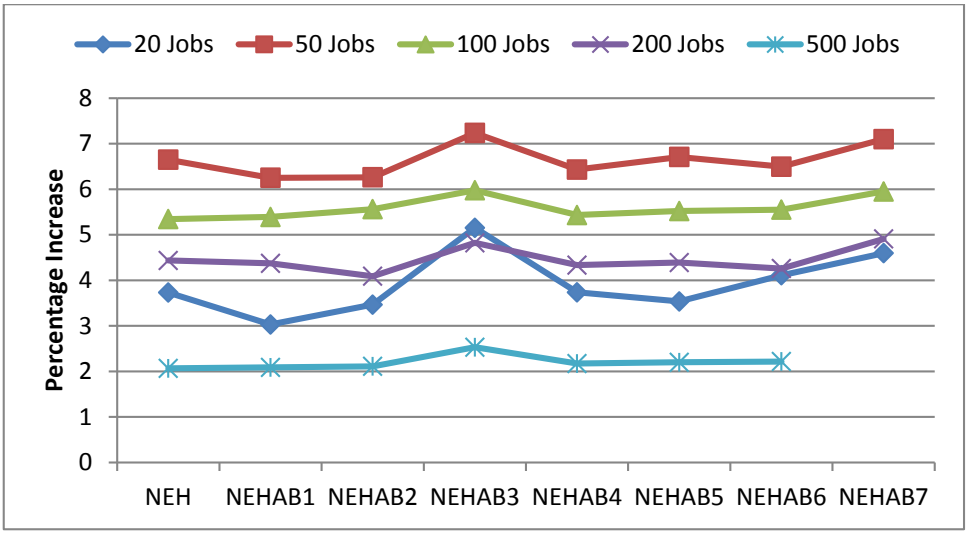

Fig. 3. Percent Increase Plot for 20 Machines

The algorithms perform better in case of 5 machines, 100 jobs; 10 machines, 200 jobs and 20 machines, 500 jobs problem instances. That is, in the highest number of jobs for a particular number of machines. When the percentage deviations are analyzed, it is more than 5 % in case of 10 machines, 50 jobs; 20 machines, 50 jobs and 20 machines, 100 jobs.

5. ANOVA

One way ANOVA has been carried out at 95% confidence level. The variance analysis has been executed using the MINITAB 17® software.

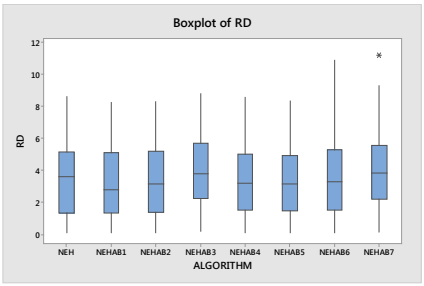

Fig.4. Box Plot for the Relative Deviations

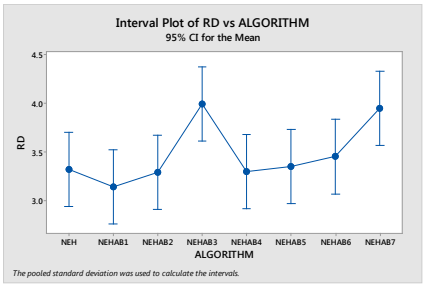

Fig. 5. Interval Plot for the Relative Deviations

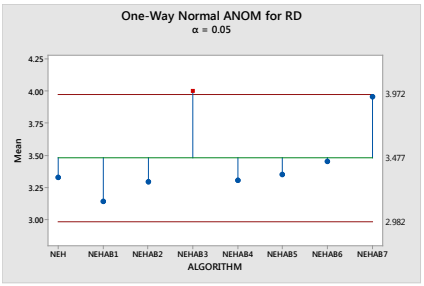

Fig. 6. One Way Normal Analysis of Means for Relative Deviations

The box plot, interval plot and normal analysis of means are shown from Fig. 4 to Fig. 6. The performance of NEHAB1 is better than the others. The means of NEHAB3 and NEHAB7 are above the average level. These two algorithms do not pre-arrange the jobs in any specific order, but, uses the job insertion technique to arrive with the final sequence. However, the performance of these two also better than many heuristics proposed over the years, the deviations being below 4 %. This clearly shows the power of the job insertion technique in minimizing the makespan in permutation flow shop problems. The output produced by the software is reproduced below:

```
Analysis of Variance

Source      DF   Adj SS  Adj MS  F-Value  P-Value
ALGORITHM    7    85.54  12.220     2.70    0.009
Error      952  4308.99   4.526
Total      959  4394.53


Model Summary

      S   R-sq  R-sq(adj)  R-sq(pred)
2.12750  1.95%      1.23%       0.29%


Means

ALGORITHM    N   Mean  StDev      95% CI
NEH        120  3.325  2.121  (2.944, 3.706)
NEHAB1     120  3.143  2.037  (2.762, 3.524)
NEHAB2     120  3.291  2.117  (2.910, 3.672)
NEHAB3     120  3.997  2.209  (3.616, 4.378)
NEHAB4     120  3.303  2.052  (2.922, 3.684)
NEHAB5     120  3.351  2.096  (2.970, 3.733)
NEHAB6     120  3.453  2.172  (3.072, 3.834)
NEHAB7     120  3.953  2.209  (3.572, 4.334)

Pooled StDev = 2.12750
```

The null hypothesis and the alternate are:
H_0: all means are same
H_A: at least one mean is different

In the analysis, the F value is high (2.70) and the P value is < 0.05 (0.009) and hence, the Null Hypothesis is rejected. The algorithms are statistically different.

6. Interaction and Surface Plots

For validating further, interaction and Surface plots have been drawn using MINITAB17® software for the results obtained. The maximum order of terms in the model has been set as 2.

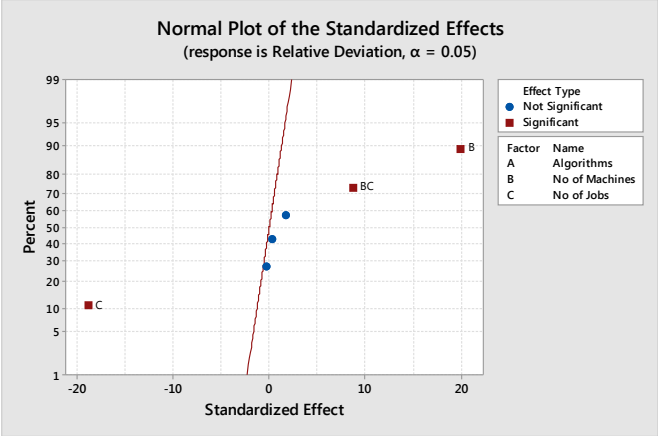

Fig. 7. Normal Plot of the Standardized Effects

In the analysis, algorithm '0' represents the NEH algorithm, '1' represents the NEHAB1 algorithm and so on. According to the Normal Plot of the Standardized Effects (Fig. 7), the factors B (number of machines), C (number of jobs) and BC have a significant effect on the response (Relative Deviation) for a particular algorithm.

Analysis of Variance

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Model	7	2358.57	336.938	157.55	0.000
1-Linear	3	882.81	294.271	137.60	0.000
Algorithms	1	5.51	5.515	2.58	**0.109**
No of Machines	1	852.26	852.264	398.51	0.000
No of Jobs	1	762.00	762.00	456.30	0.000
2-Way Interactions	3	164.13	54.710	25.58	0.000
Algorithms*No of Machines	1	0.82	0.824	0.39	**0.535**
Algorithms*No of Jobs	1	0.61	0.609	0.28	**0.594**
No of Machines*No of Jobs	1	163.31	163.307	76.36	0.000
3-Way Interactions	1	1.34	1.344	0.63	**0.428**
Algorithms*No of Machines*No of Jobs	1	1.34	1.344	0.63	**0.428**
Error	952	2035.97	2.139		
Lack-of-Fit	88	1070.95	12.170	10.90	0.000
Pure Error	864	965.02	1.117		
Total	959	4394.53			
Model Summary					
S	R-Sq(sdj)	R-sq(pred)			
1.46240	53.67%	53.33%			

Coded Coefficients

Term	Effect	Coef	SE Coef	T-Value	P-Value	VIF
Constant		1.099	0.143	7.68	0.000	
Algorithms	0.702	0.351	0.219	1.61	**0.109**	9.20
No of Machines	6.096	3.048	0.153	19.96	0.000	7.33
No of Job2	-6.686	-3.343	0.177	-18.88	0.000	4.14
Algorithms*No of Machines	-0.289	-0.145	0.233	-0.62	**0.535**	7.36
Algorithms*No of Job2	0.289	0.144	0.271	0.53	**0.594**	9.10
No of Machines*No of Job2	3.193	1.596	0.183	8.74	0.000	6.87

The p-value is small (<0.05, a level of significance) for the factors; the number of machines, the number of jobs and the interaction between the number of machines * number of jobs. They have statistically significant effect on the response. However, the algorithms and the other interactions are significantly different statistically.

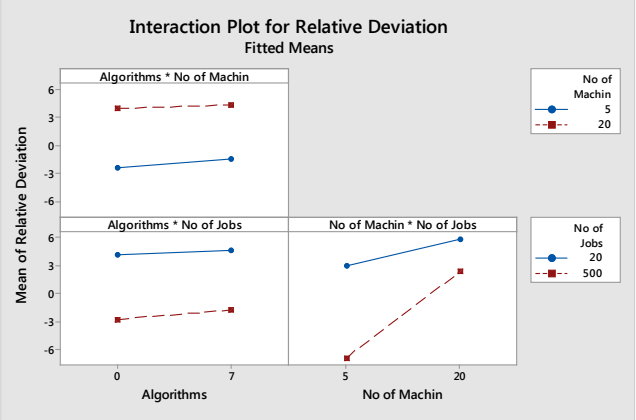

Fig. 8. Interaction Plot for Relative Deviation

The interaction plot is presented in Fig. 8. For a particular algorithm, the relative deviation is less for the combination of less number of machines with a large number of jobs.

The surface plots of relative deviations are shown from Fig. 9 and Fig. 10. The relative deviation is high as the number of machines increases. Algorithm 3 (NEHAB3) reports the highest deviations in general followed by the Algorithm 7 (NEHAB7). The performance of NEHAB1 is better than the others. Similarly, the relative deviation is decreasing with the increase in the number of jobs. This reaffirms the analysis of variance.

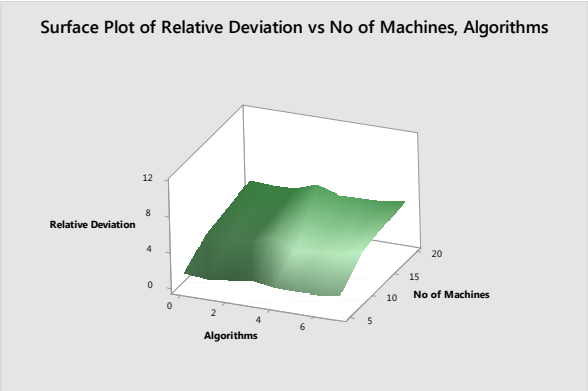

Fig. 9. Surface Plot of Relative Deviation vs No of Machines, Algorithms

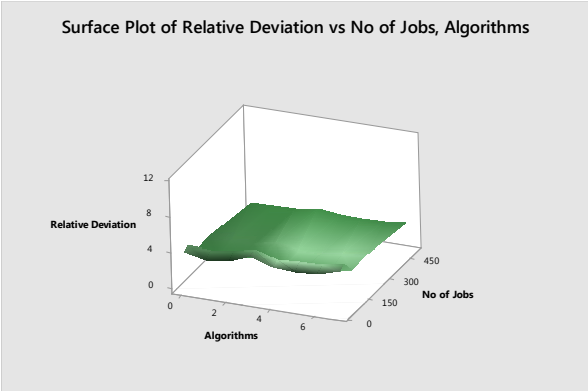

Fig. 10. Surface Plot of Relative Deviation vs No of Jobs, Algorithms

7. Conclusion and Future Work

In this paper, 7 heuristics have been analyzed along with the popular NEH heuristic. The variants are obtained by varying the initial partial sequence. Except in two, the jobs are arranged initially in non increasing order of their total processing times. The average percentage increase in the makespan from the known upper bounds is below 4 % for all the cases which is better than many known simple heuristics. Simply changing the initial partial sequence improves the makespan in three algorithms. It is concluded that, in addition to selecting the first two jobs as the initial partial sequence, other partial sequences also yield better results than the original NEH algorithm. The job insertion technique is the most powerful component of the NEH algorithm followed by the initial arrangement of jobs in non increasing order of

their total processing times. NEH algorithm has been the most popular among the simple heuristics in minimizing the makespan in flow shop scheduling problems. The results obtained using NEH have been used by many meta-heuristics to refine the solution further. Many improvement tools have been proposed for the NEH over the years. Three algorithms among the seven analyzed in this paper are the simpler variants of the NEH with the same complexity level, but yield better results. These may be analyzed further to compare the performance still better. Further research includes the application of the improvement tools proposed by other researchers for the NEH algorithm in these algorithms also and analyzing the effects for different benchmark problem instances.

References

Ancău, M. (2012). On solving flowshop scheduling problems. *Proceedings of the Romanian Academy. Series A*, *13*(1), 71-79

Baskar, A., & Xavior, M. (2013). Optimization of makespan in flow shop scheduling problems using combinational NEH family of heuristics-An analysis. *International Journal of Applied Engineering Research*, *8*(10), 1205-1217.

Campbell, H. G., Dudek, R. A., & Smith, M. L. (1970). A heuristic algorithm for the n job, m machine sequencing problem. *Management Science*, *16*(10), B-630.

Caraffa, V., Ianes, S., Bagchi, T. P., & Sriskandarajah, C. (2001). Minimizing makespan in a blocking flowshop using genetic algorithms. *International Journal of Production Economics*, *70*(2), 101-115.

Chakraborty, U. K., & Laha, D. (2007). An improved heuristic for permutation flowshop scheduling. *International Journal of Information and Communication Technology*, *1*(1), 89-97.

Chen, C. L., Vempati, V. S., & Aljaber, N. (1995). An application of genetic algorithms for flow shop problems. *European Journal of Operational Research*, *80*(2), 389-396.

Dannenbring, D. G. (1977). An evaluation of flow shop sequencing heuristics. *Management Science*, *23*(11), 1174-1182.

Dong, X., Huang, H., & Chen, P. (2008). An improved NEH-based heuristic for the permutation flowshop problem. *Computers & Operations Research*, *35*(12), 3962-3968.

Erdoğmuş, P. (2010). Particle swarm optimization performance on special linear programming problems. *Scientific Research and Essays*, *5*(12), 1506-1518.

Fernandez-Viagas, V., & Framinan, J. M. (2014). On insertion tie-breaking rules in heuristics for the permutation flowshop scheduling problem. *Computers & Operations Research*, *45*, 60-67.

Framinan, J. M., Leisten, R., & Rajendran, C. (2003). 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*, *41*(1), 121-148.

Gantt, H. L. (1919). *Organizing for work*. Harcourt, Brace and Howe.

Gajpal, Y., & Rajendran, C. (2006). An ant-colony optimization algorithm for minimizing the completion-time variance of jobs in flowshops. *International Journal of Production Economics*, *101*(2), 259-272.

Garey, M. R., Johnson, D. S., & Sethi, R. (1976). The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research*, *1*(2), 117-129.

Gupta, A., & Chauhan, S. (2015). A heuristic algorithm for scheduling in a flow shop environment to minimize makespan. *International Journal of Industrial Engineering Computations*, *6*(2), 173-184.

Gupta, J. N. (1971). A functional heuristic algorithm for the flowshop scheduling problem. *Operational Research Quarterly*, 39-47.

Hundal, T. S., & Rajgopal, J. (1988). An extension of Palmer's heuristic for the flow shop scheduling problem. *The International Journal Of Production Research*, *26*(6), 1119-1124.

Ishibuchi, H., Misaki, S., & Tanaka, H. (1995). Modified simulated annealing algorithms for the flow shop sequencing problem. *European Journal of Operational Research*, *81*(2), 388-398.

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

Kalczynski, P. J., & Kamburowski, J. (2007). On the NEH heuristic for minimizing the makespan in permutation flow shops. *Omega*, *35*(1), 53-60.

Kalczynski, P. J., & Kamburowski, J. (2008). An improved NEH heuristic to minimize makespan in permutation flow shops. *Computers & Operations Research*, *35*(9), 3001-3008.

Koulamas, C. (1998). A new constructive heuristic for the flowshop scheduling problem. *European Journal of Operational Research*, *105*(1), 66-71.

Land, A. H., & Doig, A. G. (1960). An automatic method of solving discrete programming problems. *Econometrica: Journal of the Econometric Society*, 497-520.

Liu, G., Song, S., & Wu, C. (2012). Two techniques to improve the NEH algorithm for flow-shop scheduling problems. In *Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence* (pp. 41-48). Springer Berlin Heidelberg.

Murata, T., Ishibuchi, H., & Tanaka, H. (1996). Genetic algorithms for flowshop scheduling problems. *Computers & Industrial Engineering*, *30*(4), 1061-1071.

Nagano, M. S., & Moccellin, J. V. (2002). A high quality solution constructive heuristic for flow shop sequencing. *Journal of the Operational Research Society*, *53*(12), 1374-1379.

Nawaz, M., Enscore, E. E., & Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, *11*(1), 91-95.

Nowicki, E., & Smutnicki, C. (1996). A fast tabu search algorithm for the permutation flow-shop problem. *European Journal of Operational Research*, *91*(1), 160-175.

Ogbu, F. A., & Smith, D. K. (1990). The application of the simulated annealing algorithm to the solution of the n/m/C max flowshop problem. *Computers & Operations Research*, *17*(3), 243-253.

Onwubolu, G., & Davendra, D. (2006). Scheduling flow shops using differential evolution algorithm. *European Journal of Operational Research*, *171*(2), 674-692.

Osman, I. H., & Potts, C. N. (1989). Simulated annealing for permutation flow-shop scheduling. *Omega*, *17*(6), 551-557.

Palmer, D. (1965). Sequencing jobs through a multi-stage process in the minimum total time--a quick method of obtaining a near optimum. *OR*, 101-107.

Pinedo, M. L. (2012). *Scheduling: Theory, Algorithms, and Systems*. Springer Science & Business Media.

Ponnambalam, S. G., Aravindan, P., & Chandrasekaran, S. (2001). Constructive and improvement flow shop scheduling heuristics: an extensive evaluation. *Production Planning & Control*, *12*(4), 335-344.

Pour, H. D. (2001). A new heuristic for the n-job, m-machine flow-shop problem. *Production Planning & Control*, *12*(7), 648-653.

Rajendran, C. (1993). Heuristic algorithm for scheduling in a flowshop to minimize total flowtime. *International Journal of Production Economics*, *29*(1), 65-73.

Reza Hejazi*, S., & Saghafian, S. (2005). Flowshop-scheduling problems with makespan criterion: a review. *International Journal of Production Research*, *43*(14), 2895-2929.

Ribas, I., & Mateo, M. (2010). Improvement tools for NEH based heuristics on permutation and blocking flow shop scheduling problems. In *Advances in Production Management Systems. New Challenges, New Approaches* (pp. 33-40). Springer Berlin Heidelberg.

Reeves, C. R. (1995). A genetic algorithm for flowshop sequencing. *Computers & Operations Research*, *22*(1), 5-13.

Ronconi, D. P. (2004). A note on constructive heuristics for the flowshop problem with blocking. *International Journal of Production Economics*, *87*(1), 39-48.

Ruiz, R., & Maroto, C. (2005). A comprehensive review and evaluation of permutation flowshop heuristics. *European Journal of Operational Research*,*165*(2), 479-494.

Sarin, S., & Lefoka, M. (1993). Scheduling heuristic for the n-jobm-machine flow shop. *Omega*, *21*(2), 229-234.

Sayadi, M., Ramezanian, R., & Ghaffari-Nasab, N. (2010). A discrete firefly meta-heuristic with local search for makespan minimization in permutation flow shop scheduling problems. *International Journal of Industrial Engineering Computations*, *1*(1), 1-10.

Semančo, P., & Modrák, V. (2012). A Comparison of Constructive Heuristics with the Objective of Minimizing Makespan in the Flow-Shop Scheduling Problem. *Acta Polytechnica Hungarica*, *9*(5).

Singhal, E., Singh, S., & Dayma, A. (2012). An Improved Heuristic for Permutation Flow Shop Scheduling. In *NEH ALGORITHM). International Journal of Computational Engineering Research*, *2*(6), 95-100.

Suliman, S. M. A. (2000). A two-phase heuristic approach to the permutation flow-shop scheduling problem. *International Journal of Production Economics*,*64*(1), 143-152.

Taillard, E. (1990). Some efficient heuristic methods for the flow shop sequencing problem. *European journal of Operational research*, *47*(1), 65-74.

Taillard, E. (1993). Benchmarks for basic scheduling problems. *European Journal of Operational Research*, *64*(2), 278-285.

Widmer, M., & Hertz, A. (1989). A new heuristic method for the flow shop sequencing problem. *European Journal of Operational Research*, *41*(2), 186-193.

Xiao-ping, L., Yue-Xuan, W., & Cheng, W. (2004, June). Heuristic algorithms for large flowshop scheduling problems. In *Intelligent Control and Automation, 2004. WCICA 2004. Fifth World Congress on* (Vol. 4, pp. 2999-3003). IEEE.

Yin, M., Zou, T., & Gu, W. (2010). Reverse Bridge Theorem under Constraint Partition. *Mathematical Problems in Engineering*, *2010*.