New sequences of processing times for Johnson's algorithm in PFSP

Shahriar Farahmand Rad
Department of Mathematics
Payame Noor University
P.O. Box. 19395-3697, Tehran
Iran
sh_fmand@pnu.ac.ir

Abstract. There are many researches about converting n job m machine problem to a n job 2 machine one, and finally using Johnson's rule for minimizing makespan. In one case, this converting leads to the inner product of processing times by Pascal numbers. In this paper, it is shown that there are other suitable numerical sequences with a triangle pattern or without it, producing better makespans in several cases. The quality of results is checked by the benchmark of Taillard in permutation flow shop scheduling problem.

Keywords: flow shop, scheduling, NEH algorithm, stirling numbers, Fibonacci numbers, Bell's numbers, Pascal numbers.

1. Introduction

In flow shop scheduling, the issue is to determine the best sequence of n jobs that are processed on m machines in the same order. Let t_{ij} denote deterministic processing time of job j at machine i, which is a positive integer. It is assumed that all jobs process on every single machine. Makespan or $C_{\max}$ refers to the total time for complete processing of all jobs.

It is usually supposed that all jobs are independent and available. No matter when, each machine processes at most one job and each job is processed only by one machine. No preemption is allowed. Set up times are included in the processing times. Infinite storage buffer between machines is also assumed and machines are available. There are job permutations, which change from machine to machine. Therefore, $(n!)^m$ schedules can be obtained. Having the same permutation for all machines is supposed; hence, $n!$ schedules are possible. The resulting problem is known as the permutation flow shop scheduling problem (PFSP), denoted by Fm/prmu/ $C_{\max}$ Graham et al. [7]. Only the F2/prmu/$C_{\max}$ problem is polynomially, solvable and proposed by Johnson [8]; for $m \geq 3$, the problem is NP-complete Garey et al. [6].

It seems that after several papers in 1950s and then the widespread concern about expansion complexity theory by Karp [9], the great numerical growth of papers was stopped in 1990s. Now, there are few papers about adequate heuristic algorithm for solving deterministic flow shop scheduling problems by minimizing makespan criterion.

For the reason that Johnson's algorithm is exact, authors have hardly tried to convert each arbitrary $n \times m$ PFSP to a 2-machine problem.

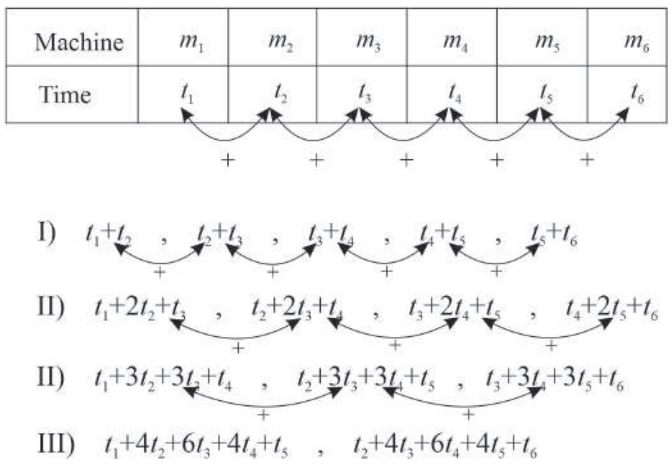
Bonney et al. [4] expressed a job in terms of the slopes of the comulative start and times. It converted the original n jab m machine to a n jab 2 machine problem.

Semanco et al. [11] employed Johnson's rule to present a good initial solution for improving heuristic and the proposed algorithm called MOD. Wei Jia et al. [13] proposed a new algorithm. Firstly, the proposed algorithm normalized the matrix A of processing times. Then it transformed the original problem containing m machines into a 2 machine one that is solved by Johnson's rule. Fernandez- Viagas et al. [5] presented two constructive heuristics based on Johnson's algorithm. Belabid et al. [2] studied the resolution of PFSP that their first method was based on Johnson's rule.

2. The extended Johnson's algorithm

Before explaining the presented algorithm, it is better to make an example about the process of reducing m machine problem into a 2 machine one.

An illustration of this is that $m = 6$ and one job must be processed in $m = 6$ machines with processing times t_1 to t_6. By adding the first two processing times, it is assigned to the first hypothetical machine. It is also continued in a similar way for all jobs and finally the problem was transformed into a $m = 2$ machine, Baskar et al. [1].


Machine m_1 m_2 m_3 m_4 m_5 m_6
Time t_1 t_2 t_3 t_4 t_5 t_6
+ + + + +
I) t_1+t_2 , t_2+t_3 , t_3+t_4 , t_4+t_5 , t_5+t_6
+ + + +
II) $t_1+2t_2+t_3$, $t_2+2t_3+t_4$, $t_3+2t_4+t_5$, $t_4+2t_5+t_6$
+ + +
II) $t_1+3t_2+3t_3+t_4$, $t_2+3t_3+3t_4+t_5$, $t_3+3t_4+3t_5+t_6$
+ +
III) $t_1+4t_2+6t_3+4t_4+t_5$, $t_2+4t_3+6t_4+4t_5+t_6$

It is observed that for m number of machines, the coefficients are the members of Pascal's Triangle for $\binom{m-2}{k}$; $k = 0, 1, 2, \ldots, n$. Indeed, the dot products of these numbers with the original times are obtained. Also at the end, the terms including last and first processing times i.e t_6 , t_1 are respectively omitted.

3. The presented algorithm

In general, suppose the deterministic times for PFSP are t_{ij}; $1 \leq i \leq n$ and $1 \leq j \leq m$. This problem is transformed to a two machine one. The job order obtained from Johnson's algorithm is used to find initial order and calculate the makespan.

Let the time martrix of the initial problem be $M = [t_{ij}]_{n\times m}$ and the time matrix after using Johnson's algorithm be $N = [T_{pq}]_{n\times 2}$. Then, the following relations can be given

$$q{=}1 \Rightarrow T_{p1}{=}(t_{p1}, t_{p2}, t_{p3}, \ldots, t_{pm}) \bullet \left(\binom{m{-}2}{0}, \binom{m{-}2}{1}, \binom{m{-}2}{2}, \ldots, 0\right)$$

$$= \sum_{k=1}^{m} \binom{m-2}{k-1} t_{pk}$$

$$q{=}2 \Rightarrow T_{p2}{=}(t_{p1}, t_{p2}, t_{p3}, \ldots, t_{pm}) \bullet \left(0, \binom{m-2}{0}, \binom{m-2}{1}, \ldots, \binom{m-2}{m-2}\right)$$

$$= \sum_{k=1}^{m} \binom{m-2}{k-2} t_{pk}.$$

The optimal permutation is resulted from Johnson's algorithm on N. This permutation is applied to M. Utilizing Belman et al.'s theorem [3] leads to the minimum makespan.

As was mentioned, inner products of t_{pk}s by Pascal's triangle elements are equal to T_{p1} and T_{p2}. The algorithm is executed on Taillard's problems [12] by Pascal numbers. The triangular neutrality of Pascal's numbers draws attention to the scalar products of the times of Taillard's problems by first and second kind Stirling numbers, Bell's numbers and Fibonacci numbers. These sequences of numbers have triangular pattern du to the next equations.

1) $s_{n+1,k} = s_{nk-1} - n s_{n,k}$, $s_{n,k}$ is a number of first kind Stirling numbers (St 1) that is in n'th row and k'th column in the triangle;

2) $S_{n+1,k} = S_{n,k-1} + k S_{n,k}$, second kind of Stirling numbers (St 2);

3) $f_{n+1} = \binom{n}{0} + \binom{n-1}{1} + \binom{n-2}{2} + \ldots + \binom{n-k}{k}, k = \left[\frac{n}{2}\right]$, f_{n+1} is $n+1$'th term in Fibonacci sequence (Fibo);

4) $\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}, k \geq 1$, Pascal numbers (Pasc);

5) $B_{n+1} = \sum_{k=0}^{n} \binom{n}{k} B_k, B_0 = 1, B_{n+1}$ is $n+1$'th term in Bell's sequence (Bell).

Now, the presented algorithm is divided into three simple steps:

1. Select the first m (m is the number of machines) elements of the above numerical sequences i.e.

i) $s_{m-1,k}, k = 0, 1, 2, \dots, m-1$ (St 1);

ii) $S_{m-1,k}, k = 0, 1, 2, \dots, m-1$ (St 2);

iii) f_{m-1} (Fibo);

iv) $\binom{m-1}{k}, k = 0, 1, 2, \dots, m-1$ (Pasc);

v) $B_k, k = 0, 1, 2, \dots, m-1$ (Bell).

For each stage, inner products of the above elements with the times of original problem are obtained. First, the term concluding t_{pm}, and then the term concluding t_{p_1} are clearly omitted.

2. Johnson's algorithm is applied to give job order from artificial n job and two machine problems with (i),(ii),...,(v) sequences.

3. The job order obtained in previous step is used to find initial order and compute the makespan in original problem.

The algorithm implemented in Visual Basic and carried out all tests on Pentium IV computer at 3.2 GHz with 2 GBytes of RAM memory.

For the statistical analysis, the well known standard benchmark set of Taillard [12] was used. This set includes 120 instances divided into 12 groups with 10 replicates each. The sizes range from 20 jobs, 5 machines to 500 jobs, 20 machines. In the flowshop scheduling literature, this benchmark has been extensively used in the past years. For each instance, a very tight lower bound and upper bound are known. All 10 instances in the 50×20 set , nine in 100×20, six in 200×20 and three in 500×20 are open. For all other instances, the optimum solution is already known.

The applied performance measure that was used, is the Relative parcentage Deviation (RPD) over the optimum or the best solution (upper bound) for each instance:

$$\text{Relative Perentage Deviation (RPD)} = \frac{Heu_{sol} - Best_{sol}}{Best_{sol}} \times 100,$$

where Heu_{sol} is the solution given by any of the tested heuristic for a given instance and $Best_{sol}$ is the optimum solution or the lowest known upper bound for Taillard's instances.

The solutions of presented algorithm are compared with the results of NEH and Taillard's benchmark. NEH was made-up by Nawas et. al. [10], that is the best heuristic that have ever been proposed for solving PFSP [5].

In the following tables, the summary of these comparisons and the results of Talllard's problems are shown. The tables also display the results of NEH.

Table 1. The least RPD for Heuristic Algorithm

Size of	Least RPD Obtained With				NEH	Taillard Upper
Problems	Prob.No	Sequence	Best Makespan	RPD	reaults	Bounds
20×5	2	St 2	1422	4.6	1365	1359
20×10	9	St 2	1848	16	1639	1593
20×20	8	St 2	2473	12.4	2249	2200
50×5	6	Fibo	3093	9.3	2835	2829
50×10	6	Pasc	3728	24	3148	3006
50×20	1	St 2	4762	26.2	4006	3771
100×5	6	Pasc	5740	11.7	5154	5135
100×10	9	St 1	6973	18.7	6016	5871
100×20	10	St 2	7994	24.8	6680	6434
200×10	4	St 2	12880	18.2	11057	10889
200×20	8	St 2	14327	21.1	11824	11334
500×20	5	Pasc	31706	20.3	26928	26334

Table 2. The greatest RPD for Heuristic Algorithm

Size of	Greatest RPD Obtained With				NEH	Taillard Upper
Problems	Prob.No	Sequence	Best Makespan	RPD	results	Bounds
20×5	3	St 2	1349	24.7	1132	1081
20×10	4	Fibo	1856	34.7	1416	1377
20×20	4	Pasc	2749	23.6	2257	2223
50×5	3	St 2	3209	22.4	2650	2621
50×10	3	Pasc	3878	36.5	2994	2839
50×20	4	Pasc	4874	34	3953	3723
100×5	2	Pasc	6171	17.1	5284	5268
100×10	2	Pasc	6795	27	5466	5349
100×20	7	St 1	8135	31.5	6578	6268
200×10	2	St 2	13142	25.4	10677	10480
200×20	3	Pasc	14693	30.2	11724	11281
500×20	10	Pasc	32782	23.9	27103	26457

It is seen that the least RPD is 4.6 which is obtained in Taillard's $20 \times 5 - 2$ problem after the inner product of second kind Stirling numbers. The greatest RPD is 36.5 that is resulted in $50 \times 10 - 3$ problem after the dot product of Pascal numbers.

The best makespan in each instance is shown in the table 1 after the dot product of sequences and comparing with each other. For example in the first section, 7 times second kind Strirling numbers, 1 time Fibonacci numbers, and only 3 times Pascal numbers are resulted the best makespan! In this research, Bell numbers are not resulted this.

The best solutions in the light of quality are those obtained from the inner product of seoond kind Stirling numbers.

For more researches, the Bank of numerical sequences is chosen. This Bank i.e oeis.org includes "The On-Line Encyclopedia Of Integer Sequences" found by N.J.A. Sloane. He has worked the sustainable collection of these sequences since 1964.

At another time, the algorithm implemented in Python and carried out all tests on a Quad-Core Intel Core i7 computer at 2.6 GHz with 16 GBytes of RAM memory. In 10 hours, first 100000 sequences were chosen and scalar products of them were determined. Then Johnson's algorithm found initial order of jobs. The average of ten obtaining makespans in each package of Taillard instances was calculated, afterward 100000 solutions in them were collected in following figures.

In these figures, l is the average of lower bounds or the average of solution; and u is the average of upper bounds in each package of Taillard's instances.

Additionally, X-axis shows the number of sequences, and the result of each correspondent sequence is a point in the direction of Y-axis. It is regarded that the specified average points have not good situation with respect to l and u.

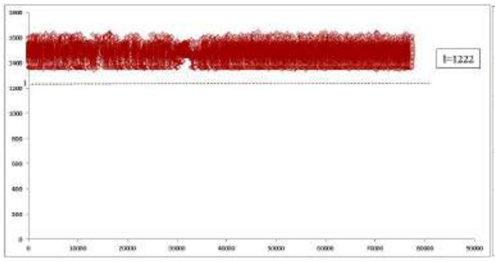


Figure 1. average results of 20×5

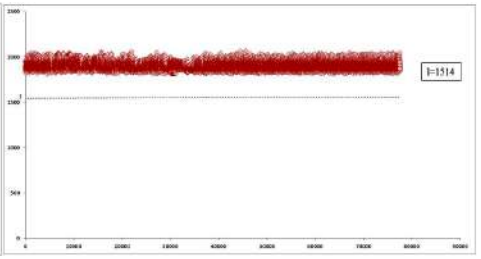


Figure 2. average results of 20×10

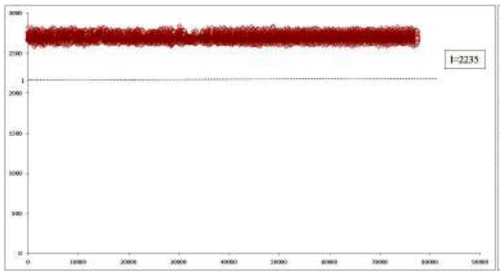


Figure 3. average results of 20×20

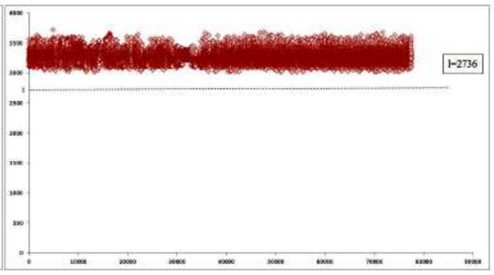


Figure 4. average results of 50×5

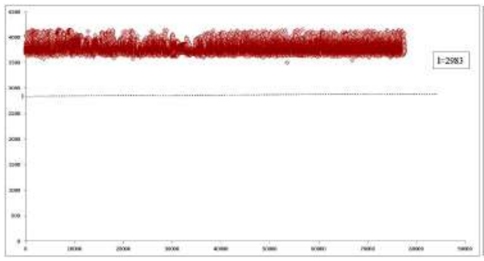


Figure 5. average results of 50×10

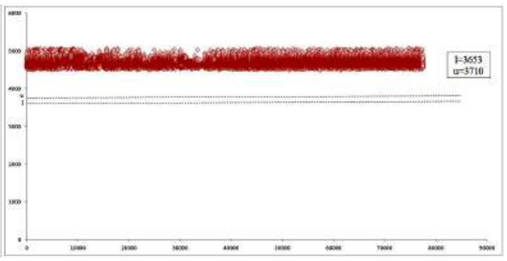


Figure 6. average results of 50×20

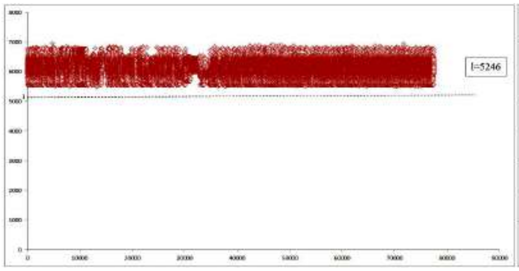


Figure 7. average results of 100×5

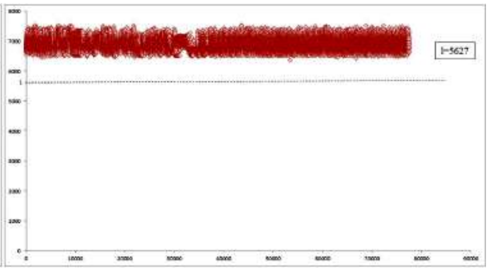


Figure 8. average results of 100×10

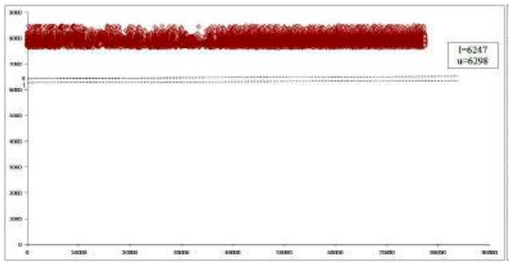


Figure 9. average results of $100{\times}20$

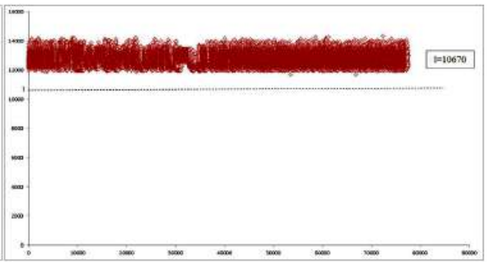


Figure 10. average results of $200{\times}10$

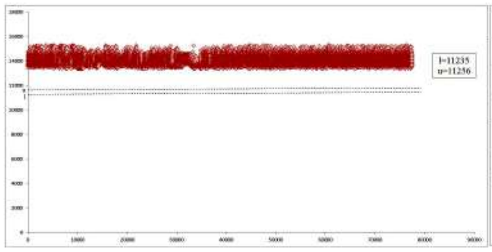


Figure 11. average results of $200{\times}20$

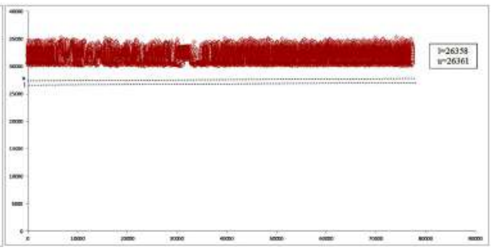


Figure 12. average results of $500{\times}20$

In all 120 instances of Taillard's and 120×100000 cases, there is only one result that is the same of Taillard's solution after fixing all steps and running Johnson's algorithm.

This is $100 \times 5 - 1$ problem and the solution is 5493. The result is consequent of the inner product of the original processing times by the sequence A088661 with the general term:

$$a_n = \sum_{k=1}^{8} \left[\frac{P_{n,k}}{P_{n-1,k}} \right]$$

when

$$P_{n,k} = \frac{\sum\limits_{i=1}^{n} \log i}{\sum\limits_{i=n-\left[\frac{3n}{4^k}\right]}^{n-\left[\frac{n}{4^k}\right]} \log i}$$

namely "A log based cantor self similar sequence" (bracket is floor). The author is Roger L. Bagula, Nov 21 2003. For $n = 3, 4, \ldots, 107$ the terms of this sequence are $8, 8, 7, 6, 7, 8, 8, 7, 6, 8, 8, 7, 7, 8, 8, 7, 7, 8, 8, 5, 7, 8, 8, 7, 6, 8, 8, 7, 7, 8, 8, 7, 7,$ $8, 8, 6, 7, 8, 8, 7, 5, 8,$ $8, 7, 7, 8, 8, 7, 7, 8, 8, 6, 7, 8, 8, 7, 6, 8, 8,$ $7, 7, 8, 8, 7, 7, 8, 8, 6, 7, 8, 8, 7, 6, 8, 8, 7, 7,$ $8, 8, 7, 7, 8, 8, 4, 7, 8, 8, 7, 6, 8, 8, 7, 7, 8, 8, 7, 7, 8, 8, 6, 7, 8, 8, 7, 5.$

A brief research is denoted that after each 4-term section, four numbers 7, 8, 8, 7 are repeated.

For the intuitive perception of sequence's behaviour, its pin plot and scatter plot are designed in the following.

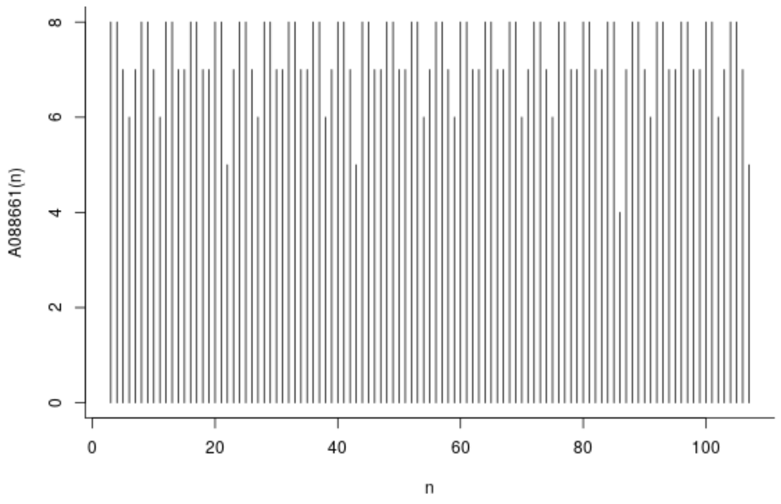


Figure 13. Pin plot of A088661(n)

4. Conclusions

In this paper, it was tried to transform the $n \times m$ problem to a $2 \times m$ problem after obtaining the inner product of proessing times in PFSP by the most famous numerical sequences. Johnson's algorithm was used and the results were compared with those of the Taillard's 120 problem solutions. The least relative percentage deviation was obtained in $20 \times 5 - 2$ Taillard's problem with the dot product of Stirling second kind numbers. Moreover, the greatest RPD

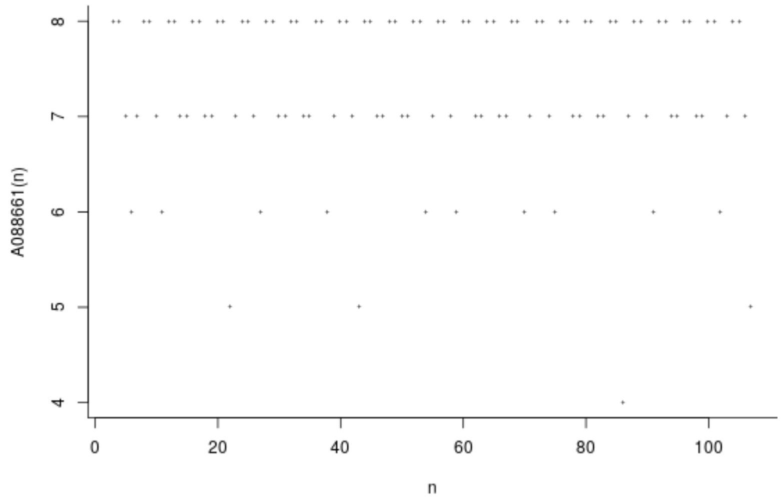


Figure 14. Scatterplot of A088661(n)

was resulted in $50 \times 10 - 3$ Taillard's problem. All these arguments lead to the conclusion that Baskar's ideas [1] about good solutions of the inner product of Pascal numbers have been invalidated.

After obtaining the dot product of 100000 different numerical sequences in processing times for $100 \times 5 - 1$ instance, Johnson's rule results optimum solutions.

References

[1] A. Baskar, M.A. Xavior, *A new Heuristic algorithm using Pascal's triangle to determine more than one sequence having optimal/near optimal make span in flow shop scheduling problems*, International Journal of Computer Applications, 975 (2012), 8887.

[2] J. Belabid, S. Aqil, K. Allali, *Solving permutation flow shop scheduling problem with sequence-independent setup time*, Journal of Applied Mathematics, 2020.

[3] R. Bellman, A.O. Esogbue, I. Nabeshima, Mathematical aspects of scheduling and applications: modern applied mathematics and computer science, Elsevier, vol. 4, 2014.

[4] M.C. Bonney, S.W. Gundry, *Solutions to the constrained flowshop sequencing problem*, Journal of the Operational Research Society, 27 (1976), 869-883.

[5] V. Fernandez-Viagas, J.M. Molina-Pariente, J.M. Framinan, *New efficient constructive heuristics for the hybrid flowshop to minimise makespan: a*

computational evaluation of heuristics, Expert Systems with Applications, 114 (2018), 345-356.

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

[7] R.L. Graham, E.L. Lawler, J.K. Lenstra, A.R. Kan, *Optimization and approximation in deterministic sequencing and scheduling: a survey*, In Annals of Discrete Mathematics, Elsevier, 5 (1979), 287-326.

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

[9] R.M. Karp, *Reducibility among combinatorial problems*, In R. E. Miller and J. W. Thatcher (eds.) Complexity of computer computations, Plenum Press, New York, 1972, 85-104.

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

[11] P. Semančo, V. Modrák, *A comparison of constructive heuristics with the objective of minimizing makespan in the flow-shop scheduling problem*, Acta Polytechnica Hungarica, 9 (2012), 177-190.

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

[13] J.Y. Wei, Y.B. Qin, D.Y. Xu, *DRPFSP algorithm for solving permutation flow shop scheduling problem*, Computer Science, 2015.

Accepted: February 10, 2021