TECHNICAL UNIVERSITY OF CLUJ-NAPOCA

ACTA TECHNICA NAPOCENSIS

Series: Applied Mathematics, Mechanics, and Engineering
Vol. 66, Issue Special I, September, 2023

MULTI-FACTORY SCHEDULING FOR A CORPORATED SUPPLY CHAIN

Yusaku TAHARA, Kosuke NAGAI, Sumika ARIMA

Abstract: *This paper introduces the n-step hybrid flow-shop scheduling (nHFS) for a corporated supply chain in which component assembly and final assembly factories are linked in series (tandem-type). We applied n-GuptaEX=SETUPBO method (Mao et al., 2022) which is advanced form of one of representative nHFS solution proposed by J.N.D Gupta et al. (2002). As a baseline, n-GuptaEX-SETUPBO method improved both the optimization level and the computational efficiency much in our previous study. Now, for the case of multi-factory, each factory has a different utility, and there is a trade-off relationship between them. The purpose of this study is to improve the performance of the entire supply chain through integrated scheduling that balance the interests of each factory. Its scheduling performance is evaluated by comparing it to other existing approaches. Discrete event simulation is used in numerical experiments.*

Key words: *Multi-objective Problem, Hybrid flow-shop, Scheduling, Corporated supply chain, Multi-factory*

1. INTRODUCTION

In recent years, labor shortages in the manufacturing industry have become increasingly serious in Japan: according to a December 2017 survey conducted by the Ministry of Economy, Trade and Industry (METI), more than 94% of companies reported that "labor shortage issues are becoming apparent," an increase of about 10% for the past four consecutive years. In addition, 32% of these companies said that "business is also being affected" [1].

In order for Japan to achieve sustainable economic growth, it is essential to create an environment in which a diverse range of human resources can play an active role and to increase the labor productivity of each individual. To this end, it is important to promote digitalization[2] and to create information systems that can be used by companies and small and medium-sized enterprises (SMEs) with limited knowledge of digitalization.

This study focuses on two hypothetical factories, which are characterized by two points. The first is that the two factories are located in series as part of a supply chain. Although both factories have indicators that have a trade-off relationship, integrated scheduling is oriented toward improving the performance of the entire supply chain.

The second point is that each factory is an n-process hybrid flow-shop model, where the sequence of processes to produce a product is identical and each process consists of multiple machines and personnel. n-Gupta et al. in their study on n-process hybrid flow-shop scheduling (HFS) developed the n-Gupta method [3], which has a particularly high performance compared to other scheduling methods due to its objective function (1). This objective function includes four indices that trade-off each other for scheduling purposes: loss E when a job is completed earlier than its due date, loss T when a job is delayed, completion time C, and due date d. Tanaka et al.

Tanaka et al. proposed an extended n-Gupta method (nGuptaEX) with loading and work leveling methods for MTO-MTS mixed production systems [4]. Furthermore, Bayesian optimization (BO) is applied to improve the nGuptaEX solution by automatically adjusting the multi-criteria parameters. Real-world examples were validated to prove the efficiency and accuracy of the BO-based n-Gupta method (n-GuptaEX-BO).

Lin and Ohno, et.al. proposed a heuristic algorithm for batch processing that outperforms the usual batch loading rules and reduces the number of batches (e.g., by less than half) while maintaining the deadline [5]. This batch processing rule can be flexibly combined with n-GuptaEx-BO.

To further improve the computational efficiency of n-GuptaEx, Mao et al. improved the tabu search method and local search with a pooling mechanism, and applied Self-Tuning Portfolio-based BO (SETUP-BO)[6] for parameter tuning to n- GuptaEX=SETUPBO method was proposed[7].

$$F = \sum_{i=1}^{n} u_i E_i + v_i T_i + w_i C_i + z_i d_i \quad (1)$$

These methods have been demonstrated for single factories and have shown better performance than other scheduling methods. However, they have not been studied for series-connected factories, which is a matter of debate. The objective of this study is to improve the performance of the entire supply chain through integrated scheduling that considers the interests of each factory.

2. PROBLEM DESCRIPTION AND PREVIOUS WORKS

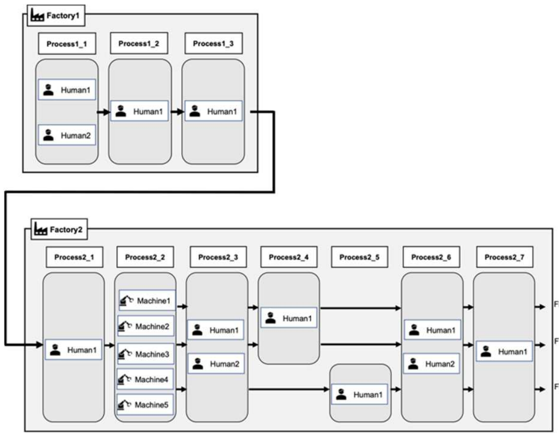

Figure 1
Actual problem setting

The supply chain in this study, as shown in Figure 1, consisted of a three-step hybrid flow-shop factory and a seven-step hybrid flow-shop factory, including batch processes, connected in series. In each process, a predetermined number of machines or personnel are deployed based on conditions. Process 2_1 and 1 Process 2_2 are batch-type processes, while the other processes are lot-type processes.

The scheduling problem is complex because different products have different flows (processing steps and available machines). Each product can only use a given range of machines and personnel pre-assigned to each process. Both machines and personnel vary in capacity and conditions, in addition to variations in product mix demand, so that one machine or personnel can be shared by different products.

To search for an optimal or near-optimal solution to nHFS with batch processing, we decomposed the problem into two subproblems: job sequencing and job batching. To solve these problems, we developed the n-GuptaEX-SETUPBO method for lot and batch sequencing and the variable-time window batching heuristic to create batches by lot in previous work.

2.1 n-Gupta method, n-GuptaEX-BO method and n-GuptaEx-SETUPBO method

To solve the job ordering problem, the n-Gupta method[3] can assign arbitrary weights to the multiobjective function for the actual process.

The objective function of original n-Gupta method is defined in (1). As the description in Section 1, E_i, T_i respectively are earliness penalty and tardiness penalty of job i. Ci is the complete time of job i, and d_i is due date decided by factory of job i. They are defined by the following formula.

$$E_i = \{0, d_i - C_i\} \quad (2)$$
$$T_i = \{0, C_i - d_i\} \quad (3)$$
$$d_i = mi\{r_i + TI \cdot \sum_{j=1}^{n} t_{ij}, d_i^0\} \quad (4)$$

Here, r_i is release time of job $i. t_{ij}$ is processing time of the j's process of job i. di0 is due date from the order receipt data and *TI* is a tightness parameter for determining the process' due date based on the number of required hours. The smaller *TI*, the more difficult it is to complete the job by due date.

The n-Gupta method uses iterative algorithms based on local search that applied some neighborhood structure to improve the current initial solution iteratively. Some basic moves (Insertion and Exchange operations, Tightness parameter shifts) are used in n-Gupta iterative algorithm (Algorithm1) for generating a neighbor.

```
**Algorithm1**: n-Gupta method
Step1: Determine the deadline $d_i$ based on the tightness parameter $TI$. The insertion order is $IO: = i_1, i_2, \dots i_n$ as $i_1, i_2, \dots i_n$ from the smallest $i$ of $d_i$ . Initialize with partial order $PO: = \{\}$, partial order length $l = 0$.
Step2: $While\ IO \neq \varphi$
    I.   Select the top $i$ of $IO$ and remove it from $IO$.
    II.  $F^* := \infty\ , k^* := mik, l + 1$
    III. $For\ j: = 1\ to\ k^*$
        1. $u := l - k^* + j + 1$
        2. Let $PO'$ be job $i$ is inserted partial order $PO$.
        3. Determine the allocation to the machine according to $PO'$. In the process with a plurality of machines, jobs are assigned to the machine where the previous processing ends the earliest. In each process, $PO'$ is not ignored and the process is not started.
        4. Calculate the evaluation value $F(PO')$.
        5. If $F\ (PO') < F^*$
    $F *: = F\ PO', PO'': = PO'$
    IV.  $PO: = PO'', l = l + 1$
Step3: Processing order $O: = PO$
Step4: In the schedule based on the processing order $O$ obtained in Step3, if $C_i - d_i > B$ is satisfied in any job $i$, check whether $TI$ improves the evaluation value $F$. $TI^0 := TI$, continuously considering values such as $TI^k = TI^0 + \Delta TI \cdot k, k$ is a natural number, and using a minimum $TI^k$ with a new $TI$. Even if $k$ is increased, $d$ does not increase, or $k$ changes in a range satisfying $C_i - d_i > B$.
Step5: The processing order $O$, and the schedule obtained from the final tightness parameter $TI$ are used as an initial solution.
```

In n-GuptaEx-BO method, we described improvement of solution by iterative method (local search method). The solution is improved by changing the job processing order O and the tightness parameter TI. Jobs to be replaced are performed between s or less adjacent jobs. The extended n-Gupta method(n-GuptaEx) is as shown in Algorithm2.

```
**Algorithm2**: extended n-Gupta method
Step1: From the range of $[TI - \delta 1, TI]$ with probability $prdec\ (\leqq 1)\ TI'$ is determined randomly from the range of $[TI - 1, TI + \delta 2]$ with probability $1 - prdec$.
Step2: Calculate the deadline $d_i$ based on $TI'$. Choose a job $i$ of $C_i \neq d_i$.
Step3: Select job $j$ from a job with earlier processing order if $C_i > d_i$ (which means the completion is later than deadline), or from a job with later processing order if $C_i < d_i$ (which means the completion is earlier than deadline). Job $j$ is selected from jobs in the adjacent order within $\boldsymbol{i}$ to $\boldsymbol{s}$.
Step4: Swap the positions of job $\boldsymbol{i}$ and job $\boldsymbol{j}$ in processing order $O$.
Step5: Change $\boldsymbol{TI}$ in the same way as in Step4.
```

We also improved the weighting parameter settings of objective function by applying four parameters (u, v, w, z) to discuss its characteristic details and optimization at each flow instead of each job (in (5)).

$minimize\ F = \sum_{i=1}^{n} (uE_i + vT_i + wC_i + wC_i z d_i)$ (5)

To optimize these interfering parameters, we applied Bayesian optimization (BO) which can tune the parameters automatically without domain knowledge implementation.

n-GuptaEX-SETUPBO improves on the tabu search method and local search with a pooling mechanism, and applies Self-Tuning Portfolio-based BO (SETUP-BO) for parameter tuning, which further improves the computational efficiency of the previous study.

The features of the local search with tabu search method and pooling mechanism are as follows. Based on Algorithm 1, the initial solution pooling approach is applied to avoid duplicate solutions and improve solution diversity. Specifically, after generating the initial solution each time, the objective function value F is calculated and the ascending set SO is created from the F value of the initial solution. When performing Bayesian optimization, the initial solution is selected from the pool and calculated. The selection criteria is to select the best initial solution with a probability of $0 < PM < 1$ and to randomly select from the inferior solutions with a probability of $1 - PM$.

The iterative method was also improved to avoid reruns by applying the tabu search method. Furthermore, a new probabilistic selection strategy was proposed to refine the solution diversity and the efficiency of the iterative method. The procedure is shown in Algorithm 3.

```
Algorithm3: Tabu Search Based Iterative Method

Step1: Determine the $TI'$ by choosing randomly a value from the interval $[TI - 1, TI]$ with probability $0 \leq prdec \leq 1$. Calculate the due date $d_i$ based on $TI'$. Collect the initial solution set $S'$ from the pool, reorder $S'$ with ascending order of $C_i - d_i$ to form set $X$.
Step2: Create the tabu list $L$. Define the stopping number of iterations $Runtimes$, length of tabu list $Length_{tb}$ and rule probability parameter $RI$.
Step3: Collect the jobs which meet the condition $C_i \neq d_i$ from $X$, and form the candidate list $P$ with the collected jobs.
Step4: Choose randomly a value $p$ from the interval [0,1]. If $0 < p \leq RI$, apply rule A. If $RI < p$, apply rule B.
    Rule A: From the candidate list $P$, choose the job (job $i$) that has the biggest $C_i - d_i$, and the job (job $j$) that has the smallest $C_i - d_i$.
    Rule B: Choose randomly two jobs (job $i$ and job $j$) from $k$ adjacent jobs in the candidate list $P$.
Step5: If the pair of job $i$ and job $j$ in $L$, move to Step6. Otherwise, move to Step7
Step6: Delete job $i$ and job $j$ from $P$, return to Step4.
Step7: Swap the order of job $i$ and job $j$. Calculate $F(S')$ with the new order $S'$. If $F(S') < F^*, F^* = F(S')$, upgrade $X, P, L$, return to Step4. Otherwise, return the previous $S'$ and return to Step4.
Step10: Repeat Step4~Step7 until number of iterations $<$ $Runtimes$ and all jobs meet the condition $C_i - d_i < B$.
```

In tuning the weighting parameters of n-GuptaEx, we apply a portfolio-based method that allows Bayesian optimization to select better acquisition strategies. The procedure is shown in Algorithm 4.

```
Algorithm4: Self-Tuning Portfolio-based BO

Step1: Set $G_j(0) = 0 \; for \; j = 1,2,\dots,J$.
Step2: Sample the hyperparameter $\eta \sim$ Gamma $(\alpha, \beta)$ and $m \sim$ Beta $(a, b)$. Nominate points from each acquisition function $h_j$:  $x_j(t) = argh_j(x)$
Step3: Compute $r\_\min(t-1) = (G_j(t-1)) \; and \; r_{max}(t-1) = G_j(t-1)$, then compute the normalized rewards:
    $$r_j(t-1) = \frac{G_j(t-1) - r_{min}(t-1)}{r_{max}(t-1) - r_{min}(t-1)}$$
Step4: Select a nominee $\boldsymbol{x}(t) = x_j(t)$ with probability $p_j(t) = \frac{ex(\eta r_j(t-1))}{\sum_{j=1}^{J} ex(\eta r_j(t-1))}$
Step5: Compute $y_t$ by evaluating the objective on point $x(t)$.
Step6: Augment the data $D_t$ with the new pair $(x(t), y(t))$.
Step7: Update the surrogate GP model.
Step8: Update the rewards $G_j(t) = mG_j(t-1) - \mu(x_j(t))$ from the updated GP posterior.
Step9: Update the posteriors $a = a + 1, if \; y(t)$ is the best point evaluate so far, otherwise update $b = b + 1$.
Step10: Update the posterior $\alpha \leftarrow \alpha + 1$, and update the posterior $\beta \leftarrow \beta + x(t)$
Step11: Repeat Step3~Step13 until stopping criterion is reached.
```

Here, h_j represents acquisition function of strategy j. In our research, the chosen strategies are Probability of Improvement (PI), Expected Improvement (EI) and Lower Confidence Bound (LCB). $y(t)$ represents objective function, and in this research, it can be calculated by (6). Here *FA* is non-weighted performance except d of the objective function (1). This is because d depends on the parameter *TI*, and *FA* is more important for the actual factory performance.

$$FA = \sum_{i=1}^{n}(E_i + T_i + C_i) \quad (6)$$

2.2 Batch Heuristic

To cope with batch processing, we proposed a batch heuristic as shown in Algorithm 5, formed batches, and calculated the job release date in (7).

$$R_i = D_i - k_i \sum_{i,j=1}^{n} P_{ij} \qquad (7)$$

Here, R_i is the latest release date and D_i is the due date of job i. k_i means the slack coefficient to adjust the suitable release date. p_{ij} is the processing time of job i in process j.

Algorithm5: Batch Heuristic

Step1: Sort all the lots in descending order based on the due date.

Step2: Select the top one as lot A according to the order that determined in Step1 and calculate the lot A's latest release dat$e$ R_i $with$ (7).

Step3: According the lot A's pressure condition, determine the machine which has the largest capacity among the available machines as the batch size. Also, only the lot with the same pressure condition as lot A has can be selected in the next steps.

Step4: Choose the lot that satisfies the batch composition conditions from the remaining lots that are determined in Step3. If the lot's R_i is within a range of x_1 days before and after the lot A's R_i and the lot's D_i is within a range of x_2 days before and after lot A's D_i, the lot can be forming a batch with the lot A selected in Step2.

Step5: Put all the lots selected in Step4 in a batch with lot A in Step2 until reach the upper limit of the batch size.

Step6: Adjust the machine limit. If the batch size that formed in Step5 is equal to the upper limit of the batch size in Step3, use the machine that has been determined in Step3. If the batch size is smaller than the upper limit and smaller than the machine which have the second largest capacity, use the second largest. The same as the other machines. If the batch size is just smaller than the upper limit while larger than the second largest, use the machine selected in Step3.

Step7: Remove the lot that already in the batch from the pre-selected queue.

Step8: Repeat Step2~Step7 until all lots are in the batches.

The following are the results of scheduling with n-GuptaEX-SETUPBO and Batch Heuristic for Factory 2, using the due date as the starting point, and the dispatching and loading rules. Comparison shows that the former greatly improves Average Earliness and Average cycle time.

Table 1

Comparison of Due-date satisfactiom rate of each method for Factory2

	Due-date satisfactiom rate [%]			
	Normal		Overload	
	FLR	PA	FLR	PA
FIFO	100	100	100	100
EDD	100	100	100	100
CR	100	100	100	100
Slack	100	100	100	100
Factory2	72.78		84.92	

Table 2

Comparison of Average tardiness of each method for Factory2

	Average tardiness [days]				consideration of procurement possibility
	Normal		Overload		
	FLR	PA	FLR	PA	
FIFO	0	0	0	0	×
EDD	0	0	0	0	×
CR	0	0	0	0	×
Slack	0	0	0	0	×
Factory2	11.59		20.2		×

Table 3

Comparison of Average Earliness of each method for Factory2

	Average Earliness [days]			
	Normal		Overload	
	FLR	PA	FLR	PA
FIFO	67.79	67.47	132.78	131.31
EDD	67.79	67.47	132.92	131.01
CR	67.64	67.42	132.46	131
Slack	67.68	67.23	133.09	131.1
Factory2	11.48		14.12	

Table 4

Comparison of Average cycle time of each method for Factory2

	Average cycle time [days]				consideration of procurement possibility
	Normal		Overload		
	FLR	PA	FLR	PA	
FIFO	10.62	10.91	8.55	10.02	×
EDD	10.62	10.92	8.42	10.31	×
CR	10.76	10.96	8.88	10.32	×
Slack	10.72	11.15	8.25	10.22	×
Factory2	8.64		8.16		×

Table 5

Instruction of Dispatching rules

Dispatching rules	Instructions
FIFO(First In First Out)	Process in the order in which they arrive at the process first.
EDD(Earliest Due Date)	Process in the order of nearest due date.
CR(Critical Ratio)	Process in order of smallest Critical Ratio. If the due date is missed Critical Ratio = (Remaining time until due date[min]/1440) * (Remaining work time[min]/1440) * 100 If the due date is on time Critical Ratio = (Remaining time until due date[min]) / (Remaining work time[min]) * 10000
Slack	Process in order of smallest Slack time. (Slack time = Remaining time until due date - Total remaining work time)

Table 6

Instruction of Loading rules

Loading rules	Instructions
FLR(Full Load Required)	Start processing from lots waiting in the quene that meet the Load Size. No processing is performed until the lot that meets Loas Size appears or waiting time reaches the limit time of batch process.
PA(Partial Load Allowed)	Start processing from lots waiting in the quene with a higher Processing priority. Processing can be started even if the Load Size is not satisfied.

3. PROPOSED METHOD

The effectiveness of n-GuptaEX-SETUPBO and Batch Heuristic has been proven in previous studies, but there are still issues to be addressed.

Past studies of these methods have focused on a single factory, and research on multiple factories in series has not progressed.
Each factory has a different utility, and trade-off exist between them.

In addition, they assume the case where production starts after receiving an order, and when the time between the order date and the due date is short, the on-time delivery rate is reduced.

To solve these problems, this study proposed an integrated scheduling procedure that considers the interests of each factory and assumes a case in which inventory is secured in advance. The procedure is shown in Algorithm 6 and 7.

3.1 Integrated Scheduling Method

Algorithm6: Integrated Scheduling Method

```
Step1: Run the scheduling for factory2 according to Algorithm3, Algorithm4 and Algorithm5 and calculate the schedule.
Step2: Connect the jobs in factory2 with the jobs in factory1.
Step3: $Process1start\ in\ factory2 - 1(day) = due\ date$ in factory1.
Step4: Run the factory1 scheduling according to Algorithm3 and Algorithm4, and calculate the schedule.
Step5: $Process3end\ of\ factory1 + 1(day) = earliest\ possible\ production\ start\ date\ o$
Step6: Reschedule factory2 in the same way as in Step1 and calculate the final schedule.
```

Step 5 gives the constraints that prevent the start of production at Factory 2 from being earlier than the end of the final process at Factory 1.

This method calculates a schedule result that considers the Earliness, Tardiness, cycle time, and due date of the job shown in Equation (1) for both factories.

3.2 Calculation of Possible Production start date

To find the appropriate production start date, the backward allocation method of the pile-up and pile-down method is used. This method distributes the excess production load $D(t,s)$ to the date prior to the date when the total load $L(t,s)$, which is the total of all the production loads of job j at the due date t of job j in each process s, exceeds the production capacity $C(t,s)$ of each process. This method allows the appropriate production start date to be set for order data with unbalanced due dates.

Algorithm7: Calculation of Possible Production start date

```
Step1: Define the number of processes as $S$.
Define total production load of all jobs as $sumL(s)$.
Define the latest due date for jobs as $maxD$.
Define any very early date as $minD$.
Define $countL = 0$.
Step2: $for\ s = S, S-1, \dots, 1$
$While\ sumL(s) > countL$
    1. $For\ t = maxD, maxD-1, \dots, min$D
        1. $While\ L(t,s) > C(t,s)$
        2. $D(t,s) = L(t,s) - C(t,s)$
        3. $L(t,s) = L(t,s) - D(t,s)$
        4. $L(t-1,s) = L(t-1,s) + D(t,s)$
        5. $countL = countL + L(t,s)$
        6. Update the distributed job due date $t$ to $t - 1$.
    2. $if\ L(t,s) \leqq C(t,s)$
        1. $countL = countL + L(t,s)$
        2. $ST = t$
Step3: Determine Possible Production start date to $ST$.
```

4. NUMERICAL STUDY

Two patterns were used in the experiment: normal and overloaded demand for data from real companies. For comparison in the experiment, the order date of the data in question was set to a date close to the due date, and the results of scheduling with Algorithm 6 for hypothetical data with strict constraints on the date when production can begin were used.
The experimental results showed that the Due-date satisfaction rate, Average tardiness, and Average cycle time improved for both 'Normal' and 'Overload' cases. This indicates that the re-setting of the appropriate production start-up date allowed more leeway for production, leading to an increase in these indicators.

On the other hand, Average Earliness worsened. This is because production begins before an order is actually received, which makes it more costly to hold inventory than to produce after receiving an order.

Table 7
Comparison of Due-date satisfaction rate of each method for 2 factories

	Due-date satisfaction rate [%]	
	Normal	Overload
Hypothetical Case	0	24.02
Proposed Method	42.6	85.2

Table 8
Comparison of Average tardiness of each method for 2 factories

	Average tardiness [days]		consideration of procurement possibility
	Normal	Overload	
Hypothetical Case	34.49	46.26	○
Proposed Method	16	24.54	○

Table 9
Comparison of Average Earliness of each method for 2 factories

	Average Earliness [days]	
	Normal	Overload
Hypothetical Case	-	8.81
Proposed Method	4.74	10.55

Table 10
Comparison of Average cycle time of each method for 2 factories

	Average cycle time [days]		consideration of procurement possibility
	Normal	Overload	
Hypothetical Case	10.5	10.68	○
Proposed Method	9.49	8.44	○

5. CONCLUSION

This study proposes an integrated scheduling method that takes into account the interests of each factory for the nHFS problem in a supply chain with two factories connected in series.

The proposed method has the potential to improve the performance of the entire supply chain, as each indicator improved compared to the results obtained by scheduling only Factory 2. In particular, when demand is concentrated, the proposed method is suitable for creating a well-balanced schedule in which no specific lot exceeds its delivery date by a large margin, since the delivery date compliance rate decreased but the delivery delay time was reduced by 42%.

The proposed method succeeded in improving the Due-date satisfaction rate significantly compared to the case where production is started after receiving an order.

On the other hand, Average Earliness worsened due to the start of production in advance. Future discussions should focus on further improving the Due-date satisfaction rate and minimizing the deterioration of Average Earliness.

6. REFERENCES

[1] Ministry of Economy, Trade and Industry, *Current situation of manpower shortage in the manufacturing industry and utilization of foreign human resources*, https://www.meti.go.jp/press/2018/07/20180712005/20180712005- 2.pdf

[2] Ministry of Economy, *Trade and Industry. White Paper on Manufacturing*,

https://www.meti.go.jp/report/whitepaper/mono/2021/pdf/all.pdf

[3] J.N.D. Gupta., K. Krüger., V. Lauff., F. Werner., Y. N. Sotskov., *Heuristics for hybrid flow shops with controllable processing times and assignable due dates*, Computers & Operations Research, pp.1417-1439, Elsevier Science Ltd, 2002.

[4] M. Tanaka., K. Nishizawa., T. Ohno., Y. Ogawa., S. Arima., *Applications on hybrid flow-shop scheduling under dynamic constraints of queue time and capacities*, Proceedings of Joint Symposium of e-Manufacturing and Design Collaboration Symposium and ISSM,IEEE, 2019.

[5] J. Lin., T. Ono., QX. Zhu., CD. Mao., H. Takahashi., S. Morie., S. Arima., *Multi-criteria optimization of n-step hybrid flow-shop scheduling*, Proceedings of the 2022 International Symposium on Flexible Automation, ISFA, pp. 271-278, 2022.

[6] T. P. Vasconcelos, D. Augusto R.M.A. Souza, Gustavo C. de M. Virgolino, C. L.C. Mattos, J. P.P. Gomes. *Self-tuning portfolio-based Bayesian optimization*, Expert Systems With Applications, 2021.

[7] CD. Mao., J. Lin., S.Arima., *Self-tuning Optimization to Compatible the Delivery and Low Energy Consumption*, IEEE, 2022.

[8] Y.Tahara., *Multi-Product, Multi-Stage Flow Considering Dynamic Change Optimal production capacity allocation*,University of Tsukuba Cllege of Policy and Planning Sciences Graduation Research Paper , Ibaraki, Japan, 2022.

PROGRAMARE ÎN MAI MULTE FABRICI PENTRU UN LANȚ DE APROVIZIONARE CORPORATIV

Această lucrare prezintă programarea hibridă în n pași (nHFS) pentru un lanț de aprovizionare corporativ în care fabricile de asamblare a componentelor și de asamblare finală sunt legate în serie (tip tandem). Am aplicat metoda n-GuptaEX=SETUPBO (Mao et al., 2022), care este o formă avansată a uneia dintre soluțiile nHFS reprezentative propuse de J.N.D Gupta et al. (2002). Ca bază de referință, metoda n-GuptaEX-SETUPBO a îmbunătățit mult atât nivelul de optimizare, cât și eficiența computațională în studiul nostru anterior. Acum, pentru cazul fabricilor multiple, fiecare fabrică are o utilitate diferită și există o relație de compromis între ele. Scopul acestui studiu este de a îmbunătăți performanța întregului lanț de aprovizionare prin intermediul unei programări integrate care să echilibreze interesele fiecărei fabrici. Performanța programării este evaluată prin compararea acesteia cu alte abordări existente. În experimentele numerice se utilizează simularea evenimentelor discrete.

Yusaku TAHARA, PhD student, University of Tsukuba, System Information Engineering Research Group, s2220507@u.tsukuba.ac.jp, +81 -298-535-558, 1-1-1, Tennodai, Tsukuba city, Ibaraki pref., Japan.