Journal of Knowledge Information Technology and Systems

ISSN 1975-7700

http://www.kkits.or.kr

Routing Policies for Flow Shop with Competing Operations

Da-Seol Jo, Jun-Woo Kim*

Department of Industrial and Management Systems Engineering, Dong-A University

ABSTRACT

This paper aims to introduce the concept of competing operations in flow shop and investigate the characteristics of the flow shop scheduling problem with competing operations. While different operations have different processing orders in traditional flow shop scheduling problem, some operations can be processed in arbitrary order in practical shop floors. In other words, the processing orders of given operations are fixed in traditional flow shop scheduling problem, however, this paper considers flow shop scheduling problem where some operations have variable processing orders. Such operations are defined as competing operations in this paper, and flow shops with competing operations require additional decisions on routing policies. In this context, we aim to propose several routing policies that can be applied to flow shops with competing operations, and to evaluate their performances in dynamic environment by applying simulation technique. Simulation experiments are performed by using a simulation model for a simple flow shop wich 2 competing operations, which is built by applying a commercial 3D factory simulation software called FlexSim. The experiment results reveal that the performances of proposed routing policies are dependent on the processing times of competing operations. Especially, routing policies based on the total processing time of predecessors have shown promising performances. This implies that the routing policies in flow shops with competing operations should be carefully designed with consideration of model parameters.

KEYWORDS: Flow shop scheduling problem, Competing operation, Dispatching rule, 3D Factory Simulation, FlexSim software

ARTICLE INFO: Received 9 October 2018, Revised 19 November 2018, Accepted 7 December 2018.

*Corresponding author is with the Department of Industrial and Management Systems Engineering, Dong-A University, Nakdongdaero 550beon-gil 37, Saha-gu, Busan, 49315, KOREA.

E-mail address : kjunwoo@dau.ac.kr

1. Introduction

Production scheduling is the process of determining start times or processing orders for pending jobs, and it is a highly important decision making task in manufacturing operations management[1-3]. Permutation flow shop scheduling problem is a sort of production scheduling problem where each job consists of same number of operations that must be processed in identical order. The objective of traditional permutation flow shop scheduling problem is to find the optimal job sequence of given n jobs, $J_1, J_2, ..., J_n$, where each job $J_i (i = 1, 2, ..., n)$ consists of m operations, $o_{i1}, o_{i2}, ..., o_{im}$ and o_{ij} is the jth job of $J_i (j = 1, 2, ..., m)$. Typically, there are m machines, $mc_1, mc_2, ..., mc_m$, which can process only one operation at a time. Moreover, the optimal job sequence is defined as a job sequence which minimizes performance measures such as makespan and average tardiness, etc[4-5]. Note that $PO(o_{ij}) = j$ in traditional permutation flow shop scheduling problem, where $PO(x)$ denotes the possible processing order of an operation x.

On the contrary, some operations can be processed in arbitrary order in practical shop floors. For example, consider a simple 4-machine flow shop with mc_1, mc_2, mc_3 and mc_4, and possible processing orders listed in <Table 1>. In this case, both o_{i2} and o_{i3} can be the second operation of J_i. Note that if o_{i3} is used as the second operation of J_i, o_{i2} must be the third operation of J_i, and vice versa. In this paper, such operations that can be processed in any order are called competing operations in that they compete for earlier processing order. Moreover, such competition among operations has been not dealt with in previous literature on production scheduling. In this context, this paper aims to investigate the characteristics of flow shop scheduling problem with competing operations.

Table 1. Compotition between two operations

Operation	$PO(o_{ij})$
o_{i1}	1
o_{i2}	2,3
o_{i3}	2,3
o_{i4}	4

The remainder of this paper is organized as follows: Section 2 introduces the backgrounds of this paper, and Section3 explains the 3D factory simulation model used to investigate the performance measures of flow shop with competing operations, and suggests several routing policies in flow shop with competing operations. Section 4 presents experiment results obtained by using the simulation model. Finally, Section 5 gives the concluding remarks and the future research topics.

2. Research Backgrounds

Traditionally, research papers on flow shop

scheduling focus on developing heuristics[5-7] or meta-heuristic scheduling algorithms, such as genetic algorithm[8-9], tabu search[10-11], simulated annealing[12] and particle swarm optimization[13], for solving a specific sort of scheduling problem[14].

In permutation flow shop, the processing orders of operations are fixed, and o_{aj} must be processed earlier than o_{bj} for j=2, 3, ..., m, if o_{a1} has been processed earlier than o_{b1}. Therefore, the main objective of the scheduling algorithms for permutation flow shop scheduling problem is to generate efficient job sequence[4-5]. Note that we need not to consider the routings of given jobs in traditional permutation flow shop scheduling problem, since all jobs visit the machines in identical orders.

On the contrary, the jobs can have different routing paths if there are competing operations. For example, both <mc_1, mc_2, mc_3, mc_4> and <mc_1, mc_3, mc_2, mc_4> are feasible routing paths for operations in <Table 1>. Thus, in permutation flow shop with competing operations, an additional decision making for determining the routing path for each job is required.

The competing operations had been not dealt with in previous literature on production scheduling, and the first objective of this paper is to introduce the concepts and characteristics of flow shop scheduling problem with competing operations.

Moreover, the second objective of this paper is to propose some routing policies that can be applied to flow shop with competing operations. Note that both decision makings for job sequencing and job routing are required if competing operations exist, however, job sequencing is not considered in this paper. In other words, this paper focuses on routing policies for determining the processing orders of competing operations for each job.

Finally, our third objective is to investigate the performances of the proposed routing policies. Traditionally, performances of scheduling algorithms are evaluated in static environment, which means that a number of jobs are given at time t=0 and no additional jobs are allowed to arrive during processing the initial jobs. However, in order to gain practical insights into the competing operations, the performances of routing policies for flow shops with competing operations should be evaluated in dynamic environment, where additional jobs are allowed to arrive after t=0. Therefore, a commercial 3D factory simulation software called FlexSim[15] is used to obtain the performance measures of the routing policies proposed in this paper.

FlexSim software is a powerful tool that can be used to build and operate virtual manufacturing systems, and it is widely applied to analyze the performance measures of manufacturing or logistics systems[16-18].

In this paper, we apply FlexSim software

to analyze the performances of the proposed routing policies with varying simulation model parameters such as processing times.

3. Routing Policies in Flow Shop with Competing Operations

3.1 Structures of flow shop with competing operations

<Figure 1> shows a simple example of flow shop with competing operations, where three competing operations, C, D and E, have consecutive processing orders. In some cases, competing operations have distributed processing orders, however, distributed competing operations are out of the scope of this paper. Other operations, A, B, F and G in <Figure 1>, are non-competing operations that have fixed processing orders. Moreover, among the non-competing operations, operation B is immediate predecessor of the consecutive competing operations.

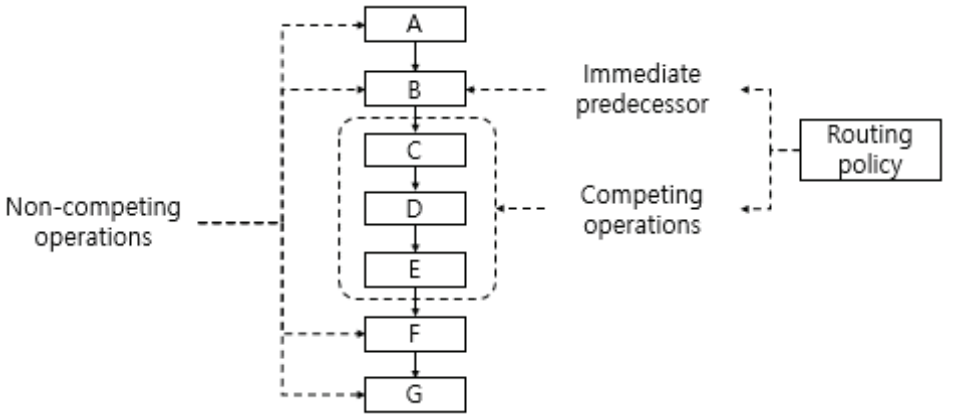

Figure 1. Structure of flow shop with consecutive competing operations

It is straightforward that we have to choose one operation among the three competing operations after operation B. Therefore, appropriate routing policies should be applied to the immediate predecessor of competing operations. Similarly, routing policies can also be applied to competing operations. For example, if operation C is chosen at operation B, we have to choose one operation among D and E after operation B. Consequently, if there are k consecutive competing operations, routing policies have to applied to $k-1$ operations among them. Note that the last kth operation do not need any

routing decision. Moreover, if $k=2$, only the immediate predecessor operation requires routing policy.

This paper focuses on the simplest form of flow shop with competing operations, where 2 consecutive competing operations exist, as shown in <Table 1>. Moreover, we aim to propose several routing policies for the immediate predecessor, o_{i1} in <Table 1>, and evaluate their performances by applying simulation technique.

3.2 Simulation model for flow shop with 2 consecutive competing operations

<Figure 2> shows a 3D simulation model for a flow shop with 4 operations listed in <Table 1>, where o_{i2} and o_{i3} are competing operations. For convenience, transport is not considered in this model. Moreover, 4 types of modeling objects provided by FlexSim software, Source, Queue (Queue1~Queue4), Processor (Machine1~Machine4) and Sink, are used in this model.

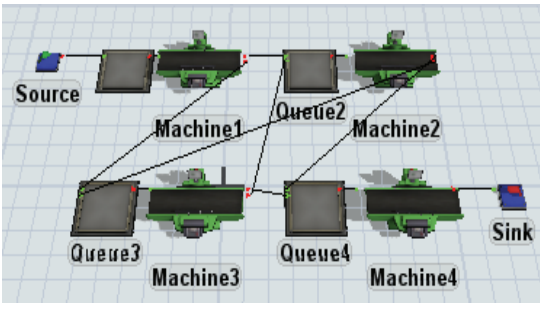

Figure 2. Simple flow shop with two competing operations

The Source object is used to create jobs to be processed by this flow shop, that is, this modeling object can be viewed as the entrance of jobs. The processing time of o_{ij}, $PT_{ij}(j=1,2,3,4)$ is assumed to follow exponential distribution with mean=$mean_j$ and min.=min_j. Since very small sample values close to 0 can be obtained from exponential distribution under min_j=0, we use min_j>0 in this paper. In addition, inter-arrival time of jobs is exponentially distributed with mean= λ^{-1} and min=0. That is, inter-arrival time can have very small value close to 0 in our model. Note that exponentially distributed processing times and inter-arrival time are used, in order to evaluate the performances of routing policies in dynamic environment.

Objects Queue1~Queue4 are waiting areas for jobs to be fed into Machine1~Machine4, where operations $o_{i1} \sim o_{i4}$ are processed, and a job in a Queue object cannot move to another Queue object until the associated operation is completed. Finally, a job is transferred to Sink object after all 4 operations are completed, and this object deletes such finished jobs from the model. Note that Machine1 object, which performs the immediate predecessor operation, can send a job to one of Queue2 and Queue3, since o_{i2} processed by Machine2 and o_{i3} processed by Machine3 are competing operations.

The arrows in <Figure 2> indicate the transport paths between two objects, and we

can see that Machine1, Machine2 and Machine3 have two outgoing arrows, indicating these objects can send a job to one of two subsequent modeling objects. While Machine1 can always choose one of two subsequent modeling object, Machine2 (Machine3) can send a job to Queue3 (Queue2) if and only if the job has not visited Machine3 (Machine2), yet. Otherwise, Machine2 and Machine3 must send a job to Queue4.

In other words, the routing at Machine1 is not dependent on other modeling objects, while the routings at Machine2 and Machine3 are dependent on the previous routing at Machine1. Therefore, only the immediate processor, processed by Machine1 in <Figure 2>, requires routing policy under $k = 2$.

Although FlexSim software enables the users to apply various queueing policies to the Queue objects, all queue objects in our model adopt conventional FIFO (first in, first out) policy. This means that our model operates in similar manner with traditional permutation flow shop with 4 operations, except that there are 2 competing operations.

3.3 Routing policies for immediate predecessor under 2 consecutive competing operations

For the immediate predecessor, Machine1 in our model, we propose following routing policies:

- Policy1 (Machine2-first) : All jobs have to visit Machine2 before Machine3.

- Policy2 (Machine3-first) : All jobs have to visit Machine3 before Machine2.

- Policy3 (random) : All jobs can choose one of the competing operations randomly after the immediate predecessor

- Policy4 (SPT-first) : One of the competing operations is chosen by using SPT (shortest processing time) criteria. That is, J_i visits Machine2 before Machine3 if $PT_{i2} < PT_{i3}$, and vice versa.

- Policy5 (LPT-first) : One of the competing operations is chosen by using LPT (longest processing time) criteria. That is, J_i visits Machine2 before Machine3 if $PT_{i2} > PT_{i3}$, and vice versa.

- Policy6 (STPTP-first) : One of the competing operations is chosen by using STPTP (smallest total processing time of predecessors) criteria. That is, J_i visits Machine2 before Machine3 if $\sum_{i \in Q2} PT_{i,2} < \sum_{i \in Q3} PT_{i3}$, and vice versa, where $Q2$ and $Q3$ are set of the jobs waiting in Queue2 and Queue3 objects, respectively.

- Policy7 (LTPTP-first) : One of the competing operations is chosen by using

LTPTP (largest total processing time of predecessors) criteria. That is, J_i visits Machine2 before Machine3 if $\sum_{i \in Q2} PT_{i,2} > \sum_{i \in Q3} PT_{i3}$, and vice versa.

- Policy8 (STPTS-first) : One of the competing operations is chosen by using STPTS (smallest total processing time of successors) criteria. That is, J_i visits Machine2 before Machine3 if $\sum_{i \in Q1} PT_{i,2} < \sum_{i \in Q1} PT_{i,3}$, and vice versa, where $Q2$ is the set of the jobs waiting in Queue1 object.

- Policy9 (LTPTS-first) : One of the competing operations is chosen by using LTPTS (largetst total processing time of successors) criteria. That is, J_i visits Machine2 before Machine3 if $\sum_{i \in Q1} PT_{i,2} > \sum_{i \in Q1} PT_{i,3}$, and vice versa.

Note that policy4 and policy5 are different from the conventional SPT and LPT dispatching rules in that they are routing policies.

The policy1~policy9 are applied to Machine1 in our model with varying the model parameters as listed in <Table 2>. We can see that the parameters regarding inter-arrival time, PT_{i1} and PT_{i4} are fixed. On the other hand, the parameters regarding PT_{i2} and PT_{i3}, processing times of the competing operations, can have values indicating high, moderate and low levels.

Consequently, we consider 3 cases, (moderate PT_{i2}, moderate PT_{i3}), (high PT_{i2}, low PT_{i3}) and (low PT_{i2}, high PT_{i3}) for simulation experiment, and for each case, 100 repetitions of experiments have been performed under simulation time=10,000, in order to estimate the performance measures such as average wait time, average flow time and average queue length.

Table 2. Model parameters for simulation experiment

Model parameter	Value	
λ^{-1}	50	
$mean_1/min_1$	30 / 20	
$mean_4/min_4$	30 / 20	
$mean_2/min_2$	High	40 / 30
	Moderate	30 / 20
	Low	20 / 10
$mean_3/min_3$	High	40 / 30
	Moderate	30 / 20
	Low	20 / 10

4. Experimental Results

<Table 3>~<Table 5> shows the results of simulation experiments performed by applying FlexSim software, where AWT, AFT and AQL indicate average wait time, average flow time and average queue length, respectively. Moreover, AQL values in these tables represent the average WIP (work in process) levels in the associated Queue objects.

In <Table 3> which summarizes the experiment results obtained under (moderate

PT_{i2}, moderate PT_{i3}), we can make the following observations:

(i) The policy6 and policy7 show promising performances in terms of both AWT and AQL. This implies that the total processing time of predecessors is useful in developing routing policies for flow shop with competing operations.

(ii) Among policy6 and policy7, policy6 has shorter AWT at Queue2, while policy7 has shorter AWT at Queue3. That is, these two policies has significant impacts on the the AWTs at competing operations.

(iii) Policy3, policy4 and policy5 show poor performances in terms of both AWT and AQL. Especially, policy4 is based on conventional SPT policy, however, it seems that the processing time of an individual operation is not useful in developing routing policies for flow shop with competing operations.

(iv) Policy8 and policy9 also show poor performances, and this suggests that the total processing time of successors is not useful in flow shop with competing operations, either.

Table 3. Performance measures under moderate PT_{i2} and moderate PT_{i3}

Routing policy		AWT					AFT	AQL				
		Queue1	Queue2	Queue3	Queue4	Total		Queue1	Queue2	Queue3	Queue4	Total
policy1	mean	405	186.16	149.68	122.2	863.04	1038	8.32	3.5359	2.7337	2.176	16.7656
	StDev	226	110.96	109.26	84.3		299	4.86	2.1917	2.0051	1.584	
policy2	mean	405	146.96	194.86	120.5	867.32	1042	8.32	2.6755	3.7054	2.149	16.8499
	StDev	226	93.59	139.05	73.1		301	4.86	1.7426	2.7349	1.373	
policy3	mean	405	180.10	191.23	128.9	905.23	1078	8.32	3.3598	3.5562	2.282	17.518
	StDev	226	102.43	126.08	79.2		298	4.86	1.9770	2.4265	1.480	
policy4	mean	405	194.66	204.16	115.4	919.22	1090	8.32	3.6150	3.7961	2.036	17.7671
	StDev	226	101.23	126.02	79.6		295	4.86	1.9588	2.4148	1.479	
policy5	mean	405	181.01	187.61	140.0	913.62	1086	8.32	3.3686	3.4930	2.482	17.6636
	StDev	226	103.98	125.51	83.4		310	4.86	1.9997	2.3901	1.570	
policy6	mean	405	142.23	145.78	138.6	**831.61**	**1008**	8.32	2.6587	2.7218	2.492	**16.1925**
	StDev	226	87.53	94.53	88.6		297	4.86	1.6893	1.8069	1.677	
policy7	mean	405	141.09	147.73	138.3	**832.12**	**1009**	8.32	2.6356	2.7547	2.486	**16.1963**
	StDev	226	88.71	97.82	88.8		295	4.86	1.7039	1.8577	1.683	
policy8	mean	405	243.45	243.03	123.8	1,015.28	1176	8.32	4.4754	4.5052	2.141	19.4416
	StDev	226	118.76	125.01	65.1		324	4.86	2.2850	2.4024	1.198	
policy9	mean	405	178.64	174.55	136.4	894.59	1066	8.32	3.3377	3.2460	2.417	17.3207
	StDev	226	100.47	111.89	84.8		307	4.86	1.9795	2.1193	1.589	

Table 4. Performance measures under high PT_{i2} and low PT_{i3}

Routing policy		AWT					AFT	AQL				
		Queue1	Queue2	Queue3	Queue4	Total		Queue1	Queue2	Queue3	Queue4	Total
policy1	mean	405	1202.31	3.75	34.0	1,645.06	1751	8.32	22.6569	0.0523	0.474	31.5032
	StDev	226	285.58	2.15	14.2		358	4.86	5.8276	0.0308	0.205	
policy2	mean	405	1194.97	10.69	28.3	1,638.96	1750	8.32	22.4438	0.2009	0.394	31.3587
	StDev	226	284.85	4.17	12.2		357	4.86	5.7650	0.0846	0.173	
policy3	mean	405	1199.39	12.41	34.7	1,651.5	1759	8.32	22.5690	0.2017	0.483	31.5737
	StDev	226	282.27	4.03	13.8		356	4.86	5.7690	0.0701	0.197	
policy4	mean	405	1197.01	13.51	31.2	1,646.72	1757	8.32	22.5141	0.2441	0.434	31.5122
	StDev	226	283.49	5.77	13.7		355	4.86	5.7675	0.1063	0.195	
policy5	mean	405	1203.47	10.80	36.8	1,656.07	1761	8.32	22.6527	0.1575	0.512	31.6422
	StDev	226	286.36	4.55	14.0		357	4.86	5.8328	0.0694	0.204	
policy6	mean	405	1178.43	11.08	29.3	**1,623.81**	**1735**	8.32	22.1726	0.2082	0.408	**31.1088**
	StDev	226	284.44	4.26	12.9		359	4.86	5.7596	0.0863	0.184	
policy7	mean	405	1179.55	11.09	29.3	**1,624.94**	**1736**	8.32	22.1840	0.2085	0.408	**31.1205**
	StDev	226	284.23	4.28	12.4		359	4.86	5.7541	0.0869	0.178	
policy8	mean	405	1194.02	12.55	29.6	1,641.17	1753	8.32	22.4295	0.2339	0.412	31.3954
	StDev	226	281.31	5.26	12.5		354	4.86	5.7194	0.1026	0.177	
policy9	mean	405	1204.00	4.93	33.8	1,647.73	1754	8.32	22.6902	0.0697	0.470	31.5499
	StDev	226	287.32	3.10	13.8		360	4.86	5.8650	0.0472	0.201	

Table 5. Performance measures under low PT_{i2} and high PT_{i3}

Routing policy		AWT					AFT	AQL				
		Queue1	Queue2	Queue3	Queue4	Total		Queue1	Queue2	Queue3	Queue4	Total
policy1	mean	405	11.37	1170.55	28.8	1,615.72	1736	8.32	0.2133	21.9840	0.405	30.9223
	StDev	226	4.69	353.26	14.8		414	4.86	0.0914	6.9645	0.225	
policy2	mean	405	3.95	1176.16	33.6	1,618.71	1735	8.32	0.0558	22.1856	0.472	31.0334
	StDev	226	1.92	354.08	14.1		414	4.86	0.0292	7.0228	0.214	
policy3	mean	405	14.44	1172.51	35.2	1,627.15	1745	8.32	0.2378	22.0650	0.496	31.1188
	StDev	226	7.13	352.92	14.9		412	4.86	0.1186	6.9836	0.226	
policy4	mean	405	13.37	1172.08	31.3	1,621.75	1741	8.32	0.2329	22.0284	0.440	31.0213
	StDev	226	5.30	348.89	14.5		408	4.86	0.0979	6.9050	0.220	
policy5	mean	405	10.62	1178.19	36.1	1,629.91	1747	8.32	0.1561	22.2038	0.507	31.1869
	StDev	226	4.70	352.98	14.9		411	4.86	0.0728	7.0019	0.228	
policy6	mean	405	11.56	1155.80	29.3	**1,601.66**	**1723**	8.32	0.2170	21.7487	0.413	**30.6987**
	StDev	226	4.72	350.00	14.8		411	4.86	0.0926	6.9184	0.224	
policy7	mean	405	11.60	1154.12	29.3	**1,600.02**	**1721**	8.32	0.2176	21.7241	0.414	**30.6757**
	StDev	226	4.67	352.95	15.0		413	4.86	0.0914	6.9613	0.227	
policy8	mean	405	13.10	1168.99	30.3	1,617.39	1738	8.32	0.2436	20.9576	0.427	30.9482
	StDev	226	5.83	353.41	15.2		413	4.86	0.1108	6.9658	0.231	
policy9	mean	405	5.41	1178.85	33.7	1,622.96	1739	8.32	0.0778	22.2261	0.473	31.0969
	StDev	226	3.17	354.95	14.0		413	4.86	0.0487	7.0384	0.211	

On the contrary, <Table 4> and <Table 5>, which represent the experiment results under (high PT_{i2}, low PT_{i3}) and (low PT_{i2}, high PT_{i3}), respectively, provides following observations:

(i) <Table 4> and <Table 5> also suggests that policy6 and policy7, based on total pocessing time of predecessors, are useful in flow shop with competing operations.

(ii) However, differences in performances of various routing policies become smaller when one of competing operations has longer processing time than the other. For instance, in <Table 3>, the best total AWT obtained by policy6 is 18% shorter than the worst total AWT obtained by policy8((1015.28–831.61)/1015.28). On the other hand, in <Table 4>, the best total AWT obtained by policy6 is 1.9% shorter than the worst total AWT obtained by policy5((1656.07–1623.81)/1656.07). In other words, the effects of the promising routing policies such as policy policy6 and policy7 become evident when the process times of competing operations are similar.

5. Conclusions

This paper introduces the concept of competing operations that can exist within practical flow shops, which has been not dealt with in previous literature on production scheduling.

Moreover, we described the structure of flow shop with competing operations, and pointed out that appropriate routing policies should be applied to immediate predecessor operation. In this context, this paper proposes 9 routing policies for immediate predecessor operation, and investigates their performances by applying simulation technique.

We have built a 3D simulation model for a simple 4-machine flow shop with 2 competing operations by using a commercial 3D factory simulation software called FlexSim, and the experiment results provide several several meaningful insights into the flow shop scheduling problem with competing operations. Firstly, conventional routing policies, including random routing policy and routing policies based on individual operation's processing time, can produce poor performance measures for entire production system. Secondly, the total processing time of predecessors has significant impacts on the performance measures for entire production system. Thus, they can be very useful in developing routing policies for flow shop with competing operations. Thirdly, the effects of the routing policies based on the total processing time of predecessors become evident when the processing times of competing operations are similar. Hence, we can conclude that the processing times of competing operations should be similar in order to utilize the routing policies proposed in this paper.

However, we still have several further topics for future research on flow shops with

competing operations. Firstly, this paper considered only flow shop with 2 consecutive competing operations, which is the simplest form of competition among operations. Thus, structures and routing policies of flow shops with 3 or more competing operations should be studied in future. Secondly, while we focused on routing policy for flow shops with competing operations, job sequencing was not considered in this paper. However, in order to solve static flow shop scheduling problem with competing operations, both decisions on routing policy and job sequencing are required. In this context, we continue to make effort for developing heuristic and meta-heuristic scheduling algorithms through integration of routing policies and job sequencing algorithms. Finally, competitions among operations can also occur in job shop, which generally has more complex structure than flow shop. Therefore, routing policies and scheduling algorithms for job shop with competing operations may be another valuable topic for future research.

References

[1] M. L. Pinedo, *Scheduling-theory, algorithms, and systems*, Springer, 2016.

[2] Sule, *Industrial Scheduling*, PWS Publishing Company, 1997.

[3] J. Branke, S. Nguyen, C. W. Pickardt, and M. Zhang, *Automated design of production scheduling heuristics: A review*, IEEE Transactions on Evolutionary Computation., Vol. 20, No. 1, pp. 110-124, 2016.

[4] J. M. Framinan, J. N. Gupta, and R. Leisten, *A review and classification of heuristics for permutation flow-shop scheduling with makespan objective*, Journal of the Operational Research Society, Vol. 55, No. 12, pp. 1243-1255, 2004.

[5] R. Ruiz, and C. Maroto, *A comprehensive review and evaluation of permutation flowshop heuristics*, European Journal of Operational Research, Vol. 165, No. 2, pp. 479-494, 2005.

[6] Q. K. Pan, and R. Ruiz, *A comprehensive review and evaluation of permutation flowshop heuristics to minimize flowtime*, Computers and Operations Research, Vol. 40, No. 1, pp. 117-128, 2013.

[7] S. Hatami, R. Ruiz, and C. Andrés-Romano, *Heuristics and metaheuristics for the distributed assembly permutation flowshop scheduling problem with sequence dependent setup times*, International Journal of Production Economics, Vol. 169 , pp. 76-88, 2015.

[8] T. Murata, H. Ishibuchi, and H. Tanaka, *Genetic algorithms for flowshop scheduling problems*, Computers and Industrial Engineering, Vol. 30, No. 4, pp. 1061-1071, 1996.

[9] J. W. Kim, *Candidate order based genetic algorithm (COGA) for constrained sequencing problems*, International Journal of Industrial Engineering: Theory, Applications and Practice, Vol. 23, No. 1, pp. 1-12, 2016.

[10] M. Ben-Daya, and M. Al-Fawzan, *A tabu search approach for the flow shop scheduling problem*, Vol. 109, No. 1, pp. 88-95, 1998.

[11] J. W. Kim, *Performance comparison of*

neighborhood structures of tabu search algorithms for sequencing problems, Advanced Science Letters, Vol. 23, No. 10, pp. 10436-10439, 2016.

[12] I. H. Osman, and C. N. Potts, *Simulated annealing for permutation flow-shop scheduling*, Omega, Vol. 17, No. 6, pp. 551-557, 1989.

[13] B. Liu, L. Wang, and Y. H. Jin, *An effective PSO-based memetic algorithm for flow shop shceduling*, IEEE Transactions on systems, Man, and Cybernetics, Part B (Cybernetics), Vol. 37, No. 1, pp. 18-27, 2007.

[14] J. S. Neufeld, J. N. Gupta, and U. Buscher, *A comprehensive review of flowshop group scheduling literature*, Computers and Operations Research, Vol. 70, pp. 56-74, 2016.

[15] FlexSim Software Products, http://www.flexsim.com, Aug. 2018.

[16] M. Beaverstock, A. Greenwood, E. Lavery, and W. Nordgren, *Applied simulation – modeling and analysis using FlexSim*, BookBaby, New Jersey, 2011.

[17] H. Jia, K. Yu, and J. Zhang, *The simulation and optimization on the certain type fuel pump assembly line balance based on Flexsim*, Applied Mechanics and Materials, Vol. 741, pp. 850-855, 2015.

[18] B. S., Kumar, V. Mahesh, and B. S. Kumar, *Modeling and analysis of flexible manufacturing system with FlexSim*, International Journal of Computational Engineering Research, Vol. 5, No. 10, pp. 1-6, 2015.

경쟁 공정을 갖는 흐름 생산 시스템의 라우팅 정책

조다설[1], 김준우[2]

[1]*동아대학교 산업경영공학과 석사과정*
[2]*동아대학교 산업경영공학과 부교수*

요 약

본 논문에서는 경쟁 공정이 있는 흐름 생산 일정계획 문제의 개념을 소개하고 이러한 문제의 특성에 대해 탐구해보고자 한다. 일반적인 흐름 생산 일정계획 문제에서 각 공정들이 서로 다른 작업 순서를 갖는 것과 달리, 일부 공정들이 임의의 순서로 수행될 수 있는 경우에는 이들 간의 경쟁이 발생한다. 다시 말해, 일반적인 흐름 생산 일정계획 문제에서 각 공정들의 작업 순서가 고정되어 있는 것과 달리, 본 논문에서는 일부 공정들의 작업 순서가 변경될 수 있는 흐름 생산 일정계획 문제를 다룬다. 본 논문은 이러한 공정들을 경쟁 공정이라 정의하며, 흐름 생산 시스템에 경쟁 공정이 있는 경우에는 라우팅 정책에 대한 부가적인 의사결정이 필요해진다. 이러한 맥락에서 본 연구에서는 경쟁 공정을 갖는 흐름 생산 시스템에 적용할 수 있는 여러 가지 라우팅 정책들을 제안하고, 시뮬레이션 기법을 이용하여 이들의 성능을 동적인 환경 하에서 평가해보고자 한다. 실제 시뮬레이션 실험에는 2개의 경쟁 공정을 갖는 간단한 흐름 생산 시스템에 대한 모형이 이용되었으며, 해당 모형은 상용 3D 공장 시뮬레이션 소프트웨어 중 하나인 FlexSim을 가지고 작성하였다. 실험 결과 제안한 라우팅 정책들의 성능이 경쟁 공정들의 처리 시간에 영향을 받는다는 점을 확인할 수 있었으며, 특히, 선행 작업들의 처리 시간 총합에 기반한 라우팅 정책들이 좋은 성능을 보였다. 이는 경쟁 공정을 갖는 흐름 생산 시스템의 라우팅 정책을 정할 때, 모형의 파라미터들이 신중하게 고려되어야 함을 의미한다.

Acknowledgments

This research was supported by the KIAT(Korea Institute for Advancement of

Technology) grant funded by the Korea Government(MOTIE: Ministry of Trade Industry and Energy). (No. N0002429).

Da Seol Jo is currently studying for a M.S. degree in Industrial and Management Systems Engineering at Dong-A University. Her current research interests include database application, information system, data mining and 3D factory simulation, etc.
E-mail address: ektjf0422@daum.net

Jun Woo Kim received his B.S. degree and M.S. degree in Industrial Engineering from KAIST, Korea, in 2001 and 2003, and Ph.D. degree in Industrial and Systems Engineering from KAIST, Korea, in 2009. He now serves as an associate professor in the Department of Industrial and Management Systems Engineering at Dong-A University, Korea. His current research interests include combinatorial optimization, meta heuristics, data mining, artificial intelligence and 3D factory simulation, etc.
E-mail address: kjunwoo@dau.ac.kr