Integrated Batch Production and Multiple Preventive Maintenance Scheduling on A Single Machine to Minimize Total Actual Flow Time

IOP Conf. Series: Materials Science and Engineering **598** (2019) 012083 doi:10.1088/1757-899X/598/1/012083

Integrated Batch Production and Multiple Preventive Maintenance Scheduling on A Single Machine to Minimize Total Actual Flow Time

R Yusriski[1], B Astuti[1], M Ilham[1] and Zahedi[2]

[1] Department of Industrial Engineering, Faculty of Engineering, Universitas Jenderal Achmad Yani (UNJANI), PO. BOX 807 Bandung Indonesia
[2] Department of Mathematics, School of Computer Science, Binus University, Syahdan Street 9, Jakarta 11480, Indonesia

yusarisaki@yahoo.co.id

Abstract. We discuss the problem of batch production and multiple preventive maintenance (PM) scheduling on a single machine. On batch production systems, there is a situation that PM activity should be executed at the time when the machine is processing the parts on a batch. Delaying that action could cause an increase in the possibility of a rejected product. However, applying the PM action would interrupt the process and also need an additional setup when starting the process again. It could impact the longer total flow time. This research proposed two algorithms (the optimal and the heuristic algorithm) which integrated both of batch production and multiple preventive maintenance schedules. The performance of the algorithm measured by minimizes the total actual flow time. Experimental results show that the effectivity of the heuristic algorithm is between 66% until 100% from the optimal algorithm.

1. Introduction

It is a common sense that the preventive maintenance (PM) action have a contribution to minimizing the probability of producing parts in failure machine. The other side, The implementation of PM action also could take up time for production. That is is a trade-off between maintenance and production schedule. However, in the real system, we usually see that both of these acts often planned individually. Similar to the real systems, so far as we know, more than two decades ago, the research in both of these fields also researched independently. Recently, there is more researcher has an interest in the area; some of them are Graves and Lee [1], Qi et al. [2], Cassadi and Cutanoglo [3], and Zahedi et al [4].

Graves & Lee [1] discuss maintenance activities on a single-machine job production scheduling problem. The performance of the schedule is the total weighted completion time. They assume that there is only one maintenance action can be scheduled during the planning horizon and also show that the more length of the planning horizon, the more complexity of the results. Qi et al. [2] also discuss on a single-machine job scheduling problem but with considering the possibility of multiple maintenance actions during a schedule period. Cassadi and Cutanoglo [3] develop a similar case with Qi et al. [3] by considering the risk when the machine does not perform maintenance. The result indicates that the total weighted completion time can be minimized by scheduling the jobs with the weighted shorter processing time forwardly.

Zahedi et al. [4] discuss an integrated of batch production and maintenance with due date consideration on Just In Time (JIT) situation. Zahedi et al. [4] show that, by adopting a backward

scheduling approach, the LPT rule is powerful to the minimum total production cost and maintenance. However, Zahedi et al. [4] assume that the processing time is fix and constant during the schedule period. Meanwhile, Yusriski et al. [5]-[6] show that the processing time could decrease continuously due to the deterioration until reaching a failure, so the maintenance action must be applied to restore machine condition "as good as new."

This research discusses a case similar to Casadi and Cutalogno [3] and also Zahedi et al. [4] in a just in time (JIT) situation with considering the deterioration effect to the processing time of batch. We assume that there are multiple failure machines which occurs during the machining process. It can be observed that the failure could take place when the machine still processes a batch. It causes the machine must stop, do the PM action, and continue the process as a new batch after the PM has done. This situation could increase the number of setups and also the flow time. We propose the algorithm that integrates the decision both of the PM action and the batch schedule with the objective is to minimize total actual flow time, defined by Halim et al. [7] as "the total of the time interval of all parts in batches, start from that arrival to the common due date."

The research is organized as follows. We discuss in the second Section the problem formulation. The solution methods would be considered in the third, and the fourth shows some numerical experiences. Finally, the last is concluding remarks.

2. The Problem Formulation

Consider there is a single machine in manufacturing systems processed some jobs into several batches ($Q_{[i]}$, i = 1, 2, ..., N). The process on a batch cannot be interrupted, and there is a setup time before the machine process a batch. The jobs arrive at the right time with the right quantity, and all completed jobs must be delivered at the common due date precisely. Since the machine usage continuously, the reliability of machine degrade. The probability of the machine degradation showed by using the two parameters (α, β) of Weibull distribution with shape parameter more than one ($\beta > 1$). The first failure shows by ROCOF function, so the scale (α) parameter considered as a limitation. Assume there are multiple failures occurs during schedule period; thus multiple preventive maintenance (PM) action must be done to restore the machine "as good as new." The problem is the schedule of batches with considering the PM action with the objective of minimizing the total actual flow time. The decisions are to determine the number of batches, batch size, schedule the resulting batches, and also time interval between two the PM action. Let us define the following constants and variables as follows:

Parameters:

n	:	the number of demands	α_0	:	initial scale parameter (Weibull parameter)
d	:	a common due date	β	:	the shape parameter (Weibull parameter)
p	:	the processing time of batch at N^{th} position (initial processing time)	γ	:	load usage parameter
s	:	batch setup time	λ	:	maintenance duration

Dependent variables

$T_{[i]}$	:	the batch processing time at the i^{th} position, i=1, 2, …, N	$B_{[i]}$	:	the starting time of batch at the i^{th} position, i=1, 2, …, N
$\alpha_{[i]}$	:	the rest of age of machine at the i^{th} position, i=1, 2, …, N	$\Lambda(T)_{[i]}$	:	the probability of machine failure at the i^{th} position, i=1, 2, …, N

Decision variables

$X_{[i]}$	:	the decision of PM action at the i^{th} position, i=1, 2, …, N
$Q_{[i]}$	:	batch sizes at the i^{th} position, i=1, 2, …, N
N_q	:	the number of batches

The objective function:

TF^a : the total actual flow time

A mathematical model as shows as follow.

$$\text{Minimize } TF^a = \sum_{i=1}^{N_q}\left(\left\{\sum_{j=1}^{i}\left(s+T_{[j]}Q_{[j]}+x_{[j]}\lambda\right)-s-X_{[i]}\lambda\right\}Q_{[i]}\right) \tag{1}$$

Subject to constraints

$$\sum_{i=1}^{N_q} Q_{[i]} = n \tag{2}$$

$$\sum_{i=1}^{N_q} T_{[i]}Q_{[i]} + \left(N_q - 1\right)s + N_f\lambda \le d \tag{3}$$

$$B_{[1]} + T_{[1]}Q_{[1]} = d \tag{4}$$

$$Q_{[i]} \ge 1 \text{ and an integer,} \tag{5}$$

$$1 \le N_q \le n \text{ and an integer,} \tag{6}$$

$$\left\lceil \alpha_{[i]}/T_{[i]} \right\rceil \le Q_{[i]} \tag{7}$$

$$X_{[i]} = \begin{cases} 1, & \text{if PM is performed} \\ 0, & \text{otherwise} \end{cases}$$

$$N_f = \sum_{i=1}^{Nq} x_{[i]} \tag{8}$$

$$T_{[i]} = p + \left[1 + \Lambda\left(T\right)_{[i]}\right], \quad \Lambda\left(T\right)_{[i]} \le 1 \tag{9}$$

$$\Lambda\left(T\right)_{[i]} = \left(\sum_{k=1-1}^{Nq} T_{[k]}Q_{[k]} / \alpha_{[k]}\right)^{\beta} \text{ where } \beta > 1 \tag{10}$$

$$\alpha_{[k]} = \alpha_0\left[\alpha_0 \Big/ \left(\alpha_0 + \sum_{k=1-1}^{Nq} T_{[k]}Q_{[k]}\right)\right]^{\gamma} \text{ where } \alpha_0,\ \gamma > 0, \tag{11}$$

Equation (1) is the objective, i.e., minimizes the total actual flow time. Expresses in Constraint (2), the total production of jobs in all batches must equal to the demand. Constraint (3) explains that all batches must be scheduled at the time interval starting from time zero to the common due date. Constraint (4) expresses that the batches scheduled backwardly where the batch in the first position must be completed at the common due date exactly. Constraint (5) shows that the discrete manufacturing system with the minimum batch sizes must consist of one job. Constraint (6) explains that the number of batches must between one and the number of demands, and an integer. Constraint (7) expresses the binary variable of the PM action. If there is exist the PM action, so the value is one, and if there is not the Preventive Maintenance (PM) action the value is zero. Constraint (8) shows that the number of PM action is the sum of the binary variables. Constraint (9) shows that the processing time of batch increase due to machine degradation (deterioration). The ROCOF function demonstrates that degradation with the value is less than or equal to one. The value is less than one indicates the failure rate increase but the machine still working, and the value is one indicate the machine gets a failure, the machine must stop and need PM action. Constraint (10) shows that the ROCOF function with the Weibull shape parameter (β) is more than one, so the failure rate would increase with time. Constraint (11) describe the accelerated failure time to show that the increase of failure rate effect of decreasing the value of Weibull scale parameter (α).

3. Solution Methods

We develop the optimal algorithm, the so-called the Common due date-Single Machine with Maintenance-Integer Composition (CSMMIC) Algorithm, by using the Integer Composition method, where the solution is searching the total actual flow time for all possible batch-size combinations. There would $2n$-1 batch size combination, where n is the number of demand. The steps for generating the combination can see on Yusriski et al. [8]. The algorithm to solve the problem can be shown as follows.

[The CSMMIC Algorithm]

Step 1 : Input parameters n, d, p, s, α_0, β, γ, μ. Continue to Step 2.

Step 2 : Generate all set of batch-size combinations (BSC) by using the Integer Composition Algorithm.
(The algorithm could be found in Shen and Evans [12])

Step 3 : For each of BSC, scheduled that backwardly, and then do the following steps:

Step 3.1 : Calculate T[i], $\boldsymbol{\alpha}_{[i]}$, Λ(T)[i], and also $\lfloor \boldsymbol{\alpha}_{[i]}/\boldsymbol{T}_{[i]} \rfloor$, (i = 1, 2, ..., n) using Eq. (9-11).
If. $\lfloor \boldsymbol{\alpha}_{[i]}/\boldsymbol{T}_{[i]} \rfloor$<Q[i], then the solution status is not feasible (NF); Otherwise feasible (F).
Continue to Step 3.2

Step 3.2 : Compute TFa by Eq. (1) with the constraints which are Eq. (2-8). Continue to Step 4.

Step 4 : Find minimum total actual flow time just from the BSC with consist feasible batch size. STOP

The demonstration of the algorithm show with a problem as follows. Suppose there are four units demand (n) processed on a single machine, and all completed jobs must be delivered at 20 days as a due date (d). However, the processing time (p) and the setup time (s) is one day per unit and 0.5 days per times respectively. Assume there is a failure with the Weibull parameters (α_0, β,) are 2, 1 and load usage parameter (γ) is one. The preventive maintenance schedule to be planned with a duration (μ) is one day per times. The solution of the CSMMIC Algorithm as follows.

Step 1 : Input parameters n, d, p, s, α_0, β, γ, μ. Continue to Step 2.
n = 4 units, d = 20 days, p = 1 day per unit, s = 0.5 day per time,
α_0 = 2 days, β = 1, γ = 1, μ = 1 day per time.

Step 2 : Generate all set of batch-size combinations (BSC) by using the Integer Composition Algorithm.
Nq =1, batch size composition: [4]
Nq =2, batch size composition: [3,1], [2,2], [1,3]
Nq =3, batch size composition: [2,1,1], [1,2,1], [1,1,2]
Nq =4, batch size composition: [1,1,1,1]

Step 3 : For each of BSC, scheduled that backwardly, and then do the following steps:

Step 3.1 : Calculate $T_{[i]}$, $\boldsymbol{\alpha}_{[i]}$, $\Lambda(T)_{[i]}$, and also $\lfloor \boldsymbol{\alpha}_{[i]}/\boldsymbol{T}_{[i]} \rfloor$, ($i$ = 1, 2, ..., n) using Eq. (9-11).
If. $\lfloor \boldsymbol{\alpha}_{[i]}/\boldsymbol{T}_{[i]} \rfloor < Q_{[i]}$, then the solution status is not feasible (NF); Otherwise feasible (F).
Continue to Step 3.2

Step 3.2 : Compute TF^a by Eq. (1) with the constraints which are Eq. (2-8). Continue to Step 4.
The results of Step 3 is shown in Table 1

Step 4 : Find minimum total actual flow time just from the BSC with consist feasible batch size. STOP

Table 1. The result calculation from Step 3

N_q	BSC	$Q_{[i]}$	$T_{[i]}$	$\alpha_{[i]}$	$\Lambda(T)_{[i]}$	$\lfloor \alpha_{[i]}/T_{[i]} \rfloor$	$X_{[i]}$	TF^a	Status
1	4	4	1.00	2.00	0.00	1	0	16	NF
2	[3,1]	3	1.75	1.33	0.75	0	0	22.5	NF
		1	1.00	2.00	0.00	2	0		NF
	[2,2]	2	1.00	2.00	0.00	2	1	15.0	F
		2	1.00	2.00	0.00	2	0		F
	[1,3]	1	1.00	2.00	0.00	2	1	17.5	F
		3	1.00	2.00	0.00	2	0		NF
3	[2,1,1]	2	1.00	2.00	0.00	2	1	16.0	F
		1	1.75	1.33	0.75	0	0		NF
		1	1.00	2.00	0.00	2	0		F
	[1,2,1]	1	1.00	2.00	0.00	2	1	20.5	F
		2	1.75	1.33	0.75	0	0		NF
		1	1.00	2.00	0.00	2	0		F
	[1,1,2]	1	1.75	1.33	0.75	0	0	18.0	NF
		1	1.00	2.00	0.00	2	1		F
		2	1.00	2.00	0.00	2	0		F
4	[1,1,1,1]	1	1.75	1.33	0.75	0	0	19.5	NF
		1	1.00	2.00	0.00	2	1		F
		1	1.75	1.33	0.75	0	0		NF
		1	1.00	2.00	0.00	2	0		F

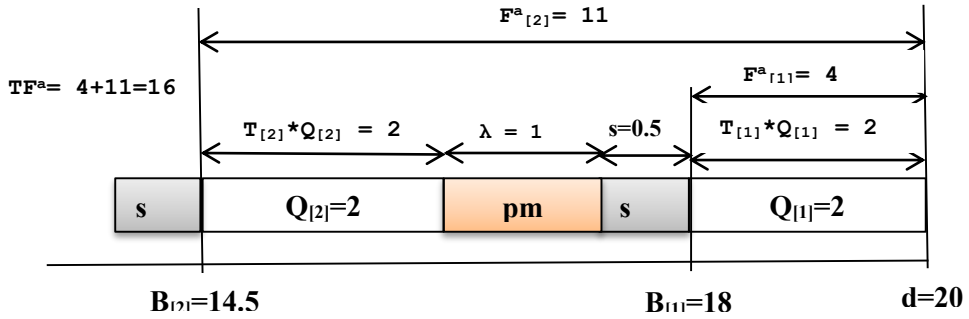

Figure 1. Gantt Chart for the example problem

Figure 1 shows the Gantt chart for the optimal schedule of the case. There are two batches scheduled on a single machine, and The PM action also arranged between two batches. The machine process the second batch ($Q_{[2]}$, last position in backward scheduled approach) and continue to do the PM action before the first failure occurs. The PM action restore the machine as good as new. After that, the machine is done with batch setup, and continue work to process the first batch.

The result of the example shows that the CSMMIC-Algorithm adequate to produce the optimal solution due to the algorithm work by enumerating all of the feasible solutions. However, It is common sense that the enumeration would impact the time to get the optimal solution. There is $n2^{n-1}$ time to generate all possible composition batch size (Yusriski et al. [8]). The result of the experience numeric (We have done by more than 100 cases) shows that the CSMMIC Algoritma appropriate to solve the small case problem (demand less than 20 unit) due to the process out of memory when demand more than 20 units. We propose a heuristic to solve the big size problem. The algorithm developed base on the batch size solution of the research of an integer batch scheduling problem with considering

deterioration (Yusriski et al. [8]). The batch size can be found by Sub-Algoritma Batch Size (SABS-algorithm) as follows.

[The SABS-Algorithm]

Step 1 : Compute Nqmax with the formula as follows.
$Nq_{\max} = \min\left\{1+\left\lfloor (d-pn)/s \right\rfloor, n\right\}$. Continue to Step 2.

Step 2 : Generate the batch size combination (BSC) for Nq =1 to Nq by using the batch size formula as follows

$$Q_{[i]} = \max\left\{\left[\left(n-\sum_{k=i+1}^{N} Q_k\right)/i + (1/2)(i+1)\left(s/T_{[i]}\right) - \left(s/T_{[i]}\right)i\right], 1\right\},\ i = N,...,1$$

where T[i] calculated using Constraint (9-11).

Step 3 : Arrange the result of BSC by the biggest to the smallest batch size.

We insert the sub-algorithm SABS to the CSMMIC algorithm. We found the heuristic procedure, the so-called the CSMMH algorithm, as follows.

[The CSMMH Algorithm]

Step 1 : Input parameters n, d, p, s, α_0, β, γ, μ. Continue to Step 2.
Step 2 : Generate batch-size combinations (BSC) by using the SABS Algorithm.
Step 3 : For each of BSC, scheduled that backwardly, and then do the following steps:
Step 3.1 : Calculate $T_{[i]}$, $\boldsymbol{\alpha}_{[i]}$, $\Lambda(T)_{[i]}$, and also $\lfloor \boldsymbol{\alpha}_{[i]}/\boldsymbol{T}_{[i]} \rfloor$, (i = 1, 2, ..., n) using Eq. (9-11). If. $\lfloor \boldsymbol{\alpha}_{[i]}/\boldsymbol{T}_{[i]} \rfloor < Q_{[i]}$, then the solution status is not feasible (NF); Otherwise feasible (F). Continue to Step 3.2
Step 3.2 : Compute TF^a by Eq. (1) with the constraints which are Eq. (2-8). Continue to Step 4.
Step 4 : Find minimum total actual flow time just from the BSC with consist feasible batch size. STOP

The demonstration of the CSMMH-algorithm show with a similar problem. The result of the solution of the algorithm as follows.

Step 1 : Input parameters n, d, p, s, α_0, β, γ, μ.Continue to Step 2.
n = 4 units, d = 20 days, p = 1 day per unit, s = 0.5 day per time,
α_0 = 2 days, β = 1, γ = 1, μ = 1 day per time.
Step 2 : Generate batch-size combinations (BSC) by using the SABS Algorithm.
Nq =1, batch size composition: [4]
Nq =2, batch size composition: [2,2]
Nq =3, batch size composition: [2,1,1]
Nq =4, batch size composition: [1,1,1,1]
Step 3 : For each of BSC, scheduled that backwardly, and then do the following steps:
Step 3.1 : Calculate $T_{[i]}$, $\boldsymbol{\alpha}_{[i]}$, $\Lambda(T)_{[i]}$, and also $\lfloor \boldsymbol{\alpha}_{[i]}/\boldsymbol{T}_{[i]} \rfloor$, ($i$ = 1, 2, ..., n) using Eq. (9-11). If. $\lfloor \boldsymbol{\alpha}_{[i]}/\boldsymbol{T}_{[i]} \rfloor < Q_{[i]}$, then the solution status is not feasible (NF); Otherwise feasible (F). Continue to Step 3.2
Step 3.2 : Compute TF^a by Eq. (1) with the constraints which are Eq. (2-8). Continue to Step 4.
The results of Step 3 is shown in Tabel 2 as follows.
Step 4 : Find minimum total actual flow time just from the BSC with consist feasible batch size. STOP.
The minimum total actual flow time is 15 found on feasible BSC [2, 2]

The result shows that the CSMMH algorithm produces the same solution with the optimal algorithm.

Table 2. The result calculation from Step 3

N_q	BSC	$Q_{[i]}$	$T_{[i]}$	$\alpha_{[i]}$	$\Lambda(T)_{[i]}$	$\lfloor \alpha_{[i]}/T_{[i]} \rfloor$	$X_{[i]}$	TF^u	Status
1	4	4	1.00	2.00	0.00	1	0	16.0	NF
2	[2,2]	2	1.00	2.00	0.00	2	1	15.0	F
		2	1.00	2.00	0.00	2	0		F
3	[2,1,1]	2	1.00	2.00	0.00	2	1	16.0	F
		1	1.75	1.33	0.75	0	0		NF
		1	1.00	2.00	0.00	2	0		F
4	[1,1,1,1]	1	1.75	1.33	0.75	0	0	19.5	NF
		1	1.00	2.00	0.00	2	1		F
		1	1.75	1.33	0.75	0	0		NF
		1	1.00	2.00	0.00	2	0		F

4. Numerical Experiences

The numerical experiences are determined by comparing the solutions of the CSMMH with the optimal solution, the CSMMIC algorithm. The two procedures are written in MATLAB programming. The both of results of the test are reported by using a computer processor of Intel Core i3 with a 6-GB Random Access Memory (2 GB used on 64-bit computer platform) and 1 GB Graphic Processor Unit. Table 3 and Table 4 presents the date of the small size case and the results of testing 10 cases respectively.

Table 3. The simulation date of the small size problem

Case No	n	d	p	s	α	β	γ	λ
1	5	15	0.5	1.0	2	2	1	1
2	7	18	1.0	1.0	4	2	1	1
3	9	20	0.5	1.0	3	2	1	1
4	10	30	0.5	0.5	5	2	1	1
5	10	50	1.0	2.0	3	2	1	1
6	12	60	0.5	2.0	5	2	1	1
7	14	60	0.5	2.0	6	2	1	1
8	16	70	0.5	1.0	3	2	1	1
9	18	80	0.5	2.0	4	2	1	1
10	20	90	0.5	0.5	4	2	1	1

Table 4. The result of comparison tests

Case	CSMMH Algoritm				CSMMIC Algorithm				Comparison	
	TF^a	N	Number of PM	Time to compute (second)	TF^a	N	Number of PM	Time to compute (second)	Efectivity	Time Effiiciency
1	18.0563	4	1	0.0018	15.5000	2	1	0.0093	86%	0.0075
2	64.4353	7	2	0.0041	42.5859	3	1	0.0408	66%	0.0367
3	41.5904	4	1	0.0040	39.8781	3	1	0.2050	96%	0.2010
4	44.6912	3	1	0.0038	44.6912	3	1	0.4029	100%	0.3991
5	151.9506	9	4	0.0038	109.1852	6	3	0.3802	72%	0.3764
6	79.5493	5	1	0.0057	72.6912	3	1	1.6945	91%	1.6888
7	100.4838	5	1	0.0120	96.7282	3	1	8.0757	96%	8.0637
8	149.8524	10	3	0.0151	116.8781	4	2	35.6574	78%	35.6423
9	145.5671	8	2	0.0157	132.1719	4	2	231.9585	91%	231.9428
10	180.7834	10	3	0.0232	159.4000	5	2	1263.7900	88%	2770.6000

Table 4 showed the CSMMH cannot always produce optimal solutions. The effectivity of solution varies between 66% until 100%. However, it also can be seen that the CSMMH algorithm is more efficient than the CSMMIC algorithm. Thus, the CSMMH algorithm can be used as an alternative solution to real problems, especially for the big size problem.

5. Concluding Remarks

In a real system, there are some the situation when batch production scheduling without involving maintenance schedule. However, this could be a problem when the machine gets a failure because there are some job would be produced on failure machine so the possibility of the rejected product would increase. On the other hand, doing maintenance may have to stop production, add the number of setups and cause the increase of the total flow time. This study proposes two algorithms to integrate the production schedule and machine maintenance schedule. The CSMMIC algorithm can produce an optimal schedule even though it is only suitable for small size cases (small number of requests, under 20 units). The second algorithm (CSMMH) is a heuristic algorithm that can solve the big size case even though it does not guarantee an optimal solution.

In some production environment, there is a situation that the demand scheduled by multiple machines so the degradation could occur on all machine. It leads that a production schedule on a multi-machine should consider various deteriorations too. We know that a maintenance schedule could efficient by using group maintenance. The next research would discuss the integration between the parallel machine with group PM action.

Acknowledgments

The authors wish to thank the editors and the referees for their constructive comments and suggestions. This research was supported by Universitas Jenderal Achmad Yani (UNJANI) Bandung-Indonesia.

References

[1] Graves G H and Lee C Y 1999 *Nav. Res. Log*. **46** 845–63.
[2] Qi X, Chen T and Tu F 1999 *J. Oper. Res. Soc*. 50 1071–8.
[3] Cassady C R and Kutanoglu E 2005 *IEEE. T. Reliab*. **54** 304-9.
[4] Zahedi Z, Samadhi T M A A, Suprayogi S and Halim A 2016 *Int. Journal of Industrial Engineering Computations*. **7** 229-44.
[5] Yusriski R, Sukoyo S, Samadhi T M A A and Halim A H 2014 *Proc. of the 2014 Int. Conf. on Industrial Engineering and Operations Management*. 1234–42.
[6] Yusriski R, Sukoyo S, Samadhi TMAA and Halim A H 2016 IOP. *Conf. Ser-Mat. Sci*. **319** 1-7.
[7] Halim A H, Miyazaki S and Ohta H 1994 *Eur. J. Oper. Res*. **72** 529-44.
[8] Yusriski R, Sukoyo S, Samadhi T M A A and Halim A H 2015 *Int. Journal of Industrial Engineering Computations*. **6** 365-78.