Jurnal Teknik: Media Pengembangan Ilmu dan Aplikasi Teknik
Vol 22, No 02, November 2023, Hal. 171-178
Journal homepage: http://jurnalteknik.unjani.ac.id/index.php/jt ISSN (e): 2580-2615, ISSN (p): 1412-8810

Penjadwalan Dinamis Menggunakan Metode *Rolling Time Window* (RTW) pada Kasus Flowshop 3 Mesin untuk Meminimumkan Total Biaya *Lateness*, *Earliness* dan *Re-Scheduling*

Rinto Yusriski[1], Budi Astuti[1], dan Andri Rachmat Kumalasian Nasution[1]
[1]Program Studi Teknik Industri, Fakultas Teknologi Manufaktur,
Universitas Jenderal Achmad Yani, Kota Cimahi, Indonesia
rinto.yusriski@lecture.unjani.ac.id, budi.astuti@lecture.unjani.ac.id, andri.rachmatk@lecture.unjani.ac.id

Abstrak

Penelitian ini membahas masalah penjadwalan *job* pada *flowshop* tiga mesin dengan tujuan untuk meminimumkan total biaya yang terdiri dari biaya *lateness*, biaya *earliness*, dan biaya *re-scheduling*. Penelitian ini mengambil sistem nyata pada salah satu perusahaan manufaktur yang bergerak pada operasi permesinan berbagai komponen. Terdapat 12 *job* dengan pola kedatangan dan saat tenggat (*due date*) yang berbeda. Perusahaan menggunakan metode penjadwalan *First In-First Out* (*FIFO*) dan mendapatkan hasil berupa beberapa produk selesai lebih awal dan beberapa lainnya selesai terlambat. Penelitian ini mencoba memperbaiki performa penjadwalan perusahaan dengan mengusulkan adanya penjadwalan ulang menggunakan metode *Rolling Time Window* (*RTW*). Hasil penjadwalan menunjukkan adanya perbaikan performa perusahaan yang diukur dari total biaya *earliness*, *lateness* dan *re-schedule* meskipun relatif kecil (3%). Perbaikan ini diperoleh dengan menyarankan perusahaan melakukan *re-schedule* setiap 15 hari sekali dengan mengevaluasi panjang *window* per 30 hari. Metode FIFO yang dipakai perusahaan masih dapat digunakan mengingat karakteristik *job* yang datang ke perusahaan masih terkendali. Perusahaan dinilai mampu membuat kesepakatan dengan konsumen mengenai saat tenggat pengiriman yang membuat antrian *job* dilantai produksi menjadi tidak terlalu padat.

Kata kunci: penjadwalan dinamis, *flowshop*, *rolling-time-window*

Abstract

This study discusses the problem of job scheduling in a three-machine flowshop to minimize the total cost of lateness, earliness, and rescheduling. This research takes a real system from one of the manufacturing companies engaged in the machining operations of various components. There are 12 jobs with different arrival patterns and due dates. The company uses the first-in, first-out (FIFO) scheduling method and gets the result that some products finish early and some finish late. This study tries to improve the company's scheduling performance by proposing a rescheduling using the rolling time window (RTW) method. Although it is relatively small, scheduling results show an improvement in company performance as measured by the total cost of earliness, lateness, and rescheduling, although it is relatively small (3%). This improvement is obtained by suggesting the company reschedule every 15 days by evaluating the length of the window per 30 days. The company can still use the FIFO method used by the company considering the characteristics of the jobs that come to it are still under control. The company is deemed capable of making agreements with consumers regarding the delivery deadline, making the job queues on the production floor less crowded.

Keywords: dynamic scheduling, flowshop, rolling-time-window

1. Pendahuluan

Penjadwalan produksi merupakan salah satu kegiatan penting dalam perencanaan dan pengendalian produksi. Pada penelitian penjadwalan, kedatangan pekerjaan (*job*) sering diasumsikan telah diketahui sebelum periode penjadwalan dimulai. Penjadwalan *job* tersebut dikenal sebagai penjadwalan *job* dengan Pola Kedatangan Statis (PKS). Pada beberapa kasus lapangan, pola kedatangan *job* ke lantai produksi dapat bersifat acak. Penjadwalan untuk kasus ini disebut sebagai penjadwalan *job* dengan Pola Kedatangan Dinamis (PKD). Penelitian penjadwalan PKD telah banyak dibahas baik pada pola aliran *job shop* maupun *flow shop*. Bagi pembaca yang memiliki ketertarikan pada penjadwalan PKD dengan pola aliran *job shop* dapat mendalami Mohan, dkk. (2019) yang telah menyusun tinjauan literatur mengenai masalah tersebut. Sementara pada pola aliran *flowshop*, penelitian-penelitian masalah ini dapat diperoleh di antaranya dalam Nurainun (2012), Ilhami (2014), dan Octanatry, dkk. (2015).

Nurainun (2012) membahas mengenai masalah penjadwalan PKD dengan mempertimbangkan saat tenggat pengiriman (*due date*). Penelitian ini mengasumsikan beberapa *job* dengan karakteristik pengerjaan yang sama dapat diproses secara berkelompok dalam suatu *batch*. Konsekuensinya adalah diperlukan tambahan waktu untuk *setup* (persiapan) setiap kali mesin akan mulai memproses suatu *batch*. Dengan

Info Makalah:
Dikirim : 09-30-21;
Revisi 1 : 04-04-22;
Revisi 2 : 10-19-23;
Diterima : 11-09-23.

Penulis Korespondensi:
Telp : +62 853-2447-0036
e-mail : rinto.yusriski@lecture.unjani.ac.id

demikian, keputusan jadwal adalah menentukan pengelompokan *job* dalam *batch* dan menjadwalkan urutan *batch* tersebut pada mesin. Performa jadwal diukur dengan meminimumkan total biaya yang terdiri dari biaya persiapan, biaya proses, dan biaya simpan.

Octanatry, dkk. (2015) mengembangkan model penjadwalan mesin dinamis pada masalah mesin paralel dengan tujuan meminimasi *weighted tardiness*. Ilhami (2014) membahas penjadwalan PKD pada kasus *flow shop* fleksibel tiga tahap dengan fungsi tujuan meminimasi *tardiness* berbobot (*weighted tardines*s). Pada penelitian tersebut, *job* diproses melalui tiga tahapan operasi. Jumlah mesin pada setiap tahap dapat lebih dari satu mesin (mesin paralel). Metode solusi menggunakan model sistem lelang. Hasil pengujian menunjukkan model usulan mampu untuk memecahkan masalah. Pengujian dilakukan dengan membandingkan solusi usulan dengan solusi perusahaan dan solusi metode *Early Due Date* (EDD). Hasil pengujian menunjukkan bahwa solusi usulan memiliki *tardiness* berbobot yang lebih minimum dibandingkan metode lainnya.

Penelitian ini akan membahas masalah penjadwalan PKD pada pola aliran *flowshop*. Motivasi penelitian diperoleh untuk memecahkan masalah pada sistem nyata di PT. ABC, yaitu salah satu perusahaan manufaktur di Indonesia yang bergerak pada bidang permesinan. Perusahaan memiliki masalah dalam menjadwalkan *job* dengan pola datang yang dinamis. Saat ini, perusahaan menjadwalkan *job* dengan menggunakan aturan prioritas *first in-first out (FIFO).* Pekerjaan diurutkan sesuai dengan saat kedatangan mulai dari *job* yang datang paling awal diikuti oleh *job* baru yang datang mengikutinya. Solusi metode *FCFS* ini masih memerlukan perbaikan karena terdapat sebagian *job* yang selesai melewati waktu tenggat pengiriman. Menurut Yusriski, dkk. (2014, 2015a, 2015b, 2016, 2018, 2019, 2023), pengiriman sesuai dengan saat tenggat merupakan faktor penting untuk memenangkan persaingan dan kepercayaan konsumen.

Permasalahan yang terdapat pada PT. ABC dapat diformulasikan sebagai berikut, diketahui terdapat 12 (dua belas) *job* (*i=1,2,3,…,12*) yang diproses pada *flowshop* 3 tahap operasi (*j=1,2,3*) dimana setiap *job* dikerjakan pada masing-masing satu mesin pada tahap persiapan seperti mesin untuk proses potong (*k=1),* jika tahap persiapan selesai , selanjutnya masuk tahap permesinan dengan tujuan membentuk komponen seperti mesin bubut (*k=2),* dan komponen yang selesai melalui tahap permesinan akan melalui tahap menghaluskan komponen menggunakan mesin gerinda (*k=3*). Setiap *job* datang ke lantai produksi (r_i) tidak secara bersamaan atau bersifat dinamis dengan jumlah unit pemesan (n_i) berbeda-beda dan saat tenggat yang berbeda juga (d_i). Waktu proses setiap *job* di setiap mesin (t_{ik}) unik atau tidak sama dan pada tahap persiapan, *job* memerlukan proses setup (s_{i11}). Perusahaan saat ini menerapkan metode penjadwalan dengan menggunakan aturan prioritas *FIFO*. Masalah yang dihadapi adalah hasil penjadwalan dengan aturan *FIFO* menunjukkan terdapat sejumlah *job* yang selesai lebih awal ($C_i < d_i$, *earliness*) dan beberapa *job* yang selesai melawati saat tenggat yang telah ditentukan ($C_i > d_i$, *lateness*). Hal ini berakibat pada meningkatnya biaya yang harus ditanggung perusahaan berupa biaya simpan untuk *earliness job* dan biaya penalti untuk *lateness job* yang akan berdampak pada penurunan profit perusahaan. Penelitian ini membahas mengenai penjadwalan dengan mempertimbangkan penjadwalan ulang untuk memperbaiki performansi jadwal. Dengan demikian, performa tujuan akan diukur dengan minimasi total biaya yang terdiri atas biaya simpan produk jadi, biaya penalti, dan biaya penjadwalan ulang.

2. Metode

Metode yang digunakan pada penelitian ini mengadaptasi model matematika Suharyanti dan Halim (2000) yang mengembangkan model Sun dan Lin (1994). Fungsi tujuan dalam Sun dan Lin (1994) menggunakan minimasi biaya *tardiness* yang dikembangkan dalam Suharyanti dan Halim (2000) menjadi meminimumkan total ongkos *lateness* dan *earliness*. Model Suharyanti dan Halim (2000) melakukan pendekatan *time window* untuk mengakomodasi penjadwalan statis dan dinamis. Penelitian ini mengadopsi model tersebut untuk kasus *Flowshop* tiga mesin dengan pengembangan berupa penambahan parameter biaya *re-schedule* (penjadwalan ulang).

$$meminimumkan\ Z_i = \sum_{s=1}^{W}\left(\sum_{i=1}^{n_s}\left(\alpha_i max\{C_{irm_i} - d_i, 0\} + \beta_i max\{d_i - C_{irm_i}, 0\}\right) + \varphi\, W\right) \quad (1)$$

Kendala:

$$C_{ijk} - C_{i(j-1)k} \geq t_{ijk}, \quad i \in N_s, j \in R_i, (k,l) \in M, k \neq l, s = 1,2,\dots,W \quad (2)$$

$$C_{ijk} - C_{i(j-1)k} \geq t_{ijk} \wedge C_{i(j-1)k} - C_{ijk} \geq t_{i(j-1)k} \quad i \in N_s, k \in M, s = 1,2,\dots,W \quad (3)$$

$$C_{i1k} - t_{i1k} \geq a_i \quad i \in N_s, k \in M, \quad s = 1,2,\dots,W \quad (4)$$

Fungsi tujuan (1) adalah meminimumkan total biaya yaitu biaya *lateness*, *earliness* dan *re-shedule* pada penjadwalan dengan *W* kali *time window* (*W* adalah variabel keputusan). Kendala (2) adalah kendala yang menjamin bahwa urutan operasi selalu sesuai dengan *routing* setiap *job* sehingga suatu operasi ke-*j* dari *job i* dapat dimulai paling cepat setelah operasi ke-(*j*-1) telah selesai. Kendala (3) menunjukkan bahwa mesin tidak bisa memproses secara bersamaan beberapa operasi pada satu saat yang sama (mesin hanya bisa memproses satu operasi pada saat yang sama). Dari fungsi (3) tersebut dapat dilihat bahwa jika operasi ke *j-1* dari *job i* mendahului operasi ke *j* dari *job i* untuk diproses pada mesin k. Kendala (4) menjamin *job* bisa dijadwalkan pada horizon jadwal dengan menetapkan bahwa saat mulai operasi pertama suatu *job* tidak akan kurang dari *available time job* tersebut.

Metode solusi yang digunakan pada penelitian ini menggunakan metode *Rolling Time Window* (RTW) dalam Suharyanti dan Halim (2000). Horizon jadwal dibagi dalam beberapa sub-periode yang disebut dengan *Window*. Penjadwalan pada suatu *window* kemudian dibagi dalam dua sesi dengan rentang yang sama panjang. Sesi penjadwalan pertama merupakan sesi jadwal statis, dan sesi jadwal kedua merupakan sesi dinamis yang merupakan sesi penjadwalan ulang dengan mempertimbangkan datangnya *job* baru ke lantai produksi. Pada *windows* berikutnya sesi jadwal kedua akan menjadi sesi statis demikian seterusnya hingga seluruh *windows* terselesaikan. *Trade off* dari masalah ini adalah, semakin panjang *window* maka semakin pendek iterasi yang akan diselesaikan namun akan mengakibatkan sesi statis menjadi panjang dan mengakibatkan banyak *job* yang tidak terevaluasi pada sesi dinamis (efisien tapi tidak efektif). Hal ini akan mengakibatkan kualitas solusi menjadi menurun. Semakin pendek panjang *window* maka kualitas solusi menjadi semakin baik namun akan memperbanyak iterasi yang bisa jadi tidak diperlukan (efektif tapi tidak efisien). Dengan demikian perlu ditentukan panjang *window* yang optimal. Perhitungan panjang *window* dapat dirumuskan sebagai berikut:

$$D = \frac{2h}{(W+1)} \text{ atau } W + 1 = \frac{2h}{D} \quad (5)$$

Suharyanti dan Halim (2000) telah mengusulkan algoritma untuk memecahkan masalah yang terbagi dalam dua tahap yaitu:

Tahap 1: Tahap 1 merupakan algoritma utama yaitu algoritma untuk menghitung panjang *time window* dan juga fungsi tujuan (total biaya *lateness* dan *earliness*). Algoritma ini pada dasarnya adalah algoritma pencarian nilai *D*, dengan mencoba-coba secara iteratif (menambahkan nilai *D* sebelumnya dengan nilai tertentu (ΔD) untuk mendapatkan nilai *D* berikutnya. Nilai *D* awal merupakan nilai inisialisasi yang biasanya dibangkitkan secara acak. Keputusan nilai *D* ini menjadi variabel *input* untuk menghitung total biaya *lateness* dan *earliness*. Algoritma akan berhenti jika total biaya *lateness* dan *earliness* pada suatu iterasi tidak lebih baik dari iterasi sebelumnya.

Tahap 2: Tahap 2 terdiri dari dua sub algoritma. Sub algoritma pertama adalah algoritma untuk menentukan alokasi *job* dari suatu *window* ke *window* berikutnya. Input dari sub algoritma pertama adalah urutan *job* hasil dari jadwal statis yang diperoleh dari sub algoritma kedua. Pada kasus Suharyanti dan Halim (2000) sub algoritma kedua menggunakan algoritma jadwal *non-delay* yang sesuai untuk kasus *jobshop*. Pada penelitian ini, pendekatan solusi algoritma kedua mengadopsi aturan prioritas *early due date* (EDD).

Detail mengenai algoritma RTW dapat dilihat pada Suharyanti dan Halim (2000).

3. Hasil dan Pembahasan

Data *job* dan status pengiriman pada kondisi penjadwalan hasil penjadwalan eksisting menggunakan metode FIFO disajikan pada Tabel 1.

Tabel 1. Data *Job* dan Status Hasil Penjadwalan Eksisting.
(Sumber: Pengolahan Data Berdasarkan Kasus pada Sistem Nyata)

(i)	n_i	r_i	d_i	$[t_{i11}, t_{i22}, t_{i33}]$	s_{i1}	C_{i11}	C_{i22}	C_{i33}	Δ_i	Pembulatan Δ_i	*Status job (earliness/ lateness)*	Biaya *Earliness/ Lateness*
1	4	1	8	[0.1, 0.3, 0.4]	1	1,4	2,6	4,2	3,8	4	*earliness*	480.000
2	5	1	2	[0.1, 0.1, 0.3]	1	2,9	3,4	5,7	-3,7	-4	*Lateness*	1.000.000
3	5	3	12	[0.2, 0.4, 0.6]	1	5,00	7	10	2	2	*earliness*	300.000
4	4	3	9	[0.1, 0.3, 0.2]	1	6,40	8,2	10,8	-1,8	-2	*Lateness*	400.000
5	4	4	8	[0.3, 0.7, 0.6]	1	8,60	11,4	13,8	-5,8	-6	*Lateness*	1.200.000
6	3	4	14	[0.3, 0.7, 0.9]	1	11,40	15,6	21	-7	-7	*Lateness*	2.100.000
7	3	5	17	[0.1, 0.1, 0.1]	1	12,70	15,9	21,3	-4,3	-5	*Earliness*	750.000
8	3	6	20	[0.5, 0.3, 0.6]	1	15,20	16,8	23,1	-3,1	-4	*Lateness*	600.000
9	5	6	18	[0.1, 0.1, 0.1]	1	16,70	17,3	23,6	-5,6	-6	*Lateness*	1.500.000
10	5	7	20	[0.1, 0.2, 0.1]	1	18,20	19,2	24,1	-4,1	-5	*Lateness*	1.250.000
11	3	8	26	[0.4, 0.3, 0.6]	1	21,60	23,4	27,7	-1,7	-2	*Lateness*	600.000
12	3	8	28	[0.1, 0.4, 0.1]	1	22,90	24,6	28	0	0	*On time*	-
TOTAL BIAYA												**10.180.000**

Keterangan: n_i dalam satuan unit, $t_{i1}, t_{i2}, t_{i3}, s_i, \Delta_i$ dalam satuan hari, $r_i, d, C_{i1}, C_{i2}, C_{i3}$ dalam satuan hari ke-

3.1. Contoh perhitungan menentukan *job earliness* pada kondisi eksisting

Berdasarkan Tabel 1 dapat diketahui bahwa *job* 1 akan memproses 4 unit komponen (n_1), pekerjaan tersebut datang pada saat hari ke-0 (r_1), dan direncanakan harus selesai pada saat tenggat (d_1) di hari ke 8. Jika pekerjaan selesai lebih dari hari ke-8, maka perusahaan akan dikenakan biaya pinalti per hari per komponen, sebaliknya jika pekerjaan selesai sebelum hari ke-8, maka perusahaan harus menyimpan komponen yang selesai diproses sehingga menimbulkan biaya simpan. Formula yang digunakan untuk menghitung waktu penyelesaian (C_{ijk}) adalah sebagai berikut:

$$(Cijk = max[C(i-1)jk, ri] + sij + ni\ x\ tijk), C(1-1)jk = 0 \quad (6)$$

$$(Ci(j+1)(k+1) = Cijk + ni + 1\ x\ t\ i(j+1)(k+1)), \quad (7)$$

$$\lceil \Delta_i = d_i - C_{irm} \rceil \quad (8)$$

$$\text{j} = 1,2,\dots,r;\ \text{k} = 1,2,\dots,m \quad (9)$$

Waktu Setup *job* ke-1 untuk proses ke-1 (s_{11}) adalah satu (1) satuan hari, dan waktu dari pengerjaan *job* 1 yang melewati tiga proses pada tiga mesin berbeda (t_{111}, t_{122}, t_{133}) masing-masing adalah (0,1; 0,3; 0,4) sehingga dapat dihitung:

A. Waktu penyelesaian **proses ke-1 *job* ke-1 pada mesin ke-1 (C_{111})** adalah 1,4 satuan waktu yang dihasilkan dari perhitungan waktu pekerjaan siap di lantai produksi ditambah dengan waktu setup kemudian ditambah dengan hasil perkalian antara jumlah komponen dengan waktu proses per komponen pada proses kesatu menggunakan formula (6) ($C_{111}=max[0,r_1]+s_{11}+n_1\ x\ t_{111}$), selanjutnya hitung waktu penyelesaian proses ke-2 *job* ke-1 pada mesin ke-2 (C_{122}),
B. Waktu penyelesaian **proses ke-2 *job* ke-1 pada mesin ke-2 (C_{122})** adalah 2,6 yang dihasilkan dari perhitungan waktu selesai proses ke-1 ditambah dengan hasil perkalian antara jumlah komponen dengan waktu proses per komponen pada proses kedua menggunakan formula (7) ($C_{122}=C_{111}+ n_2\ x\ t_{122}$),
C. Waktu penyelesaian **proses ke-3 *job* ke-1 pada mesin ke-3 (C_{122})** adalah 4,2 yang dihasilkan dari perhitungan waktu selesai proses ke-2 *job* ke-1 ditambah dengan hasil perkalian antara jumlah komponen dengan waktu proses per komponen pada proses ketiga menggunakan formula (7) ($C_{133}=C_{122}+ n_1\ x\ t_{133}$),
D. Berdasarkan perhitungan waktu proses tersebut maka dapat dihitung selisih antara saat *job* selesai di proses dengan saat tenggat yang telah direncanakan menggunakan formula (8) ($\Delta_i = d_1 - C_{133}$) adalah 3,8 yang artinya pekerjaan selesai lebih cepat dari saat tenggat sebanyak 3,8 ≈ 4 satuan hari (*earliness*). Pembulatan dilakukan karena satuan hari yang dihitung untuk menentukan perhitungan biaya merupakan bilangan bulat.

3.2. Contoh perhitungan menentukan *job earliness* pada *job* eksisting

Berdasarkan Tabel 1 dapat diketahui bahwa *job* 2 akan memproses 5 komponen (n_2), pekerjaan tersebut datang pada saat hari ke-1 (r_2), dan direncanakan harus selesai pada saat tenggat (d_2) di hari ke-2. Jika pekerjaan selesai lebih dari hari ke-2, maka perusahaan akan dikenakan biaya pinalti/hari/komponen, sebaliknya jika pekerjaan selesai sebelum hari ke-2, maka perusahaan harus menyimpan komponen yang selesai diproses sehingga menimbulkan biaya simpan/hari/komponen. Waktu *Setup job* ke-2 untuk mesin ke-1 (s_{21}) adalah satu (1) satuan hari, dan waktu dari pengerjaan *job* 2 yang melewati tiga proses pada tiga mesin berbeda (t_{211}, t_{222}, t_{233}) masing-masing adalah (0,1; 0,1; 0,1) sehingga dapat dihitung:

A. Waktu penyelesaian **proses ke-1 *job* ke-2 pada mesin ke-2 (C_{211})** adalah 2,9 satuan waktu yang dihasilkan dari perhitungan nilai maksimal antara saat pekerjaan datang ke lantai produksi dengan saat proses ke-1 selesai mengerjakan pekerjaan sebelumnya ditambah dengan waktu setup kemudian ditambah dengan hasil perkalian antara jumlah komponen dengan waktu proses per komponen pada proses kesatu menggunakan formula (6) ($C_{211}=max[C_{111},r_2]+s_{21}+n_2\ x\ t_{211}$),
B. Waktu penyelesaian **proses ke-2 *job* ke-2 pada mesin ke-2 (C_{222})** adalah 3,4 yang dihasilkan dari perhitungan waktu selesai proses ke-1 *job*-2 atau proses ke-2 *job* 1 (pilih yang terbesar) ditambah dengan hasil perkalian antara jumlah komponen dengan waktu proses per komponen pada proses kedua menggunakan formula (7) ($C_{222}= max[C_{122}, C_{211}] + n_2\ x\ t_{222}$),
C. Waktu penyelesaian **proses ke-3 *job* ke-2 pada mesin ke-3 (C_{233})** adalah 5,7 yang dihasilkan dari perhitungan waktu selesai proses ke-2 *job* ke-2 atau proses ke-3 *job* 1 (pilih yang terbesar) ditambah dengan hasil perkalian antara jumlah komponen dengan waktu proses per komponen pada proses ketiga menggunakan formula (7) ($C_{233}= max[C_{133}, C_{222}] + n_2\ x\ t_{233}$),

Berdasarkan perhitungan waktu proses tersebut, maka dapat dihitung selisih antara saat *job* selesai di proses dengan saat tenggat yang telah direncanakan menggunakan formula (8) ($\Delta_i = d_2 - C_{233}$) adalah -3,7 yang artinya pekerjaan selesai melewati saat tenggat sebanyak -3,7 ≈ -4 satuan hari (*lateness*).

3.3. Menghitung Total biaya kondisi eksisting

Total Biaya *lateness* dan *earliness* pada Tabel 1 dihitung menggunakan formula $Z1_i = |\Delta_i| \; x \; \beta_i \; x \; n_i$ jika status *job earliness dan* $Z2_i = |\Delta_i| \; x \; \alpha_i \; x \; n_i$ jika status *job* adalah *lateness.* Sedangkan biaya *re-scheduling* ($Z3 = (W+1) \; x \; \varphi_i$) pada kondisi eksisting masih 0 karena belum ada proses penjadwalan ulang. Asumsi rata-rata biaya simpan produk jadi (*hc*) adalah Rp.30.000/unit/hari dan rata-rata biaya pinalti akibat keterlambatan (*lc*) adalah Rp.50.000/unit/hari. Berdasarkan hal tersebut, maka perusahaan harus menanggung Total biaya penyimpanan dan pinalti keterlambatan sejumlah Rp. 10.180.000,-. Total biaya penyimpanan dan pinalti keterlambatan dihitung menggunakan formula:

$$Z = \sum_{i=1}^{N} Z1_i + \sum_{i=1}^{N} Z2_i + \sum_{i=1}^{N} Z3 \tag{10}$$

3.4. Solusi pemecahan masalah

Solusi penjadwalan dimulai dengan melakukan simulasi pada berbagai nilai panjang *window* (*D*) dengan delta pertambahan (ΔD) tertentu. Pada kasus ini, percobaan dimulai dengan menentukan inisial *window* $D = 2$, $\Delta D = 2$, dan *h*=30 hari kemudian menghitung nilai *W+1* dengan formula (5) dan Total Biaya (*Z*) dengan formula (1). Hasil perhitungan dan simulasi disajikan pada Tabel 2.

Tabel 2. Hasil Simulasi.

No.	D (hari)	HD (hari ke-)	W+1 (kali)	Pembulatan W+1 (kali)	$Z1_i$ (Rp)	$Z2_i$ (Rp)	$Z2_i$ (Rp)	Z (Rp)
1	*2*	*1*	30,0	30	360.000	10.950.000	3.000.000	14.310.000
2	4	2	15,0	15	780.000	9.400.000	1.500.000	11.680.000
3	6	3	10,0	10	780.000	9.300.000	1.000.000	11.080.000
4	8	4	7,5	8	990.000	9.000.000	800.000	10.790.000
5	*10*	*5*	6,0	6	780.000	9.300.000	600.000	10.680.000
6	12	6	5,0	5	780.000	9.400.000	500.000	10.680.000
7	14	7	4,3	4	780.000	9.400.000	300.000	10.480.000
8	16	8	3,8	4	780.000	9.400.000	300.000	10.480.000
9	18	9	3,3	3	780.000	5.300.000	300.000	9.980.000
. . .								
15	30	15	2,0	2	780.000	8.900.000	200.000	9.880.000*

Keterangan: D adalah *Window*, HD adalah *Half Window*, W+1 adalah Frekuensi *Re-Schedule*
*) solusi optimal

Pada setiap penjadwalan ulang, *job* disusun dengan menggunakan pendekatan EDD. Setiap baris pada Tabel 2 menunjukkan hasil simulasi yang dilakukan dimana setiap penjadwalan ulang *job* akan disusun berdasarkan pendekatan *early due date (EDD)*. Pada kondisi optimal, ditetapkan panjang *window* (*D*) adalah 30 hari, sehingga jadwal statis dilakukan hari ke-30 dan jadwal dinamis pada hari ke-15. Pada kondisi ini, perubahan jadwal dinamis menjadi jadwal statis (*HD*) dilakukan setiap 15 hari sekali sehingga menghasilkan frekuensi penjadwalan ulang (*W*) sebanyak satu kali. Total Biaya yang dihasilkan pada kondisi *D*=30 adalah sebesar Rp.9.880.000 yang terdiri atas biaya *earliness* sebesar Rp.780.000, biaya *lateness* sebesar Rp.8.900.000, dan biaya *re-schedule* sebesar Rp.200.000. Nilai Panjang *window* (*D*) pada kondisi optimal tidak selalu sama dengan asumsi Panjang *Window* (*D*) yang digunakan pada contoh kasus, hal tersebut tergantung pada parameter sistem yang dibahas. Hasil penjadwalan dan performansi dari kondisi optimal tersebut disajikan pada Tabel 3.

Tabel 3. Solusi untuk Penjadwalan dengan *D*= 30, *HD*=15, dan *W+1*=2 kali.

$L_{[i]}$	d_i	r_i	C_{i11}	C_{i33}	**Saat HD**	**EC (Rp)**	**LC (Rp)**	**RC (Rp)**	**TC (Rp)**
Job 1	8	1	1,4	4,2		480.000	-		
Job 2	2	1	2,9	5,7		-	1.000.000		
Job 3	12	3	5,00	10		300.000	-		
Job 4	9	3	6,40	10,8		-	400.000		
Job 5	8	4	8,60	13,8		-	1.200.000		
Job 6	14	4	11,40	21		-	2.100.000		
Job 7	17	5	12,70	21,3	15	-	750.000	100.000	
Job 9	18	6	14,20	21,8		-	1.000.000		
Job 8	20	6	16,70	23,6		-	600.000		
Job 10	20	7	18,20	24,1		-	1.250.000		
Job 11	26	8	21,60	27,7		-	600.000		
Job 12	28	8	22,90	28		-	-		
					30			100.000	
TOTAL						**780.000**	**8.900.000**	**200.000**	**9.880.000**

Pada Tabel 3 dapat dilihat bahwa pada delapan hari pertama (baris ke-1 sampai baris ke-7) *job* dijadwalkan mengikuti urutan kedatangan atau menggunakan metode FIFO. Urutan *job* adalah 1-2-3-4-5-6-7. Hal ini karena *job* masih dijadwalkan dalam jadwal statis sampai hari ke-15 (lihat kolom C_{i11}). Pada baris ke-8 sampai ke-12, terjadi penjadwalan ulang (saat *t*=15 di kondisi FIFO) untuk *job* yang ada di jadwal dinamis menjadi statis dengan menggunakan aturan prioritas EDD. Pada periode penjadwalan ulang pertama ini urutan *job* menjadi 6-7-9-8-11-12. Periode penjadwalan ulang ke-2 ketika seluruh *job* telah terjadwalkan. Urutan *job* secara keseluruhan adalah 1-2-3-4-5-6-7-9-8-10-11-12.

Selanjutnya, dilakukan analisis perbandingan antara solusi yang diterapkan perusahaan dan solusi yang diusulkan. Pada Tabel 1, perusahaan menggunakan metode *FCFS* dan menghasilkan total biaya sebesar Rp. 10.180.000,-. Sementara dengan menggunakan metode usulan (RTW) berbasis pada EDD diperoleh biaya sebesar Rp. 9.880.000,-. Terdapat efisiensi biaya sebesar Rp.300.000,- (3%). Nilai efisiensi relatif kecil karena disebabkan karena urutan jadwal yang digunakan perusahaan relatif tidak banyak berubah. Perubahan hanya ada pada jadwal *job* ke-8 dan ke-9 yang posisinya ditukar akibat adanya penjadwalan ulang pada hari ke-15. Hal ini juga menunjukkan bahwa kedatangan *job* ke perusahaan relatif terkendali dan perusahaan dinilai dapat mengatur kesepakatan penentuan saat tenggat pengiriman sehingga tidak terjadi antrian pada lantai produksi. Pada penelitian selanjutnya dapat dipertimbangkan untuk menjadwalan pekerjaan dengan pendekatan mundur agar dapat mengatisipasi pekerjaan terlambat dan meminimasi persediaan pada lantai produksi (Yusriski, dkk., 2021) dan mempertimbangkan penentuan jumlah sumber daya untuk mempercepat proses pekerjaan (Rahmawati, dkk., 2021)

Kesimpulan

Penelitian ini membahas masalah penjadwalan pada fasilitas *Flowshop* 3 mesin yang terdapat pada PT. ABC. Pola kedatangan *job* bersifat dinamis dan setiap *job* memiliki saat tenggat masing-masing. Penelitian ini mengusulkan metode *Rolling Time Window* (RTW) untuk memperbaiki performa jadwal perusahaan yang menggunakan aturan *First In Fist Out (FIFO*). Hasil penelitian menunjukkan adanya perbaikan performa perusahaan yang diukur dari total biaya *earliness, lateness* dan *re-schedule* (3%). Perbaikan ini diperoleh dengan menyarankan perusahaan melakukan *re-schedule* setiap 15 hari sekali dengan mengevaluasi panjang *window* per 30 hari (*D*=30 hari dan *W*=2 kali). Meskipun begitu, metode FIFO yang dipakai perusahaan masih dapat digunakan mengingat karakteristik *job* yang datang ke perusahaan masih terkendali. Selain itu, perusahaan dinilai mampu membuat kesepakatan dengan konsumen mengenai saat tenggat pengiriman yang membuat antrian *job* di lantai produksi menjadi tidak terlalu padat.

Perlu ada penelitian lebih lanjut menggunakan data-data *job* yang lebih lengkap untuk memastikan karakteristik *job* yang datang ke perusahaan. Data tersebut dapat dibuat distribusi kedatangan yang selanjutnya dapat dijadikan *input* simulasi percobaan guna memastikan saran penjadwalan usulan yang lebih baik.

Ucapan Terima Kasih

Penulis mengucapkan terima kasih kepada para pimpinan di lingkungan Fakultas Teknologi Manufaktur Unjani dan rekan-rekan dosen di program studi Teknik Industri, Fakultas Teknologi Manufaktur, Universitas Jenderal Achmad Yani yang telah membantu selama proses penelitian.

Daftar Notasi

α_i = ongkos keterlambatan (*lateness*) *job-i* per unit per satuan waktu
β_i = ongkos simpan (*earliness*) *job-i* per unit per satuan waktu

φ_i = ongkos penjadwalan ulang (*re-schedule*) *job-i* per frekuensi *re-schedule*
Δ_i = Rentang Waktu pengerjaan *job* i
C_{ijk} = saat selesai operasi ke-*j* dari *job-i* pada mesin *k*
a_i = *available time job i*
d_i = *due date job i*
D = variabel panjang *window*
h = Parameter untuk panjang horizon jadwal
i = indeks untuk *job*, $i = 1, 2, \ldots, n$
j = indeks untuk operasi, 1, 2, …, r
k = indeks untuk mesin, $k = 1, 2, \ldots, m$
m_i = indeks untuk mesin yang dipakai oleh operasi terakhir *job i*
M = set proses mesin yang terlibat
n_i = Jumlah komponen *job i* yang dikerjakan
N = set *job* yang terlibat
r_i = Waktu awal *job i* siap diproses pada sistem
R_i = set operasi untuk *job i*
S_{i1} = Waktu *setup job* ke-*i* sebelum masuk ke proses 1
t_{ijk} = *job* ke-*i* proses ke-*j* pada mesin-*k*
W = variabel banyaknya *window*.
Z = total biaya *eardiness*, *lateness* dan *re-schedule seluruh job*
$Z1_i$ = total biaya *earliness job* ke-i
$Z2_i$ = total biaya *lateness job* ke-i
$Z3_i$ = total biaya *re-schedule*

Daftar Pustaka

Ilhami, M. A. (2014). Pengembangan Model Penjadwalan Dinamis Flexible Flow Shop 3-Stages untuk Meminimasi Weighted Tardiness dengan Sistem Lelang. *Seminar Nasional IENACO-2014*, 363–373. http://publikasiilmiah.ums.ac.id/handle/123456789/4538

Mohan, J., Lanka, K., & Rao, A. N. (2019). A review of dynamic job shop scheduling techniques. *Procedia Manufacturing*, *30*, 34–39. https://doi.org/10.1016/j.promfg.2019.02.006

Nurainun, T. (2012). Penjadwalan Batch pada Flow Shop Dinamis untuk Meminimasi Biaya Produksi. *Prosiding Seminar Nasional ReSaTek II-2012*. 1–13.

Octanatry, D., Ilhami, M. A., & Herlina, L. (2015). Pengembangan Model Penjadwalan Dinamis Mesin Paralel dengan Sistem Lelang untuk Meminimasi Weighted Tardiness (Studi Kasus di PT . XYX). *Jurnal Teknik Industri Untirta*, *3*(3). 1-7

Rahmawati, S., Nasution, A.R.K., Nurholik, M.B. (2021). Penentuan Jumlah Tenaga Kerja pada Departemen Produksi di PT. XYZ dengan Metode Programa Dinamis untuk Meminimasi Biaya Tenaga Kerja. *Go-Integratif: Jurnal Teknik Sistem dan Industri*, 2(1). 24-36

Suharyanti, Y., & Halim, A. H. (2000). Model penentuan panjang time window pada penjadwalan job shop dinamik. *Jurnal Teknologi Industri*, *4*(4), 217–228.

Sun, D., & Lin, L. (1994). A dynamic job shop scheduling framework: A backward approach. *International Journal of Production Research*, *32*(4), 967–985. https://doi.org/10.1080/00207549408956982

Yusriski, R., Astuti, B., Ilham, M., & Zahedi. (2019). Integrated Batch Production and Multiple Preventive Maintenance Scheduling on A Single Machine to Minimize Total Actual Flow Time. *IOP Conference Series: Materials Science and Engineering*, *598*(1), 1–8. https://doi.org/10.1088/1757-899X/598/1/012083

Yusriski, R., Astuti, B., Sukoyo, Samadhi, T. M. A. A., & Halim, A. H. (2015). Integer Batch Scheduling Problems for a Single-Machine to Minimize Total Actual Flow Time. *Procedia Manufacturing*, *2*(February), 118–123. https://doi.org/10.1016/j.promfg.2015.07.021

Yusriski, R., Samadhi, T. M. A. A., & Halim, A. H. (2014). Batch Scheduling for a Single Machine with Forgetting Effect to Minimize Total Actual Flow Time. *Proceedings of the Asia Pacific Industrial Engineering & Management Systems Conference 2014. 1(October)*. 2055-2060

Yusriski, R., Sukoyo, S., Ari Samadhi, T. M. A., & Halim, A. H. (2015). Integer batch scheduling problems for a single-machine with simultaneous effects of learning and forgetting to minimize total actual flow time. *International Journal of Industrial Engineering Computations*, *6*(3), 365–378. https://doi.org/10.5267/j.ijiec.2015.2.005

Yusriski, R., Sukoyo, Samadhi, T. M. A. A., & Halim, A. H. (2016). An Integer Batch Scheduling Model for a Single Machine with Simultaneous Learning and Deterioration Effects to Minimize Total Actual Flow Time. *IOP Conference Series: Materials Science and Engineering*, *114*(1). https://doi.org/10.1088/1757-899X/114/1/012073

Yusriski, R., Sukoyo, Samadhi, T. M. A. A., & Halim, A. H. (2018). An integer batch scheduling model considering learning, forgetting, and deterioration effects for a single machine to minimize total inventory holding cost. *IOP Conference Series: Materials Science and Engineering*, *319*(1). https://doi.org/10.1088/1757-899X/319/1/012038

Yusriski, R., Astusi, B., Biksono, D., Wardani, T. (2021). A Single Macmhine Multi-Job Integer Batch Scheduling Problem with Multi Due Date to Minimize Total Actual Flow Time. *Decision Science Letter*, 10(3). 231-240

Yusriski, R., Pardiyono, R. Zahedi, Ramhawati S., dan Ramdhani A. (2023). Dynamic Flexible flow shop scheduling with theory of constraint and time window approach to minimize mean tardiness. *AIP Conference Proceedings, 2510*(1).