

BAB II

TINJAUAN PUSTAKA

2.1. Kajian Deduktif

2.1.1. Proses Produksi

Menurut Ahyari (2002) proses dijelaskan sebagai suatu cara, metode, maupun teknik untuk penyelenggaraan atau pelaksanaan dari suatu hal tertentu. Sedangkan pengertian produksi menurut Hadiprodjo dan Sudarmo (2000) adalah kegiatan untuk mengetahui penambahan manfaat atau penciptaan faedah, bentuk, waktu, dan tempat atas faktor-faktor produksi yang bermanfaat bagi pemenuhan konsumen. Dengan dua uraian tersebut maka dapat disimpulkan bahwa pengertian proses produksi yaitu suatu cara, metode, maupun teknik bagaimana penambahan manfaat atau penciptaan faedah, bentuk, waktu, dan tempat atas faktor-faktor produksi sehingga dapat bermanfaat bagi pemenuhan kebutuhan konsumen (Setiawati, 2014).

2.1.2. Penjadwalan

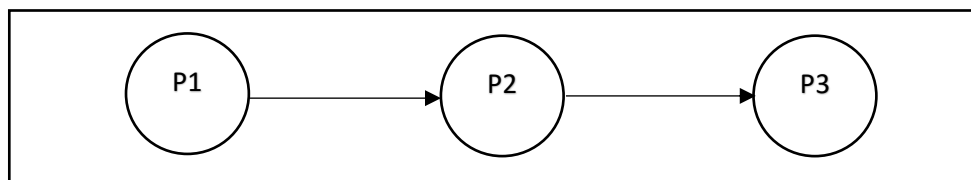
Penjadwalan menurut Yohanes (2014) diartikan sebagai pengurutan atau pengerjaan secara menyeluruh dalam suatu lintasan produksi yang dikerjakan pada beberapa buah mesin. Selanjutnya juga dijelaskan bahwa dalam masalah penjadwalan melibatkan pengerjaan beberapa komponen, mesin atau aktivitas mendasar yang sering disebut dengan istilah *job*.

Blazewics *et al.* (2007) menjelaskan bahwa penjadwalan terdiri dari tiga karakteristik, yaitu sejumlah $T = \{T_1, T_2, \dots, T_n\}$ dimana n adalah *job*, sejumlah $P = \{P_1, P_2, \dots, P_m\}$ dimana m merupakan mesin, dan sejumlah $R = \{R_1, R_2, \dots, R_s\}$ dimana s

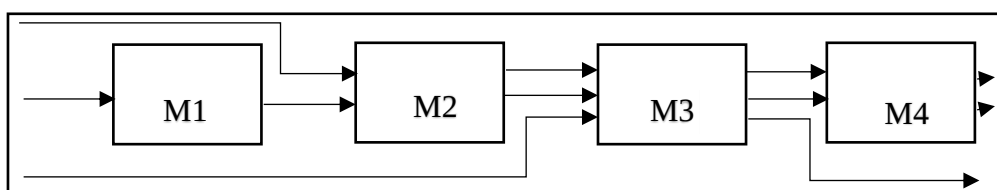
merupakan sumberdaya lainnya, yang kemudian diartikan sebagai penugasan sejumlah *job* untuk diproses disejumlah mesin dengan sumberdaya yang ada serta dengan batasan yang ditentukan.

Penjadwalan dilakukan ketika ada n buah *job* yang harus dikerjakan menggunakan mesin secara bergantian, sehingga diperlukan pengurutan pengerjaan *job* yang sesuai dengan kriteria performansi penjadwalan yang ditetapkan (Sukoyo *et al*, 2010). Dengan beberapa definisi tersebut maka dapat disimpulkan bahwa pengaplikasian penjadwalan diperlukan ketika terdapat n -*jobs* yang akan diproses di m -*machines*, agar dapat menemukan hasil yang optimal dalam produksi. Seperti tujuan dari penjadwalan menurut Narasimhan (1995) yaitu untuk mengoptimalkan penggunaan sumber daya yang ada sehingga tujuan produksi secara keseluruhan dapat terpenuhi. Menurut Kusiak (1990) teori *scheduling* diartikan sebagai sesuatu yang menentukan suatu *set parts* harus dikerjakan pada suatu *set machines* agar dapat meminimasi (maksimasi) suatu ukuran performansi.

Terdapat beberapa macam penjadwalan secara garis besar seperti *single machine*, *parallel machine*, *job shop*, *open shop*, dan selanjutnya adalah *flowshop*. Hal yang akan dibahas pada penelitian ini bersangkutan dengan penjadwalan *flowshop*, dan Baker (1974) menjelaskan definisi penjadwalan *flowshop* sebagai suatu pergerakan unit-unit yang terus-menerus melalui suatu rangkaian stasiun-stasiun kerja yang disusun berdasarkan produk. Gambar 2.1 dan 2.2 merupakan gambaran dari penjadwalan proses produksi di lantai *flowshop*.



Gambar 2.1 Diagram penjadwalan *flowshop*

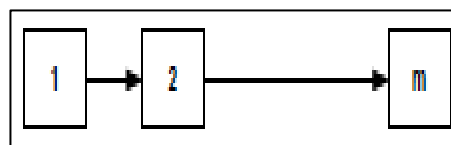


Gambar 2.2 Diagram penjadwalan *flowshop* menurut Kusiak (1990)

Sumber lain yang menjelaskan tentang *flowshop* didalamnya terdapat penjelasan mengenai beberapa macam pola pada *flowshop*, seperti yang diungkapkan Morton dan Pentico (1993) yaitu sebagai berikut :

1. *Pure Flow Shop*

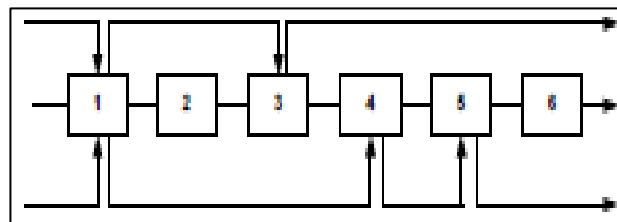
Semua job memerlukan satu operasi pada setiap mesin dan karena tiap *job* memerlukan jumlah mesin dan urutan yang sama. Gambar 2.3 menggambarkan pola *pure flowshop*.



Gambar 2.3 Pola *pure flowshop*

2. *Skip Flow Shop*

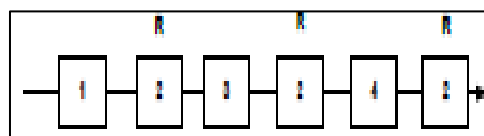
Pola melompat yaitu dimana mesin-mesin tertentu dapat dilompati oleh *job-job* tertentu. Gambar 2.4 menggambarkan pola *skip flowshop*.



Gambar 2.4 Pola *skip flowshop*

3. *Reentrant Flowshop*

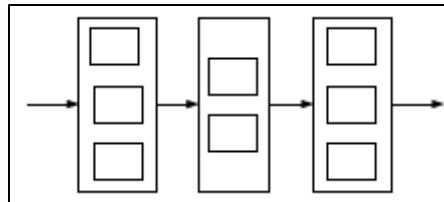
Pada pola ini beberapa mesin dapat dilalui lebih dari sekali, seperti yang ditunjukkan oleh Gambar 2.5 berupa pola *reentrant flowshop*.



Gambar 2.5 Pola *reentrant flowshop*

4. *Compound Flowshop*

Pola ini menunjukkan bahwa satu mesin dalam set mesin dapat digantikan oleh sekelompok mesin, kelompok mesin tersebut umumnya adalah mesin-mesin *parallel* atau jalur *batch* yang diikuti oleh mesin-mesin *parallel*. Gambar 2.6 menunjukkan pola *compound shop*.



Gambar 2.6 Pola *compound flowshop*

2.1.3. *Batch Splitting*

Batch dalam proses produksi dapat diartikan sebagai beberapa item yang akan dikerjakan dikelompokkan menjadi satu karena memiliki kriteria pemrosesan yang sama. Jumlahnya *batch* tidak ada batasan, selama mesin atau stasiun kerja tersebut dapat menampungnya. Namun tidak berarti seluruh item pengerjaan dijadikan satu *batch*, karena jumlah *batch* dapat dioptimalkan dengan menggunakan konsep *batch splitting* agar dapat memaksimalkan performansi sistem produksi.

Hal ini pernah diungkapkan oleh Han *et al.* (2013) sebagai konsep *batch splitting* yaitu dimana *lot size* produksi yang besar dibagi menjadi beberapa *sub-lot* yang berukuran lebih kecil maka proses operasi di tiap mesin atau stasiun kerja akan mengalami percepatan. Hal yang sama juga diungkapkan oleh Kropp dan Smunt (1990) bahwa *lot splitting* merupakan teknik untuk mempercepat alur pengerjaan dengan membagi *job lots* menjadi *sublots*. Mempercepat alur proses pengerjaan juga merupakan salah satu kriteria memaksimalkan performansi sistem produksi.

Terdapat salah satu contoh penggunaan *batch splitting model* yang dilakukan oleh Kempf *et al.* (1998) untuk mengatur alokasi jumlah *batch* menjadi lebih optimal untuk *scheduling a semiconductor burn-in-oven*, yang mana objektifnya adalah meminimalkan *makespan*. Pembuatan model tersebut dimulai dengan membangun fungsi objektif untuk model matematis *batch splitting* nya, yang dijelaskan dalam Persamaan (2.1)

$$\sum_{i=1}^n \sum_{j=1}^n ip x_{ij} \quad (2.1)$$

Setelah membentuk fungsi objektif atau tujuan, hal berikutnya yang perlu dilakukan adalah membuat batasan-batasan yang mempengaruhi tujuan. Untuk mencapai objektif meminimasi *makespan*, terdapat beberapa batasan yang diperlukan agar model dapat sedekat mungkin dengan kondisi sistem nyatanya. Beberapa batasan tersebut dijelaskan pada Persamaan (2.2) sampai (2.4).

$$\sum_{j=1}^n a_j^k \cdot x_{ij} \leq T_k \quad \text{for } k = 1, \dots, m; i = 1, \dots, n \quad (2.2)$$

$$\sum_{k=1}^m \sum_{j=1}^n a_j^k \cdot x_{ij} \leq B \quad \text{for } i = 1, \dots, n \quad (2.3)$$

$$\sum_{i=1}^m x_{ij} = 1 \quad \text{for } j = 1, \dots, n \quad (2.4)$$

dengan keterangan,

ip = completion time batch i

x_{ij} = job j di batch i

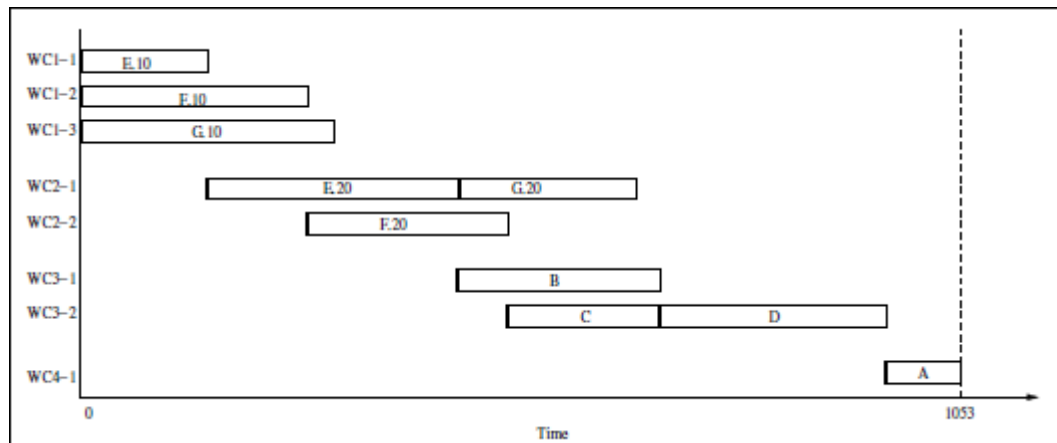
a_j = job size untuk job j

T_k = jumlah type k boards yang tersedia

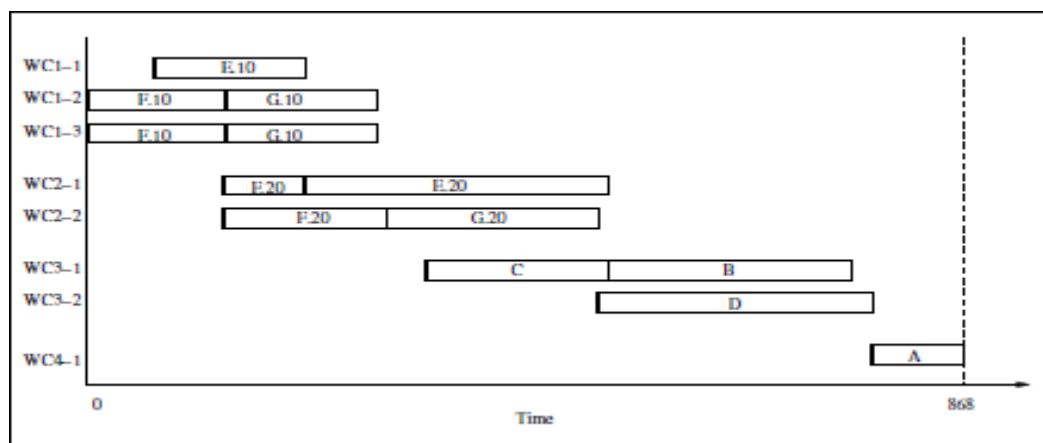
B = oven (machine) capacity

Persamaan (2.2) memastikan bahwa jumlah *boards* untuk tiap *type* k yang dibutuhkan oleh setiap *batch* tidak melebihi jumlah *boards* yang tersedia. Persamaan (2.3) digunakan untuk memastikan jumlah *boards* pada suatu batch tidak melebihi kapasitas B . Persamaan (2.4) menjelaskan setiap *job* berjumlah satu *batch*.

Model *batch splitting* dapat dibuat untuk meminimasi tujuan lainnya, dengan konsep yang sama yaitu menentukan fungsi objektif dari model tersebut lalu membuat batasan-batasannya, namun dalam hal ini yang difokuskan adalah meminimasi *makespan* seperti contoh tersebut. Dastidar dan Nagi (2007) membuat ilustrasi dalam sebuah *chart* atau grafik pada Gambar 2.7 dan 2.8, yang menggambarkan konsep *batch splitting* untuk mencari ukuran *batch* yang optimal agar dapat meminimasi *makespan*..



Gambar 2.7 Gantt chart schedule untuk proses *without batch splitting*



Gambar 2.8 Gantt chart schedule untuk proses *with batch splitting*

Dari grafik di atas dapat diketahui bahwa ukuran *batch* pada beberapa proses telah dibagi menjadi beberapa *batch* yang berukuran lebih kecil, yang berefek pada *total completion time (makespan)* yaitu terdapat penurunan waktu dibanding kondisi *without batch splitting*.

2.1.4. Scheduling Models for N-Operations and M-Machines

Seperti yang telah dijelaskan sebelumnya bahwa melakukan penjadwalan merupakan hal yang penting bagi perusahaan terkhusus manufaktur, agar produksi lebih optimal maka metode yang akan digunakan adalah pemodelan penjadwalan produksi. Salah satu contoh *Scheduling models* dalam penelitian ini menggunakan referensi yang bersumber dari buku karangan Kusiak (1990). Didalam bukunya dijelaskan mengenai model yang dapat

digunakan untuk masalah penjadwalan n -operations atau n -jobs dengan adanya batasan *precedences* yang dikerjakan di m -machines.

Didalamnya juga dijelaskan beberapa notasi yang digunakan untuk membentuk algoritma, yaitu sebagai berikut :

n = nomer *parts*

n_i = nomer operasi dari *part* P_i , $i = 1, \dots, n$

m = nomer mesin

R_i = sepasang operasi $[k, l]$ untuk *part* P_i , dimana proses k *precedes* proses l , $i = 1, \dots, n$

Q_i = sepasang operasi $[k, l]$ untuk *part* P_i , dimana proses k dan proses l dapat dilakukan kapanpun, $i = 1, \dots, n$

I_i = sepasang operasi tanpa batasan *precedences*, $i = 1, \dots, n$

N_p = sepasang operasi yang akan dikerjakan di mesin p , $p = 1, \dots, m$

t_{il} = waktu proses dari operasi l untuk *part* P_i , $l = 1, \dots, n$, $i = 1, \dots, n$

M = nomer yang memiliki angka positif besar

x_{ik} = lama waktu menyelesaikan operasi k dari *part* P_i , $k = 1, \dots, n$, $i = 1, \dots, n$

$x_{[i]}$ = lama waktu menyelesaikan seluruh operasi (hingga proses terakhir) dari *part* P_i , $i = 1, \dots, n$

$$y_{kl} = \begin{cases} 1, & \text{jika operasi } k \text{ precedes operasi } l \\ 0, & \text{kebalikannya} \end{cases}$$

$$[k, l] \in Q_i, i = 1, \dots, n$$

$$z_{kl} = \begin{cases} 1, & \text{jika operasi } k \text{ precedes operasi } l \\ 0, & \text{kebalikannya} \end{cases}$$

$$\text{untuk semua } [k, l] \in N_p, p = 1, \dots, m$$

Selanjutnya dijelaskan pembuatan *scheduling model* yang diawali dengan membentuk objektif yaitu meminimasi *makespan*, dan dijelaskan pada Persamaan (2.5).

$$\min \sum_{i=1}^n x[i] \quad (2.5)$$

Untuk mencapai nilai objektif yang diinginkan, maka perlu dibentuk batasan-batasan yang dapat menggambarkan keadaan sistem nyatanya, dan dijelaskan pada Persamaan (2.6) sampai (2.12),

$$x_{il} - x_{ik} \geq t_{il} \quad \begin{array}{l} \text{for all } [k, l] \in R_i, \\ \text{for all } i \end{array} \quad (2.6)$$

$$x_{il} - x_{ik} + M(1 - y_{kl}) \geq t_{il} \quad \begin{array}{l} \text{for all } [k, l] \in Q_i, \\ \text{for all } i \end{array} \quad (2.7)$$

$$x_{ik} - x_{il} + My_{kl} \geq t_{ik} \quad (2.8)$$

$$x_{jl} - x_{ik} + M(1 - z_{kl}) \geq t_{jl} \quad \begin{array}{l} \text{for all } [k, l] \in N_p, \\ \text{for all } p \end{array} \quad (2.9)$$

$$x_{ik} - x_{jl} + Mz_{kl} \geq t_{jk} \quad i \neq j \quad (2.10)$$

$$x_{ik} \geq \begin{cases} t_{ik} & \text{for all } [i, k] \in I_i, i = 1, \dots, n \\ 0 & \text{for all other } i, k \end{cases} \quad (2.11)$$

$$y_{kl} = 0, 1 \quad \text{for all } [k, l] \in Q_i, \text{ for all } i \quad (2.12)$$

$$z_{kl} = 0, 1 \quad \text{for all } [k, l] \in N_p, \text{ for all } p$$

Persamaan (2.6) menunjukan bahwa operasi tiap *part* di proses sesuai dengan *precedences* yang ditentukan. Persamaan (2.7) dan (2.8) untuk memastikan bahwa dua operasi yang berasal dari *part* yang sama tidak bisa diproses bersamaan. Persamaan (2.9) dan (2.10) untuk memastikan bahwa satu mesin tidak dapat memproses lebih dari satu *part* di waktu yang bersamaan. Persamaan (2.11) dan (2.12) untuk menjelaskan *nonnegativity and integrality, respectively*.

Persamaan rumus di atas akan sangat membantu dalam membangun model untuk *scheduling problem* pada lingkungan *flowshop*. Hal ini disebabkan lingkungan *flowshop*

yang mana prosesnya membentuk satu jalur akan sangat terpengaruh oleh adanya *precedences* disetiap langkah proses produksinya dan pada persamaan rumus tersebut terdapat contoh pembuatan batasan untuk proses yang memiliki *precedences*.

2.2. Kajian Induktif

2.2.1. Penelitian Terdahulu

Penelitian mengenai *scheduling* sudah banyak dilakukan sekarang ini, berbagai macam metode juga telah digunakan untuk menyelesaikan permasalahan yang berkaitan. Salah satu metodenya adalah menggunakan algoritma *scheduling model*. Objektif yang sering digunakan dalam beberapa *scheduling problem* adalah minimasi *makespan*.

Beberapa jurnal yang membahas permasalahan *scheduling* pada rantai produksi *flowshop* yang digunakan sebagai referensi, adalah seperti jurnal penelitian dari Semančo dan Modrák (2012) yang meneliti minimasi *makespan* menggunakan algoritma *scheduling* bernama MOD untuk solusi *general flowshop problem* dengan $m > 2$. Pada penelitiannya algoritma tersebut dibandingkan dengan empat metode lainnya yaitu NEH, *Palmer's Slope Index*, CDS dan *Gupta's algorithm*. Dari kesimpulannya dijelaskan bahwa algoritma *scheduling* dapat digunakan untuk penyelesaian *n-jobs* dan *m-machines* pada *flowshop problem*, yang dapat menghasilkan solusi terbaik.

Gangraj dan Nahavandi (2014), didalamnya dijelaskan mengenai minimasi *makespan* untuk *batch sizing* dan *scheduling problem* pada *flexible flowshop*. *Batch sizing* dalam penelitiannya mengarah pada pengurangan atau penambahan jumlah *batch* yang dapat meminimasi *makespan*, dikarenakan limit pada mesin yang memaksakan jumlah order dari kostumer di bagi ke dalam *batches*. Metode yang digunakannya adalah *mathematical model* dan MSGA (*mixed simulated genetic algorithm*). Pada perhitungan *mathematical model* terdapat beberapa parameter digunakan seperti *batch*, *processing time*, *dependent setup time*, dan lain-lain. Serta untuk *decision variable* dibuat menjadi $\text{Min } Z = \max \{C_{ijm}\}$. Disimpulkan bahwa *matemactical model* mendukung minimasi *makespan*.

Dati *et al.* (2009) meneliti tentang *batch scheduling* pada *2-machines flowshop*, dengan mengasumsikan *unit processing time jobs*, *batch availability*, dan *machine-dependent setup time*. Dengan objektif untuk mencari alokasi *job* untuk *batch* yang dapat minimasi *makespan*. Pada penelitiannya telah dibentuk dua versi model yaitu *relaxed version* dan *integer version*. Terdapat beberapa perbedaan hasil *makespan* diantara kedua versi yang bergantung pada jumlah *batch* (k) nya, yang dijelaskan dalam sebuah grafik untuk $S_1 < S_2$. Dari grafik tersebut terlihat bahwa *integer version* menghasilkan *makespan* lebih besar dibanding *relaxed version* untuk beberapa jumlah *batch* (k).

Gerstl *et al.* (2014), meneliti mengenai *two-stage flexible flowshop* dimana terdapat beberapa *parallel identical machines* pada *stage one* dan *single machine* pada *stage two*. Beberapa asumsi yang diterapkan adalah *identical jobs*, *option of batching*, *setup time* untuk memproses *batch* baru, serta dapat menggunakan satu *subset available machine*. Didalam kesimpulannya dijelaskan bahwa permasalahan yang diangkat pada penelitian ini yaitu minimasi *makespan* telah terpecahkan dengan dibentuknya sebuah *solution algorithm* yang lebih efisien, dan penulis atau peneliti juga mempertimbangkan proses *batching* sebagai langkah optimal.

Bukchin *et al.* (2002) melakukan penelitian untuk upaya meminimasi *total flow-time*. Penulis tersebut membangun sebuah model matematis yang dapat menyelesaikan permasalahan *lot splitting scheduling* pada lingkungan *2-machines flowshop* dengan *detached setups* dan *batch availability*. Di akhir kesimpulannya dijelaskan bahwa penelitian ini menghasilkan solusi untuk *non-equal subplot sizes* yang sangat efisien yang dinamakan SMB (*Single Machine Bottleneck*) untuk menghadapi permasalahan tersebut.

Selanjutnya adalah jurnal penelitian yang membahas *batching* dan *scheduling n-jobs* di *flowshop* dengan $m\text{-machine} \geq 2$, yang dilakukan oleh Ng dan Kovalyov (2007). Metode yang digunakan adalah *dynamic programming algorithm*, dengan objektif untuk minimasi maksimum *job completion time*. Diakhir penelitiannya diungkapkan bahwa telah ditemukan *optimal schedule* untuk *job sequences*-nya, serta terdapat peningkatan dibanding metode yang digunakan sebelumnya.

Beberapa jurnal lainnya yang juga membahas *scheduling* untuk *flowshop problem* adalah sebagai berikut Fink dan Volß (2001) tentang minimasi *total processing time* pada *continuous flowshop scheduling problem*. Cheng *et al.* (2015) tentang minimasi *weight sum of makespan* dan *total completion time* pada *two-machine flowshop scheduling problem with deteriorating jobs*. Samad *et al.* (2014) meneliti *scheduling flowshop batch process* dengan *zero-wait* dan *no-intermediate storage*, dengan objektif optimal *makespan*. Nurainun *et al.* (2016) meneliti tentang permasalahan *batch scheduling model* untuk lingkungan *dynamic flowshop*, dengan menggunakan metode bernama *Mix Integer Non Linear Programing* (MINLP) untuk tujuan minimasi *total production cost*.

Saluja dan Jain (2014) membahas optimasi *makespan* untuk *m-stages* dan *n-jobs flexible flowshop* menggunakan *generatic algorithm* (GA). Jayakumar *et al.* (2016) meneliti *scheduling prblem* pada *2-machines* dan *n-jobs flowshop*, menggunakan pendekatan *Johnson's algorithm* dengan *makespan* sebagai objektif. Sanowar *et al.* (2014) tentang minimasi *makespan* untuk permasalahan *scheduling* pada *4-jobs* dan *10-machines flowshop* dengan menggunakan tiga metode yaitu *Palmer's heuristic*, *CDS heuristic* dan *NEH heuristic*.

2.2.2. Penelitian yang Dilakukan

Pada penelitian ini yang berbeda dengan penelitian sebelumnya adalah penggunaan model matematis untuk *batch scheduling* yang digunakan untuk menyelesaikan permasalahan *multi stage* dan *m-machines > 2* pada rantai produksi *flowshop* dengan parameter seperti waktu proses mesin, proses *batch splitting*, *zero setup time*, dan tanpa pengaruh *due date*.

Penggabungan model penjadwalan matematis dengan konsep *batch splitting*, maka akan dapat membentuk model yang dapat mengoptimalkan penjadwalan *batch-batch (batch scheduling)*, yang sesuai dengan lingkungan *flowshop*. Model inilah yang akan dibangun pada penelitian ini dan menjadi fokus penelitian. Dimana objektif yang dicari dari penelitian ini adalah ukuran *batch* yang optimal pada setiap mesin setelah dilakukan *batch splitting* agar dapat meminimasi *total completion time (makespan)*.