ISSN 2088-4842 / 2442-8795 OPTIMASI SISTEM INDUSTRI

ALGORITMA PENENTUAN UKURAN *BATCH* INTEGER PADA PENJADWALAN *FLOWSHOP* SATU MESIN

Hadigufri Triha[1], Ahmad Syarifuddin Indrapriyatna[1], Jonrinaldi[1], Berry Yuliandra[2]

[1]Jurusan Teknik Industri, Fakultas Teknik, Universitas Andalas, Padang

[2]Jurusan Teknik Mesin, Fakultas Teknik, Universitas Andalas, Padang

Email: hadigufri@gmail.com (korespondensi)

Abstract

Scheduling is an important area of production planning and control. Scheduling is required to produce an existing job by allocating existing resources in the proper execution order. Production schedules arranged effectively and efficiently to maximize resources utility, minimize waiting and idle time and increase productivity. Flowshop batch scheduling model for one machine that take inventory and quality cost into account has been developed by Indrapriyatna et al (2007a). However, the model yet effective in converting the results of batch size into integers. This study tried to resolve this problem by using modification of Branch and Bound Algorithm approach.

Keywords: *Scheduling, batch, flowshop, Branch and Bound Algorithm*

Abstrak

Salah satu area penting dari perencanaan dan pengendalian produksi adalah penjadwalan. Penjadwalan diperlukan untuk memproduksi job yang ada dengan mengalokasikan sumber daya yang ada pada urutan pengerjaan komponen yang tepat. Pengaturan jadwal produksi yang efektif dan efisien akan memaksimalkan utilitas sumber daya, meminimumkan waktu tunggu dan waktu menganggur serta meningkatkan produktivitas. Model penjadwalan batch flowshop untuk 1 mesin yang mempertimbangkan biaya simpan dan biaya kualitas telah dikembangkan oleh Indrapriyatna et al (2007a). Akan tetapi model tersebut masih belum efektif dalam mengkonversikan ukuran batch ke dalam bilangan integer. Penelitian ini mencoba menyelesaikan permasalahan ini dengan menggunakan pendekatan Algoritma Branch and Bound Modifikasi.

Kata kunci: *Penjadwalan, batch, flowshop, Algoritma Branch and Bound*

1. PENDAHULUAN

Perencanaan dan pengendalian produksi merupakan aktivitas internal yang penting bagi perusahaan manufaktur. Tujuan dari perancanaan dan pengendalian produksi adalah mengefektifkan utilisasi sumber daya sambil memenuhi keinginan konsumen dan menciptakan keuntungan bagi investor [1]. Berbagai *input* digunakan untuk mencapai tujuan tersebut, antara lain: peramalan penjualan, program produksi, rencana produksi, penjadwalan produksi, *job order*, laporan penyelesaian, data persediaan, deskripsi produk, gambar produk, spesifikasi produk, deskripsi proses, estimasi biaya, standar pekerjaan, pesanan, tuntutan pembelian, pesanan pembelian, laporan penerimaan, laporan inspeksi penerimaan, laporan inspeksi proses, laporan inspeksi produk akhir dan laporan pengiriman [2]. Berdasarkan berbagai jenis dokumen tersebut, input dari penjadwalan meliputi penjadwalan produksi, *job order*, laporan penyelesaian, data persediaan, standar pekerjaan, tuntutan pembelian, pesanan pembelian dan laporan pengiriman. Aktivitas penjadwalan meliputi ruang lingkup yang cukup luas dalam perencanaan dan pengendalian produksi.

Penjadwalan merupakan salah satu aktivitas penting dari perencanaan dan pengendalian produksi. Aktivitas ini diperlukan untuk memproduksi *job* yang

ada dengan mengalokasikan sumber daya (mesin, operator, dan kebutuhan material) secara efisien pada urutan pengerjaan komponen yang tepat.

Pengaturan jadwal produksi yang baik akan memaksimalkan utilitas sumber daya perusahaan. Melalui penjadwalan yang efektif dan efisien, waktu tunggu dan waktu menganggur material dapat diminimumkan, sehingga akan mempersingkat waktu proses sebuah *job*. Sementara itu, jika penjadwalan dilakukan secara tidak optimal, maka dapat menyebabkan:

1. Pekerja maupun mesin menganggur karena tidak ada pekerjaan untuk dikerjakan sehingga sumber daya yang tersedia akan terbuang percuma karena tidak dimanfaatkan.
2. Meningkatnya persediaan komponen *work-in-process* karena tidak ada mesin yang *available*.

Oleh karena produktivitas merupakan rasio nilai produk yang dihasilkan dengan nilai sumber daya yang digunakan dalam produksi, maka pengaturan jadwal yang optimal akan meningkatkan produktivitas perusahaan [1].

Penjadwalan *batch* digunakan untuk menentukan ukuran dan urutan *job* yang telah dibagi menjadi beberapa bagian (*batch*). Halim dan Ohta (1993), Halim dan Ohta (1994), Halim *et al*. (2001), serta Bukchin *et al*. (2002) membahas penentuan urutan dan ukuran *batch* yang merupakan ukuran "*job*" [3,4,5]. Fokus utama dari jenis penjadwalan ini adalah bagaimana menentukan ukuran *batch* (*batching*) dan urutan pemrosesan *batch* yang dihasilkan (*sequencing*).

Indrapriyatna *et al* (2007a) telah mengembangkan model penjadwalan *batch flowshop* untuk 1 mesin dengan mempertimbangkan biaya simpan dan biaya kualitas [6]. Biaya simpan pada model tersebut telah dihitung dengan membedakan jenis persediaan *work-in-process* dan *finished batch*.

Permasalahan utama dalam penjadwalan *batch* adalah ukuran dari *batch* harus berupa bilangan *integer*. Model yang yang dikembangkan Indrapriyatna *et al* (2007b) mengusulkan tiga metode untuk mengatasi permasalahan tersebut, yaitu Metode Jumlah-Desimal-Atas, Metode Jumlah-Desimal-Bawah, dan Metode Pembulatan [7]. Akan tetapi diantara ketiga metode tersebut tidak ada yang selalu memberikan nilai total biaya terkecil pada semua set data. Oleh karena itu penelitian ini akan mencoba menggunakan pendekatan yang berbeda dalam memecahkan permasalahan *integer* ini, yaitu dengan menggunakan Algoritma *Branch and Bound*.

2. TINJAUAN PUSTAKA

2.1. Penjadwalan

Penjadwalan memiliki definisi yang cukup bervariasi. Beberapa diantaranya adalah:

1. Proses pengalokasian sumber daya dalam jangka waktu tertentu untuk melakukan sejumlah pekerjaan [8]
2. Proses meramalkan sumber daya yang akan digunakan suatu pekerjaan dan penentuan waktu awal pengerjaan dengan tepat (Carlier dan Chretienne (1988) didalam T'kindt et al. (2006)) [9]
3. Pengalokasian sumber daya yang terbatas melewati suatu horizon waktu (Pinedo (1995) didalam T'kindt et al. (2006)) [9]

Berdasarkan definisi-definisi tersebut dapat ditarik kesimpulan bahwa penjadwalan adalah sebuah teknik untuk penugasan sumber daya untuk menyelesaikan pekerjaan dalam rentang waktu yang layak. Penjadwalan merupakan tahapan akhir dari perencanaan produksi dan merupakan fase yang menjembatani antara rencana dan eksekusi.

Proses penjadwalan yang baik harus mampu mencapai tujuan spesifik suatu pekerjaan secara realistis. Untuk mewujudkan hal ini, terdapat beberapa kriteria yang perlu diperhatikan dalam proses penjadwalan:

1. Pekerjaan
2. Kendala potensial
3. Sumber daya yang tersedia
4. Fungsi tujuan

Penjadwalan pada dasarnya merupakan proses pengambilan keputusan untuk mengoptimalkan satu atau lebih kriteria untuk mencapai tujuan akhir pekerjaan. Proses penjadwalan pada dasarnya tidak bisa dipisahkan dari komponen biaya simpan dan biaya kualitas. Oleh karena itu, kedua komponen biaya tersebut seharusnya ikut diperhatikan pada saat melakukan penjadwalan.

2.2. Hubungan Penjadwalan dengan Biaya Simpan

Biaya simpan adalah semua biaya yang terkait dengan persediaan. Biaya simpan memiliki keterkaitan langsung dengan jadwal produksi, atau lebih tepat dikatakan bahwa jadwal produksi yang diterapkan akan berpengaruh terhadap besar atau kecilnya biaya simpan. Hubungan ini akan tampak nyata pada sistem produksi yang memiliki kapasitas terbatas. Jika *due date* relatif ketat terhadap kapasitas pabrik, maka jadwal produksi yang bisa membuat waktu selesai seluruh komponen tepat pada saat *due date* tidak mungkin dilakukan. Konsekuensi dari hal ini adalah sebagian komponen diproduksi lebih awal sehingga diselesaikan lebih cepat dari *due date*, sehingga komponen tersebut harus menunggu penyelesaian komponen lain sebelum dikirimkan [4].

Herjanto (2008) mengemukakan beberapa elemen biaya simpan, antara lain [10]:

1. Biaya sewa gudang,
2. Biaya administrasi pergudangan,
3. Gaji pelaksana pergudangan,
4. Biaya listrik,
5. Biaya modal yang tertanam dalam persediaan,
6. Biaya asuransi,
7. Biaya kerusakan, kehilangan atau penyusutan barang selama penyimpanan.

2.3. Hubungan Penjadwalan dengan Biaya Kualitas

Proses produksi selalu memiliki variasi alami yang terjadi secara acak. Variasi alami ini dapat mempengaruhi kualitas produk yang dihasilkan. Oleh karena itu, untuk memastikan agar produk yang dikirimkan benar-benar sesuai dengan keinginan pelanggan, maka produk yang dihasilkan perlu dibandingkan terlebih dahulu dengan standar baku. Aktivitas ini disebut sebagai pengendalian kualitas.

Pelaksanaan aktivitas pengendalian kualitas menyebabkan munculnya biaya kualitas. Biaya kualitas adalah semua biaya yang terkait dengan penyesuaian produk atau pelayanan yang diberikan oleh suatu perusahaan berdasarkan syarat-syarat yang diminta oleh pelanggan. Biaya kualitas berhubungan dengan proses penciptaan, identifikasi, perbaikan dan pencegahan kerusakan. Berdasarkan Model Juran, biaya kualitas dapat dibagi ke dalam tiga kategori utama [11]:

1. Biaya pencegahan (*Cost of Prevention*)
2. Biaya pemeriksaan/ penilaian
3. Biaya Kegagalan

Model penjadwalan *batch flowshow* yang mempertimbangkan biaya kualitas telah pernah dikembangkan oleh Halim (2001). Pada model tersebut biaya kualitas ditunjukkan melalui penerapan *acceptance sampling* pada proses akhir dan pada saat konsumen menerima produk [5].

Biaya kualitas dalam konteks penelitian ini dikelompokkan sebagai berikut:

1. Biaya pemeriksaan sampel
   Jenis biaya ini terkait dengan aktivitas pengujian, evaluasi atau pengukuran agar setiap komponen yang dihasilkan mampu memenuhi spesifikasi yang diinginkan. Biaya ini meliputi:
   a. Biaya untuk melakukan pemeriksaan sampel.
   b. Biaya penyimpanan komponen selama pemeriksaan sampel.
2. Biaya Kegagalan Internal
   Jenis biaya ini muncul ketika sejumlah komponen yang diproduksi tidak memenuhi spesifikasi kualitas sebelum komponen tersebut dikirimkan kepada konsumen. Biaya ini meliputi:
   a. Biaya pemeriksaan komponen yang tidak termasuk ke dalam sampel pemeriksaan (pemeriksaan 100%).
   b. Biaya penyimpanan komponen selama pemeriksaan 100%.

c. Biaya pengerjaan ulang komponen yang tidak memenuhi spesifikasi kualitas.
d. Biaya penyimpanan komponen selama pengerjaan ulang.

3. Biaya Kegagalan Eksternal
   Jenis biaya ini muncul ketika sejumlah komponen yang diproduksi tidak memenuhi spesifikasi kualitas dan diketahui setelah produk diserahkan kepada konsumen. Biaya ini meliputi:
   a. Biaya untuk melakukan pemeriksaan 100%.
   b. Biaya penyimpanan komponen selama pemeriksaan 100%.
   c. Biaya pengerjaan ulang seluruh komponen yang tidak memenuhi standar kualitas.
   d. Biaya penyimpanan komponen selama pengerjaan ulang.
   e. Biaya komplain konsumen.

2.4. Teori Optimasi

Optimasi dapat didefinisikan sebagai proses pencarian nilai minimum atau maksimum dari suatu fungsi secara sistematis melalui pemilihan nilai variabel untuk memberikan solusi yang optimal. Bentuk umum dari permasalahan optimasi terdiri atas fungsi tujuan dan batasan-batasan yang berada dalam ruang dari variabel-variabel keputusan. Permasalahan minimasi (pencarian nilai minimum) mensyaratkan bahwa fungsi tujuan berbentuk fungsi *convex*, sementara permasalahan maksimasi (pencarian nilai maksimum) mensyaratkan fungsi tujuan berbentuk fungsi *concave*. Perbedaan kedua fungsi ini dapat dilihat pada Gambar 1.

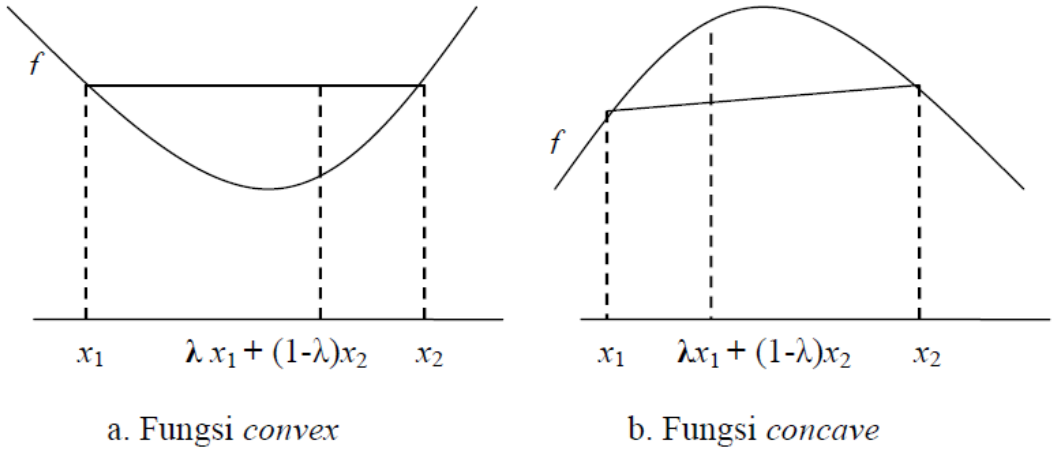
Model penjadwalan yang dikembangkan oleh Indrapriyatna *et al* (2007a) menggunakan fungsi tujuan minimasi, oleh karena itu fungsi tujuan dari model tersebut berbentuk *convex* [6].

Pada $f: S \rightarrow E_i$
Fungsi *f* dikatakan convex pada *S* jika memenuhi:
$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$
Untuk setiap $x_1, x_2 \in S$ dan untuk setiap $\lambda \in (0,1)$.

2.5. Model Penjadwalan Batch

Indrapriyatna *et al*. (2007a) telah mengembangkan model penjadwalan *batch* dengan memperhitungkan biaya simpan *work-in-process part* dan *finished-part.* Metode penjadwalan dalam model tersebut. Beberapa asumsi dasar yang digunakan oleh Indrapriyatna *et al.* (2007a) antara lain [6]:

1. *Job* yang diproses memiliki *routing* sama.
2. Penjadwalan dilakukan berdasarkan dua keputusan, penentuan ukuran *batch* dan penentuan urutan pemprosesan *batch*.



Gambar 1. Perbedaan antara: (a) Fungsi *convex* dan (b) Fungsi *concave*
(sumber: Bazaraa et al (2007)) [12]

3. Penjadwalan dilakukan secara *backward*.
4. Kriteria penjadwalan adalah total biaya minimum, yang merupakan turunan dari minimasi total waktu tinggal aktual. Biaya yang diperhatikan adalah biaya simpan dan kualitas.
5. Variabel keputusan yang digunakan adalah jumlah, ukuran dan jadwal produksi *batch*.
6. Aktivitas perawatan meliputi inspeksi, *restorasi dan preventive maintenance.*

Notasi-notasi yang digunakan dalam model tersebut antara lain:

Indeks
i : nomor *batch*, i = 1, 2 ... N

Variabel Keputusan
$B_{[i]}$: Saat mulai *batch* $L_{[i]}$
$L_{[i]}$: *Batch* yang dijadwalkan pada posisi ke-i
N : Jumlah *batch*
$Q_{[i]}$: Ukuran *batch* $L_{[i]}$

Parameter
q : Kuantitas permintaan komponen dalam unit
d : *Due date* bersama untuk seluruh aktivitas produksi (termasuk inspeksi kualitas dan *rework*)
d' : *Due date* untuk aktivitas *set-up* dan pengerjaan seluruh komponen dalam satuan waktu
t : Waktu proses per komponen dalam satuan waktu
s : Waktu *set-up batch* dalam satuan waktu
u : Proporsi ukuran sampel terhadap ukuran *batch*
$n_{[i]}$: Ukuran sampel untuk batch $L_{[i]}$ dalam unit
c_1 : Biaya simpan untuk *finished-part* per unit per satuan waktu dalam satuan biaya
c_2 : Biaya simpan untuk komponen *work-in-process* per unit per satuan waktu dalam satuan biaya
f_1 : Total biaya simpan per *batch* untuk *finished-part* dalam *in-process-batch* per *batch* dalam satuan biaya
f_2 : Total biaya simpan per *batch* untuk untuk komponen *work-in-process* dalam *in-process-batch* dalam satuan biaya
v : Ukuran penerimaan *batch* pada *acceptance sampling* dalam unit
y : Jumlah komponen tidak memenuhi spesifikasi yang ditemukan pada masing-masing *batch* dalam unit
k_1 : Biaya inspeksi per komponen per satuan waktu dalam satuan biaya
k_2 : Biaya pengerjaan ulang per komponen per satuan waktu dalam satuan biaya
k_3 : Biaya penalti per *batch* untuk batch yang ditolak oleh konsumen dalam satuan biaya
w : Waktu inspeksi per komponen dalam satuan waktu
P_a : Probabilitas penerimaan *batch* dalam *acceptance sampling*
p : Probabilitas kemunculan komponen yang tidak memenuhi spesifikasi
r : Waktu pengerjaan ulang per komponen dalam satuan waktu

2.6. Pembentukan Model CSA (1 Mesin)

Model penjadwalan *batch* pada mesin tunggal dengan *due date* bersama yang memperhatikan biaya simpan dan biaya kualitas berdasarkan variasi ukuran sampel yang bergantung pada ukuran *batch* disebut Model CSA. Indrapriyatna *et al.* (2007a) memformulasikan biaya kualitas dengan mempertimbangkan [6]:
a. Biaya pemeriksaan sampel
b. Biaya kegagalan internal
c. Biaya kegagalan eksternal

Total biaya pada Model CSA merupakan penjumlahan dari biaya simpan dan biaya kualitas, ditulis dengan notasi *TC(N,Q)*. Tujuan dari Model CSA adalah meminimumkan total biaya. Model CSA adalah sebagai berikut:

Model CSA

Minimumkan:
Total Biaya = Harapan total biaya simpan + Total biaya pemeriksaan sampel + Harapan total biaya kegagalan internal + Harapan total biaya kegagalan eksternal.

$$TC(N,Q) = c_1 \sum_{i=1}^{N-1} \left\{ \sum_{j=1}^{i} (tQ_{[j]} + s) \right\} Q_{[i+1]} + \frac{c_1 + c_2}{2} t \sum_{i=1}^{N} Q_{[i]}^2 + \frac{c_2 - c_1}{2} t \sum_{i=1}^{N} Q_{[i]} + uk_1 w \sum_{i=1}^{N} Q_{[i]} + uc_1 w \sum_{i=1}^{N} Q_{[i]}^2$$
$$+ (1 - P_a) \left((1-u) k_1 w \sum_{i=1}^{N} Q_{[i]} + (1-u) c_1 w \sum_{i=1}^{N} Q_{[i]}^2 + k_2 rp \sum_{i=1}^{N} Q_{[i]} + c_1 rp \sum_{i=1}^{N} Q_{[i]}^2 \right)$$
$$+ P_a (1 - P_a) \left(k_1 w \sum_{i=1}^{N} Q_{[i]} + c_1 w \sum_{i=1}^{N} Q_{[i]}^2 + k_2 rp \sum_{i=1}^{N} Q_{[i]} + c_1 rp \sum_{i=1}^{N} Q_{[i]}^2 + k_3 N \right) \quad (1)$$

$$(N-1)s + \sum_{i=1}^{N} tQ_{[i]} + uw \sum_{i=1}^{N} Q_{[i]} + (1-u)(1-P_a) w \sum_{i=1}^{N} Q_{[i]} + (1-P_a) rp \sum_{i=1}^{N} Q_{[i]} \le d \quad (2)$$

$$\sum_{i=1}^{N} Q_{[i]} = q \quad (3)$$

$$d' = d - \left(uw \sum_{i=1}^{N} Q_{[i]} + (1-u)(1-P_a) w \sum_{i=1}^{N} Q_{[i]} + (1-P_a) rp \sum_{i=1}^{N} Q_{[i]} \right) \quad (4)$$

= (Biaya simpan *finished part* + Biaya simpan *work-in-process*) + (Biaya pemeriksaan sampel + Biaya penyimpanan komponen selama pemeriksaan) + (Harapan biaya pemeriksaan komponen yang tidak termasuk sampel + Harapan biaya simpan komponen yang tidak termasuk sampel selama pemeriksaan + Harapan biaya pengerjaan ulang untuk komponen *non-comforming* + Harapan biaya simpan selama pengerjaan ulang) + Harapan total biaya kegagalan eksternal (persamaan 1).

Dengan batasan:

1. Seluruh aktivitas yang berkaitan dengan penyelesaian komponen (aktivitas pengendalian dan perbaikan kualitas, aktivitas setup serta pemrosesan seluruh komponen) tidak boleh melebihi due date (persamaan 2).
2. Selama horizon perencanaan, sistem hanya berproduksi sebanyak jumlah permintaan (persamaan 3).
3. Terdapat due date untuk aktivitas setup dan pemrosesan seluruh komponen, yang didefinisikan sebagai (persamaan 4).
4. Saat penyelesaian *batch* pertama harus sama dengan *due date* untuk aktivitas *setup* dan pemrosesan seluruh komponen

$$B_{[1]} + tQ_{[1]} = d' \quad (5)$$

5. Waktu mulai suatu *batch* harus sama dengan saat penyelesaian *batch* sebelumnya

$$B_{[i]} = B_{[i-1]} - (s + tQ_{[i]}) \quad i = 2, 3 \ldots N \quad (6)$$

6. Ukuran *batch* paling kecil adalah 1 (yaitu pada kondisi seluruh permintaan dijadikan satu *batch*) dan ukuran maksimum *batch* sama dengan jumlah permintaan komponen (yaitu pada kondisi jumlah permintaan dibagi menjadi *q batch* dengan ukuran masing-masing *batch* adalah 1)

$$1 \le N \le q \quad (7)$$

7. Ukuran batch harus lebih besar dari 0

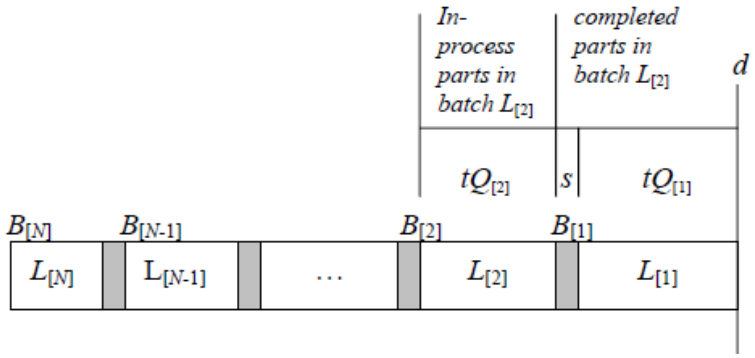
$$Q_{[i]} > 0, \; i = 1, 2 \ldots N \quad (8)$$

Ukuran *batch* optimal dari Model CSA, untuk setiap nilai *N*, diperoleh menggunakan Metode Lagrange, yaitu:

$$Q_{[i]} = \frac{q}{N} + \frac{c_1 s(N+1) - (2c_1 s)i}{2\left[2\left(\frac{c_2+c_1}{2}\right)t + 2uc_1 w - c_1 t + 2(1-P_a)\{(1-u)c_1 w + c_1 rp + P_a c_1 w + P_a c_1 rp\}\right]}$$

$$= \frac{q}{N} + \frac{c_1 s(N+1) - (2c_1 s)i}{2[c_2 t + 2uc_1 w + 2(1-P_a)\{(1-u)c_1 w + c_1 rp + P_a c_1 w + P_a c_1 rp\}]} \quad (9)$$

Posisi *batch* $L_{[i]}$ dalam sistem manufaktur yang terdiri atas satu mesin dengan pendekatan *backward* selama horizon perencanaan (yaitu dalam selang saat 0 sampai dengan *due date*) ditunjukkan dalam Gambar 2.



Gambar 2. Posisi *batch* dalam Sistem Manufaktur yang terdiri atas Satu Mesin (sumber: Indrapriyatna et al (2007a)) [6]

Algoritma usulan untuk menyelesaikan Model CSA, untuk selanjutnya akan disebut sebagai [Algoritma CSA], adalah sebagai berikut:

[Algoritma CSA]

Langkah 0 Tentukan nilai-nilai parameter *q, d, s, t, u, c_1, c_2, k_1, k_2, k_3, w, Pa,* *p* dan *r*. Tentukan N = 1. Lanjutkan ke Langkah 1.

Langkah 1 Tetapkan Q[1] = q dan B[1] sesuai dengan persamaan (4) dan (4). Periksa apakah Persamaan Persamaan (2) dan $B_{[1]} \geq 0$ dipenuhi. Jika dipenuhi maka lanjutkan ke Langkah 2. Jika tidak, maka tetapkan jadwal tidak layak dan lanjutkan ke Langkah 9.

Langkah 2 Hitung *TC(N,Q)* menggunakan Persamaan (1). Lanjutkan ke Langkah 3.

Langkah 3 Tentukan N = N + 1. Lanjutkan ke Langkah 4.

Langkah 4 Hitung $Q_{[i]}$ menggunakan Persamaan (9) dan $B_{[i]}$ sesuai Persamaan (4) – (6). Periksa apakah Persamaan (2) dan $B_{[N]} \geq 0$ dipenuhi. Jika dipenuhi maka lanjutkan ke Langkah 5. Jika tidak, maka lanjutkan ke Langkah 8.

Langkah 5 Hitung *TC(N,Q)* menggunakan Persamaan (1). Lanjutkan ke Langkah 6.

Langkah 6 Periksa apakah $N \leq q$. Jika ya maka lanjutkan ke Langkah 7. Jika tidak, lanjutkan ke Langkah 8.

Langkah 7 Periksa apakah *TC(N,Q)* ≤ *TC(N-1,Q)*. Jika ya maka kembali ke Langkah 3. Jika tidak maka lanjutkan ke Langkah 8.

Langkah 8 Tetapkan solusi yang diperoleh:
Jumlah *batch*: N=N-1.
Ukuran *batch* ke-i: $Q_{[i]}=Q_{[i]}$, i = 1, 2 ... N
Saat mulai *batch* pertama: $B_{[1]}=d' - tQ_{[1]}$
Saat mulai *batch* ke-i: $B_{[i]}=B_{[i-1]} - (s + tQ_{[i]})$, i = 2, 3 ... N
Total biaya=*TC(N,Q)*.
Lanjutkan ke Langkah 9.

Langkah 9 Selesai

Ukuran *batch* yang diperoleh melalui [Algoritma CSA] masih bersifat kontinu,

sementara ukuran *batch* seharusnya bersifat diskrit. Untuk mengatasi permasalahan tersebut, Indrapriyatna *et al* (2007b) menggunakan tiga metode pembulatan ukuran *batch*, yaitu [7]:

1. Metode Jumlah-Desimal-Atas (JDA)
   Jika penjumlahan nilai desimal ≥ 1 maka dibulatkan ke atas, tetapi jika penjumlahan nilai desimal < 1 maka dibulatkan ke bawah. Perhitungan nilai desimal dimulai dari urutan *batch* terbesar.
2. Metode Jumlah-Desimal-Bawah (JDB)
   Jika penjumlahan nilai desimal ≥ 1 maka dibulatkan ke atas, tetapi jika penjumlahan nilai desimal < 1 maka dibulatkan ke bawah. Perhitungan nilai desimal dimulai dari urutan *batch* terkecil.
3. Metode Pembulatan
   Jika nilai desimal ≥ 0,5 maka dilakukan pembulatan ke atas, sedangkan jika nilai desimal < 0,5 maka dilakukan pembulatan ke bawah.

Meskipun ukuran batch bernilai *integer*, ukuran sampel *u* yang proporsional terhadap ukuran *batch* masih bisa bernilai kontinu. Oleh karena itu, ukuran sampel ke-i juga perlu dijadikan bilangan integer (dinotasikan sebagai $n_{[i]}$). Hal ini dilakukan dengan cara menentukan nilai *integer* terkecil yang lebih besar dari ukuran sampel kontinu, atau secara matematis:

$$\boldsymbol{n_{[i]} = \lceil u Q_{[i]}' \rceil} \quad (10)$$

Perubahan ukuran *batch* dan sampel menjadi integer dapat menyebabkan terjadinya perubahan nilai Total Biaya yang diperoleh dari Model CSA Awal. Persamaan Total Biaya pada Model CSA Awal akan berubah menjadi:

Model CSA_Dis

$$TC(N,Q') = c_1 \sum_{i=1}^{N-1}\left\{\sum_{j=1}^{i} tQ_{[i]}' + s\right\} Q_{[i+1]}' + \frac{c_1+c_2}{2} t \sum_{i=1}^{N}\left[Q_{[i]}'\right]^2 + \frac{c_2-c_1}{2} t \sum_{i=1}^{N} Q_{[i]}' + k_1 w \sum_{i=1}^{N} n_{[i]} + c_1 w \sum_{i=1}^{N} Q_{[i]}' n_{[i]}$$
$$+ (1-P_a)\left(k_1 w \sum_{i=1}^{N}\left(Q_{[i]}' - n_{[i]}\right) + c_1 w \sum_{i=1}^{N} Q_{[i]}'\left(Q_{[i]}' - n_{[i]}\right) + k_2 rp \sum_{i=1}^{N}\left[Q_{[i]}'\right]^2\right)$$
$$+ P_a(1-P_a)\left(k_1 w \sum_{i=1}^{N} Q_{[i]}' + c_1 w \sum_{i=1}^{N}\left[Q_{[i]}'\right]^2 + k_2 rp \sum_{i=1}^{N} Q_{[i]}' + c_1 rp \sum_{i=1}^{N}\left[Q_{[i]}'\right]^2 + k_3 N\right) \quad (11)$$

Dengan batasan:

$$\boldsymbol{(N-1)s + \sum_{i=1}^{N} tQ_{[i]}' + w\sum_{i=1}^{N} n_{[i]} + (1-P_a)w\sum_{i=1}^{N}\left(Q_{[i]}' - n_{[i]}\right) + (1-P_a)rp\sum_{i=1}^{N} Q_{[i]}' \le d} \quad (12)$$

$$\boldsymbol{\sum_{i=1}^{N} Q_{[i]}' = q} \quad (13)$$

$$\boldsymbol{d' = d - \left(w\sum_{i=1}^{N} n_{[i]} + (1-P_a)w\sum_{i=1}^{N}\left(Q_{[i]}' - n_{[i]}\right) + (1-P_a)rp\sum_{i=1}^{N} Q_{[i]}'\right)} \quad (14)$$

$$\boldsymbol{B_{[1]} + tQ_{[1]}' = d'} \quad (15)$$

$$\boldsymbol{B_{[i]} = B_{[i-1]} - s - tQ_{[i]}' \quad i = 2,3 \ldots N} \quad (16)$$

$$\boldsymbol{1 \le N \le q} \quad (17)$$

$$\boldsymbol{Q_{[i]}' > 0,\ i = 1,2 \ldots N} \quad (18)$$

Penjelasan mengenai batasan yang digunakan pada Model CSA_Dis sama dengan batasan pada Model CSA. Karena ukuran *batch* merupakan bilangan *integer*, maka penyelesaian tidak bisa dilakukan menggunakan diferensiasi (turunan). Oleh sebab itu Indrapriyatna *et al.* (2007a) memformulasikan ulang Algoritma CSA menjadi Algoritma CSA_Dis untuk menghitung nilai Total biaya (*TC[N,Q']*) yang baru [6].

[Algoritma CSA_Dis]

Langkah 0 Gunakan [Algoritma CSA] untuk memperoleh jumlah dan ukuran *batch*. Beri indeks pada ukuran *batch* secara *backward*, dimulai dari *due date* hingga saat 0 ($Q_{[i]}$). Lanjutkan ke Langkah 1

Langkah 1 Ubah nilai $Q_{[i]}$ menjadi *integer*, (dinotasikan sebagai $Q_{[i]}'$) menggunakan Metode JDA, JDB dan PMB. Lanjutkan ke Langkah 2.

Langkah 2 Hitung nilai $n_{[i]}$ menggunakan Persamaan 10. Lanjutkan ke Langkah 3.

Langkah 3 Hitung *TC(N,Q')* menggunakan Persamaan 11 untuk metode JDA, JDB dan PMB. Periksa apakah Persamaan 12 sampai 18 terpenuhi. Jika ya, maka jadwal layak dan Total Biaya = *TC(N,Q')*. Jika tidak maka jadwal tidak layak. Lanjutkan ke Langkah 4.

Langkah 4 Bandingkan total biaya untuk setiap jadwal layak yang ditemukan. Lanjutkan ke Langkah 5. Jika tidak ada jadwal layak yang ditemukan maka tetapkan jadwal tidak layak dan lanjutkan ke Langkah 6.

Langkah 5 Tetapkan solusi yang diperoleh:
Jumlah *batch*: N = N
Ukuran *batch* ke-i: $Q_{[i]}'$ dihitung menggunakan metode terpilih (JDA, JDB atau PMB),
i = 1, 2 ... N
Saat mulai *batch* pertama: $B_{[1]} = d' - tQ_{[i]}'$
Saat mulai *batch* ke-i: $B_{[i]} = B_{[i-1]} - \left(s + tQ_{[i]}'\right)$,
i = 2, 3 ... N
Total Biaya: *TC(N,Q')* = Biaya Minimum
Lanjutkan ke Langkah 6.

Langkah 6 Selesai.

2.7. Algoritma Branch and Bound

Algoritma *Branch and Bound* merupakan algoritma yang dikembangkan untuk mencari hasil variabel keputusan *integer* dari permasalahan *linier programming* [13]. Algoritma ini didasarkan pada prinsip Metode Pencarian Melebar (*Breadth First Search*/ BFS). Basis penerapannya adalah persoalan-persoalan optimasi. Beberapa terminologi yang digunakan dalam implementasi algoritma ini antara lain:

1. *Feasible Solution*: Poin-poin dalam ruang pencarian yang memenuhi kendala batasan.
2. *Optimal Solution*: *Feasible solution* yang memenuhi fungsi tujuan.

Komponen utama dari algoritma ini adalah [14]:

1. *Branching* (Percabangan)
   Memecah persoalan menjadi satu atau lebih sub-persoalan.
2. *Bounding* (Batas)
   Menentukan nilai batas atas atau batas bawah yang memungkinkan.
3. *Pruning* (Pemotongan)
   Membandingkan nilai hasil percabangan dengan nilai batas atas atau batas bawah. Jika salah satu cabang yang dibandingkan tidak optimal, maka cabang tersebut akan diputus.
4. *Retracting* (Menarik kembali)
   Jika solusi telah diperoleh pada salah satu cabang terbawah, maka operasi mundur dilakukan kembali ke level teratas untuk membandingkan hasil solusi.

Hampir seluruh persoalan *integer programming* dapat diselesaikan menggunakan Algoritma *Branch and Bound*. Teknik ini mencari solusi optimal dengan mengenumerasi titik-titik dalam daerah *feasible* sebuah sub-persoalan [15]. Setiap simpul percabangan diasosiasikan dengan sebuah biaya yang menyatakan nilai batas (*bound*).

Pohon dinamis biasa digunakan untuk menggambarkan status persoalan pada saat pencarian solusi Algoritma *Branch and Bound* berlangsung. Status persoalan (*problem state*) dinyatakan dalam bentuk simpul-simpul percabangan di dalam pohon dinamis yang memenuhi kendala batasan (*constraints*). Status solusi (*solution state*) merupakan satu atau lebih

status yang menyatakan solusi persoalan. Status tujuan (*goal state*) adalah status solusi yang merupakan simpul daun.

Ruang status (*state space*) adalah seluruh simpul percabangan di dalam suatu pohon dinamis, sementara pohon dinamis tersebut dinamakan *state space tree*. Algoritma *Branch and Bound* menggunakan *state space tree* untuk mencari solusi persoalan.

3. METODOLOGI PENELITIAN

Metodologi penelitian merupakan hal yang sangat diperlukan dalam suatu penelitian. Metode penelitian menggambarkan langkah - langkah yang akan dilaksanakan dalam melakukan penelitian.

1. Studi Pendahuluan
   Tujuan studi pendahuluan ini adalah untuk memperoleh teori-teori yang menjadi landasan dalam melakukan pemecahan masalah dengan baik.
2. Identifikasi dan Perumusan Masalah
   Tujuan dari identifikasi masalah adalah untuk menjelaskan apa yang akan diselesaikan, kemudian merumuskan masalah, menjelaskan dan mengidentifikasikan masalah-masalah dalam batasan tertentu.
3. Pengumpulan Data
   Data yang dipakai dalam penelitian ini adalah data teoritis berdasarkan peneliti sebelumnya yaitu Indrapriyatna *et al.* (2007a).
4. Perancangan Algoritma
   Pengolahan yang dilakukan berupa penyusunan Algoritma *Branch and Bound* untuk mencari $Q_{[i]}$ yang bernilai *integer* dan pengujian logika algoritma. Selanjutnya ditentukan total biaya dan jadwal masing-masing *batch* berdasakan rumus yang telah ada.
5. Penutup
   Hasil perancangan dan hasil yang didapat kemudian disimpulkan dan diberikan saran-saran untuk perbaikan.

4. HASIL PENELITIAN

4.1. Pengumpulan Data

Data yang dipakai dalam penelitian ini adalah 7 set data yang diambil dari Indrapriyatna *et al.* (2007a). Di sini satuan untuk waktu dan biaya tidak dispesifikasikan, dengan alasan bahwa satuan apa pun (asalkan sesuai, misalkan menit untuk waktu, rupiah untuk biaya) dapat digunakan. Hal ini dilakukan untuk menunjukkan bahwa model dapat berlaku secara umum [6].

Tabel 1. Set Data yang Digunakan

Input	Set 1	Set 2	Set 3	Set 4	Set 5	Set 6	Set 7
q	10.000,00	100.000,00	100,00	50.000,00	50.000,00	550.000,00	550.000,00
d	8.000,00	110.000,00	120,00	50.000,00	60.000,00	600.000,00	1.100.000,00
s	30,00	30,00	2,00	2,00	10,00	40,00	50,00
t	0,10	0,10	0,10	0,20	0,50	0,90	0,70
w	1,20	1,20	1,20	5,00	2,00	1,40	1,00
c1	10,00	10,00	3,00	15,00	12,00	10,00	20,00
c2	4,00	4,00	2,00	10,00	10,00	6,00	15,00
k1	1,00	1,00	1,00	1,00	4,00	3,00	3,00
k2	15,00	20,00	15,00	20,00	20,00	20,00	20,00
k3	20,00	15,00	10,00	20,00	20,00	25,00	15,00
r	0,10	0,10	0,10	0,10	0,30	0,30	0,20
p	0,03	0,03	0,03	0,03	0,01	0,02	0,02

4.2. Perancangan Algoritma Branch and Bound

Algoritma CSA menjadi acuan dasar implementasi Algoritma *Branch and Bound* dalam model CSA.

Tahap awal perancangan dimulai dengan mengambil nilai Q[i] dan banyak *batch* (N) optimal hasil Algoritma CSA yang telah didapatkan dimana i = 1,2,3, ...,N. Dalam penerapan Algoritma *Branch and Bound* ini, diperlukan variabel-variabel tambahan sebagai berikut:

a = Banyaknya perulangan/ iterasi yang dilakukan dimana a = 1,2,3,…,N-1.

Qup[a] = Nilai Q[a] dibulatkan ke atas.

Qdown[a] = Nilai Q[a] dibulatkan ke bawah.

TC*up* = TC[N,Q] saat Q[a] dibulatkan ke atas.

TC*down* = TC[N,Q] saat Q[a] dibulatkan ke bawah.

q_awal = Jumlah permintaan (q) pada Algoritma CSA.

Sisa_*up* = q hasil dari

$$\text{q\_awal} - \sum_{i=1}^{a} Q[i]$$

saat Q[a] dibulatkan ke atas.

Sisa_*down* = q hasil dari q_awal – $\sum_{i=1}^{a} Q[i]$ saat Q[a] dibulatkan ke bawah.

Qbaru[N,a]= Variabel yang menampung nilai Q[i] *integer* (Q[i]').

TC_BB[N] = Variabel yang menampung Total Biaya hasil *Branch and Bound*.

Berikut ini adalah penerapan Algoritma *Branch and Bound* modifikasi untuk meng-*integer*-kan Q[i] (Q[i]'), disebut Algoritma CSA_BB_M.

Langkah 0. Ambil nilai Q[i] dengan jumlah *batch* N hasil Algoritma CSA.

Langkah 1. Tetapkan a =1 dimana a = 1, 2,3,…,N-1.

Langkah 2. Periksa apakah a < N. Jika ya, maka lanjut ke langkah 3. Jika tidak, lanjut ke langkah 12.

Langkah 3. Tetapkan nilai TC*up* = 0 dan TC*down* = 0.

Langkah 4. Bulatkan ke atas nilai dari Q[a] (*Qup*[a]).

Langkah 5. Bulatkan ke bawah nilai dari Q[a] (*Qdown*[a]).

Langkah 6. Untuk pembulatan ke atas: cari nilai q = q_awal – $\sum_{i=1}^{a} Q[i]$ kemudian tetapkan sisa_*up* = q. Cari nilai Q[i + a] dengan persamaan 9 dimana i = 1,2,3,…,N-a. Hitung TC*up* menggunakan Persamaan 1.

Langkah 7. Untuk pembulatan ke bawah: cari nilai q = q_awal – $\sum_{i=1}^{a} Q[i]$ kemudian tetapkan sisa_*down* = q. Cari nilai Q[i + a] dengan persamaan 9 dimana i = 1, 2,3,…, N-a. Hitung TC*down* menggunakan Persamaan 1.

Langkah 8. Periksa apakah TC*up* < TC*down*. Jika ya, maka lanjut ke langkah 9. Jika tidak, lanjut ke langkah 10.

Langkah 9. Tetapkan:
Qbaru [N, a] = Q*up*[a].
Q[a] = Qbaru [N, a].
TC_BB [N] = TC*up*.
Qbaru [N, N] = sisa_*up*.
Lanjut ke langkah 11.

Langkah 10. Tetapkan:
Qbaru [N, a] = Q*down*[a].
Q[a] = Qbaru [N, a].
TC_BB [N] = TC*down*.
Qbaru [N, N] = sisa_*down*.
Lanjut ke langkah 11.

Langkah 11. Tentukan a = a + 1. Kembali ke langkah 2.

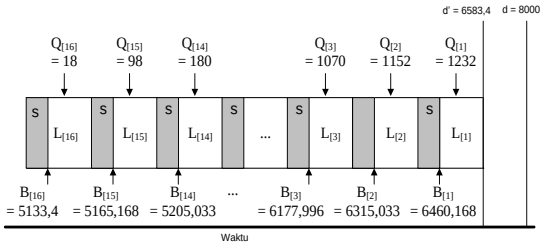
Langkah 12. Selesai.

Flowchart dari algoritma CSA_BB_M dapat dilihat pada Gambar 3.

Setelah mendapatkan ukuran *batch* yang *integer* (Q'[i]), maka dicari Total Biaya (TC[N,Q']) berdasarkan Model CSA_Dis untuk Q'[i] dengan menggunakan Persamaan 11. Sebelumnya dicari ukuran sampel yang *integer*. Ukuran sampel *integer* untuk *batch* hasil Algoritma CSA_BB_M ditentukan dengan cara: nilai *integer* paling kecil yang lebih besar daripada nilai dari ukuran sampel kontinu, yaitu: $n[i] = \lceil uQ[i]' \rceil$, disebut Metode CSA_Dis_BB_M. Setelah didapatkan ukuran sampel yang *integer*, maka dapat dicari Total Biaya TC([N.Q']) dengan Persamaan 11. Berikut ini adalah total biaya Metode CSA_Dis yang menggunakan ukuran sampel *integer* hasil Metode CSA_Dis_BB_M, JDA, JDB dan Pembulatan (Persamaan 11).

4.3. Analisis Hasil Perancangan Algoritma CSA_BB_M

Perancangan Algoritma CSA_BB_M bertujuan mendapatkan ukuran Q[i] yang *integer* (Q[i]'), jadwal untuk masing-masing *batch* dan *due date* untuk aktivitas *setup* dan pemrosesan seluruh *part* yang baru (d') serta Total Biaya (TC[N,Q']) yang minimum. Untuk jadwal dari masing-masing *batch* dan d' dapat dilihat pada Gambar 9.

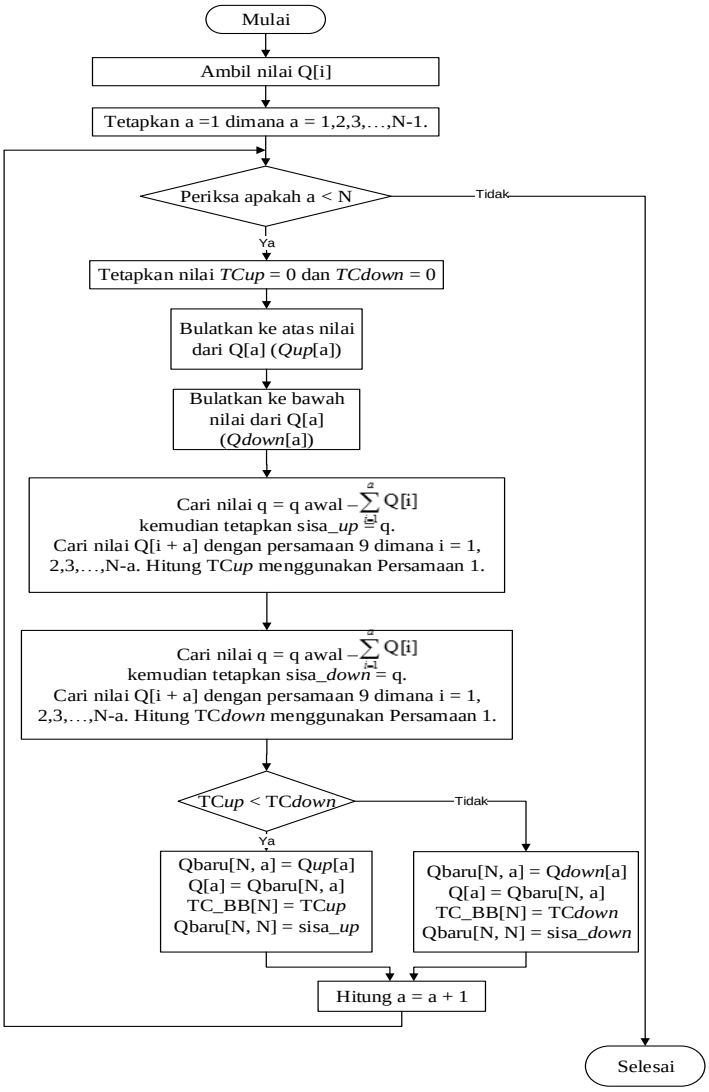


Gambar 9. Jadwal untuk Masing-Masing *Batch*

Berdasarkan Gambar 9, terlihat *due date* untuk aktivitas *setup* dan pemrosesan seluruh *part* yang baru (d') lebih kecil dari *due date* awal. Hal ini menunjukkan jadwal dan *due* date baru (d') yang dihasilkan memenuhi batasan-batasan yang telah ditetapkan.

Sedangkan hasil yang didapatkan untuk total biaya adalah Total Biaya dengan Algoritma CSA_BB_M (TC[N,Q']) lebih besar dari TC[N,Q] Algoritma CSA. Hal ini dipengaruhi oleh ukuran ukuran *batch* yang sudah di-*integer*-kan. TC[N,Q'] dengan Algoritma CSA_Dis_BB tidak semuanya lebih kecil dari metode peng-*integer* (JDA, JDB, dan Pembulatan) yang dipakai oleh Indrapriyatna *et al*. (2007b) [7]. Hal ini disebabkan ukuran *batch* masing-masing

Berikut ini adalah *flowchart* dari algoritma CSA_BB_M :



Gambar 3. *Flowchart* Algoritma CSA_BB_M

metode ada yang tidak sama, sedangkan pada Model CSA_Dis terdapat operasi pengurangan, perkalian dan perpangkatan untuk masing-masing ukuran *batch*.

Tabel 2. Rekapitulasi Jumlah Permintaan Sebelum dan Sesudah Proses Peng-*integer*-an untuk u = 10%

Data Set	Permintaan Awal (q)	Permintaan Saat CSA_BB_M	Permintaan Saat JDA	Permintaan Saat JDB	Permintaa Saat Pembulata
1	10.000	10.000	10.000	10.000	10.00
2	100.000	100.000	100.000	100.000	100.00
3	100	100	100	100	10
4	50.000	50.000	50.000	50.000	50.00
5	50.000	50.000	50.000	50.000	49.99
6	550.000	550.000	550.000	550.000	550.00
7	550.000	550.000	550.000	550.000	550.00

Tabel 3. Rekapitulasi Jumlah Permintaan Sebelum dan Sesudah Proses Peng-*integer*-an untuk u = 20%

Data Set	Permintaan Awal (q)	Permintaan Saat CSA_BB_M	Permintaan Saat JDA	Permintaan Saat JDB	Permintaa Saat Pembulata
1	10.000	10.000	10.000	10.000	10.00
2	100.000	100.000	100.000	100.000	100.00
3	100	100	100	100	9
4	50.000				
5	50.000	50.000	50.000	50.000	50.00
6	550.000				
7	550.000	550.000	550.000	550.000	550.00

Tabel 2 dan Tabel 3 memperlihatkan rekap jumlah permintaan setelah dilakukannya proses peng-*integer*-an.

Rekapitulasi Total Biaya 7 set data dapat dilihat pada Tabel 4 dan Tabel 5. Berdasarkan Tabel 4 dan 5 (pada kolom Pembulatan), terdapat kotak yang diberi warna abu-abu dan diberi simbol NA. Maksudnya adalah perhitungan Total Biaya (TC[N,Q']) untuk set data 5 dengan proporsi sampel (u) = 10% dan set data 3 dengan proporsi sampel (u) = 20% pada Metode Pembulatan tidak dilakukan karena jumlah permintaan hasil peng-*integer*-annya tidak memenuhi batasan pada model CSA dimana jumlah ukuran *batch* tidak sama dengan jumlah permintaan awal seperti yang terlihat pada Tabel 2 dan 3.

5. KESIMPULAN DAN SARAN

Berdasarkan perancangan Algoritma *Branch and Bound* modifikasi yang telah dibuat, maka dapat diambil beberapa kesimpulan yaitu:

1. Algoritma CSA_BB_M mendapatkan ukuran *batch* yang diskrit (*integer*) tanpa mengubah banyak *batch* dan jumlah permintaan yang akan diproduksi. Penerapan Algoritma CSA_BB_M dilakukan pada 7 set data yang terdapat pada Indrapriyatna *et al*. (2007a) [6]. Untuk 7 set data dan metode peng-*integer* yang digunakan,

Tabel 4. Hasil Perhitungan Model CSA , CSA_BB_M dan Model CSA_Dis dengan u = 10%

Set Data	Model CSA	CSA_BB_M	Model CSA_Dis menggunakan metode			
			Jumlah-desimal-atas	Jumlah-desimal-bawah	Pembulatan	CSA_Dis_BB_M
Set 1	Rp 79.946.295,95	Rp 79.946.302,67	Rp 80.009.882,97	Rp 80.011.304,63	Rp 80.000.334,60	**Rp 79.995.360,68**
Set 2	Rp 6.036.997.149,52	Rp 6.037.012.190,60	**Rp 6.037.539.487,26**	Rp 6.037.543.861,93	Rp 6.037.542.765,08	Rp 6.037.542.765,08
Set 3	Rp 4.683,16	Rp 4.683,80	**Rp 4.844,31**	Rp 4.851,55	Rp 4.846,88	Rp 4.846,88
Set 4	Rp 4.024.001.544,39	Rp 4.024.004.959,71	Rp 4.025.698.186,67	Rp 4.025.700.717,75	Rp 4.025.662.344,82	**Rp 4.025.660.400,21**
Set 5	Rp 7.945.875.401,08	Rp 7.945.878.610,37	Rp 7.946.387.171,50	Rp 7.946.389.160,92	NA	**Rp 7.946.370.307,07**
Set 6	Rp 1.385.189.731.996,40	Rp 1.385.189.782.217,74	**Rp 1.385.193.097.488,45**	Rp 1.385.193.107.318,21	Rp 1.385.193.419.190,73	Rp 1.385.193.408.802,91
Set 7	Rp 2.171.126.128.412,81	Rp 2.171.126.207.002,90	Rp 2.171.130.933.878,69	Rp 2.171.130.947.604,92	Rp 2.171.130.805.941,50	**Rp 2.171.130.804.806,65**

Tabel 5. Hasil Perhitungan Model CSA , CSA_BB_M dan Model CSA_Dis dengan u = 20%

Set Data	Model CSA	CSA_BB_M	Model CSA_Dis menggunakan metode			
			Jumlah-desimal-atas	Jumlah-desimal-bawah	Pembulatan	CSA_Dis_BB_M
Set 1	Rp 88.701.045,51	Rp 88.701.051,16	Rp 88.760.219,44	**Rp 88.751.015,46**	Rp 88.761.962,08	Rp 88.761.962,08
Set 2	Rp 6.299.363.829,06	Rp 6.299.365.042,70	Rp 6.299.877.394,03	Rp 6.299.849.137,30	**Rp 6.299.823.235,01**	**Rp 6.299.823.235,01**
Set 3	Rp 5.164,72	Rp 5.167,53	Rp 5.314,47	Rp 5.326,04	NA	**Rp 5.301,45**
Set 4	Jadwal tidak layak	Jadwal Tidak Layak	Jadwal tidak layak	Jadwal tidak layak	Jadwal tidak layak	Jadwal Tidak Layak
Set 5	Rp 8.008.561.073,33	Rp 8.008.564.331,18	Rp 8.009.020.976,71	Rp 8.009.024.453,28	**Rp 8.009.018.134,11**	Rp 8.009.018.847,50
Set 6		Jadwal Tidak Layak				Jadwal Tidak Layak
Set 7	Rp 2.176.008.487.542,06	Rp 2.176.008.487.698,73	Rp 2.176.012.679.086,23	Rp 2.176.012.705.086,14	Rp 2.176.012.555.358,62	**Rp 2.176.012.540.375,54**

jumlah permintaan setelah dilakukan proses peng-*integer*-an tetap atau sama dengan jumlah permintaan awal kecuali untuk Metode Pembulatan. Pada Metode Pembulatan jumlah permintaan ada yang berbeda yaitu untuk set data 5 pada proporsi sampel (u) = 10% dan set data 3 pada proporsi sampel (u) = 20% dimana jumlah permintaan kurang 1 unit dari jumlah permintaan awal. Hal ini menunjukkan, pada set data tersebut, Metode Pembulatan gagal memenuhi salah satu batasan pada model CSA yaitu jumlah dari ukuran *batch* yang telah diskrit harus sama dengan jumlah permintaan awal.

2. Metode CSA_Dis_BB_M (untuk meng-*integer*-kan ukuran sampel hasil Algoritma CSA_BB_M) tidak selalu menghasilkan total biaya yang minimum jika dibandingkan dengan metode peng-*integer* yang lain (JDA, JDB dan Pembulatan). Hal ini disebabkan oleh ukuran *batch* masing-masing metode ada yang tidak sama, sedangkan pada Model CSA_Dis terdapat operasi pengurangan, perkalian dan perpangkatan untuk masing-masing ukuran *batch.* Hal ini berpengaruh terhadap hasil akhir, yaitu total biaya yang didapatkan. Hasil perhitungan dengan proporsi sampel 10 % menunjukkan bahwa Metode Jumlah-Desimal-Atas menghasilkan solusi terbaik untuk set data 2, 3, dan 6. Metode Pembulatan menghasilkan solusi terbaik untuk set data 5, tetapi karena jumlah permintaannya kurang dari jumlah permintaan awal (q) maka total biaya terkecil untuk set data 5 dicari dari 3 metode lainnya (JDA, JDB, dan CSA_Dis_BB_M). Metode CSA_Dis_BB_M menghasilkan solusi terbaik untuk set data 1, 4, 5, dan 7.

Hasil perhitungan dengan proporsi sampel 20 % menunjukkan bahwa Metode Jumlah-Desimal-Bawah menghasilkan solusi terbaik untuk set data 1. Metode Pembulatan menghasilkan solusi terbaik untuk set data 2, 3, dan 5, tetapi karena jumlah permintaan pada data set 3 kurang dari jumlah permintaan awal (q) maka total biaya terkecil untuk set data 3 dicari dari 3 metode lainnya (JDA, JDB, dan CSA_Dis_BB_M). Metode CSA_Dis_BB_M menghasilkan solusi terbaik untuk set data 2, 3 dan 7.

Setelah melakukan perancangan Algoritma CSA_BB_M dan Metode CSA_Dis_BB_M dan agar penelitian ini lebih baik kedepannya, disarankan agar:

1. Mencoba metode peng-*integer* ukuran *batch* yang lain, karena algoritma dan metode peng-*integer* ukuran *batch* dan sampel yang telah dicobakan (Algoritma CSA_BB_M dan Metode CSA_Dis_BB_M) belum menghasilkan solusi optimal.
2. Penelitian selanjutnya dapat menerapkan untuk model-model lainnya dimana penelitian ini hanya mengacu pada model 1 mesin (Model CSA), sedangkan Indrapriyatna *et al.* (2007b) mengembangkan model penjadwalan untuk 2 mesin, 3 mesin, dan m mesin [7].
3. Menggunakan data *real* atau data berdasarkan pengamatan di lapangan dengan kondisi yang sesuai dengan model yang ada, agar dapat diuji apakah Algoritma CSA_BB_M dan Metode CSA_Dis_BB_M ini berlaku untuk data apapun.

DAFTAR PUSTAKA

[1] D. D. Bedworth dan J. E. Bailey. (1987). *Integrated Production Control Systems: Management, analysis, design Second edition*. Singapore: John Wiley & Sons Inc.

[2] J. E. Biegel. (1971). *Production Control: A quantitative approach*, New Jersey, USA: Prentice-Hall, Inc.

[3] A. H. Halim dan H. Ohta. (1993). "Batch Sheduling Problem Through the Flow Shop with Both Receiving and Delivery Just In Time", *International Journal of Production Research*, Vol. 31, pp. 1943-1955.

[4] A. H. Halim dan H. Ohta. (1994). "Batch Scheduling Problem to Minimize Inventory Cost in the Shop with Both Receiving and Delivery Just In Time, *International Journal of Production Eco*, Vol. 33, pp. 185-195.

[5] A. H. Halim, J. Silalahi dan H. Ohta. (2001). "A Batch Scheduling Model Considering Quality Costs for the Shop with Receiving and Delivery Just In Time", *Proceeding of the 2001 International Conference on Production Research*, Prague, Czech Republic. 29 July – 3 August.

[6] A. S. Indrapriyatna, Suprayogi, B. P. Iskandar dan A. H. Halim. (2007). "A Batch Scheduling Model for A Single Machine Processing Discrete Parts to Minimize Total Inventory and Quality Cost", *Proceeding of the 1st Asia Pacific Conference on Manufacturing Systems*, Bali, Indonesia, 5 – 6 Septermber.

[7] A. S Indrapriyatna Suprayogi, B. P. Iskandar dan A. H. Halim. (2007). "Model Penjadwalan Batch pada Flowshop untuk Minimasi Biaya Simpan dan Kualitas', *Jurnal Teknik dan Manajemen Industri ITB*, Vol. 27, pp. 142-163.

[8] K. R. Baker. (1974). *Introduction to Sequencing and Scheduling*, New York, USA: John Wiley & Sons Inc.

[9] T'kindt, Vincent dan Jean-Charles Billaut. (2006). *Multicriteria Scheduling, Theory, Models, and Algorithms, Second Edition*. France: Springer.

[10] E. Herjanto. (2008). *Manajemen Operasi Edisi Ketiga*, Jakarta, Indonesia: Grasindo.

[11] H. Prasetya dan F. Lukiastuti. (2009). *Manajemen Operasi*, Yogyakarta, Indonesia: Media Pressindo.

[12] M. S. Bazaraa, H. D. Sherali, dan C. M. Shetty. (2007). *Nonlinear Programming,* 2nd ed. Canada: John Wiley & Sons Inc.

[13] A. H. Land, dan A. G. Doig. (1960). *An Automatic Method of Solving Discrete Programming Problems*. Econometrica 28 (3). pp. 497–520.

[14] M. J. Brusco dan S. Stahl. (2005). *Statistics and Computing: Branch and Bound Applications In Combinatorial Data Analysis*, New York, USA: Springer Science + Business Media, Inc.

[15] T. T. Dimyati dan A. Dimyati. (2006). *Operations Research: Model-Model Pengambilan Keputusan*, Bandung, Indonesia: Sinar Baru Algensindo.