UNIVERSITAS INDONESIA

MODIFIKASI *BEE COLONY ALGORITHM* DENGAN *TABU LIST* PADA PENJADWALAN *JOB SHOP* DENGAN KRITERIA BIAYA KETERLAMBATAN

TESIS

ANDRE SUGIOKO
1006787413

.

PROGRAM PASCA SARJANA TEKNIK INDUSTRI
FAKULTAS TEKNIK INDUSTRI
UNIVERSITAS INDONESIA
JAKARTA
2012

UNIVERSITAS INDONESIA

MODIFIKASI *BEE COLONY ALGORITHM* DENGAN *TABU LIST* PADA PENJADWALAN *JOB SHOP* DENGAN KRITERIA BIAYA KETERLAMBATAN

TESIS

Diajukan sebagai salah satu syarat untuk mendapatkan gelar Magister

ANDRE SUGIOKO
1006787413

.

FAKULTAS TEKNIK
PROGRAM PASCA SARJANA JURUSAN TEKNIK INDUSTRI
JAKARTA
2012

PERNYATAAN KEASLIAN TESIS

Saya menyatakan dengan sesungguhnya bahwa Tesis dengan judul:

MODIFIKASI *BEE COLONY ALGORITHM* DENGAN *TABU LIST* PADA PENJADWALAN *JOB SHOP* DENGAN KRITERIA BIAYA KETERLAMBATAN

yang dibuat untuk melengkapi sebagian persyaratan menjadi Sarjana Teknik pada Program Studi Teknik Industri Departemen Teknik Industri Fakultas Teknik Universitas Indonesia, sejauh yang saya ketahui bukan merupakan tiruan atau duplikasi dari tesis yang sudah dipublikasikan atau pernah dipakai untuk mendapatkan gelar kesarjanaan di lingkungan Universitas Indonesia maupun di Perguruan Tinggi atau instansi manapun, kecuali bagian yang sumber informasinya dicantumkan sebagaimana mestinya.

Nama : Andre Sugioko
NPM : 1006 7874 13
Tanda Tangan :
Tanggal :

HALAMAN PENGESAHAN

Tesis ini Diajukan oleh:
Nama : Andre Sugioko
NPM : 1006787413
Program Studi : Teknik Industri
Judul Tesis : MODIFIKASI BEE COLONY ALGORITHM DENGAN TABU LIST PADA PENJADWALAN JOB SHOP DENGAN KRITERIA BIAYA KETERLAMBATAN

Telah berhasil dipertahankan di hadapan Dewan Penguji dan dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Magister pada Program Studi Teknik Industri, Fakultas Teknik, Universitas Indonesia

DEWAN PENGUJI

Pembimbing 1 : Ir. Isti Surjandari Prajitno MT., MA., Ph.D ( )

Pembimbing 2 : Ir. Amar Rachman, MEIM ( )

Penguji 1 : Prof. Dr. Ir. T. Yuri M. Zagloel, M.Eng.Sc ( )

Penguji 2 : Dr. Ir. Akhmad Hidayatno, MBT ( )

Penguji 3 : Ir. Sri Bintang Pamungkas, MSISE, Ph.D ( )

Penguji 4 : Ir. Djoko S. Gabriel, MT ( )

Ditetapkan di : Depok

Tanggal : Januari 2012

KATA PENGHANTAR/UCAPAN TERIMA KASIH

Puji dan Syukur saya panjatkan kepada Tuhan karena atas berkat dan rahmat-Nya sehingga penulis dapat menyelesaikan Tesis ini. Penulisan Tesis ini disusun sebagai salah satu syarat untuk mencapai gelar Magister Teknik Jurusan Teknik Industri, Fakultas Teknik, Universitas Indonesia.

Pada kesempatan ini penulis ingin mengucapkan terima kasih atas bantuan dan bimbingan yang telah diberikan kepada penulis dalam menyelesaikan Tesis ini kepada :

1. Keluarga, atas curahan kasih sayang , dukungan, dan doa yang diberikan.
2. Ir. Isti Surjandari, MT, MA, Ph.D, selaku dosen pembimbing pertama yang telah banyak memberi bantuan, masukan dan bimbingan yang berharga bagi penulis.
3. Ir. Amar Rachman, MEIM, selaku dosen pembimbing kedua yang telah banyak memberi bantuan, masukan dan bimbingan yang berharga bagi penulis.
4. Teman-teman penulis, khususnya rekan-rekan 2010 yang telah memberikan dukungan, semangat, serta kebersamaan selama dua tahun ini.
5. Pihak-pihak lain yang juga telah membantu penyelesaian tesis ini namun tidak dapat disebutkan satu per satu..

Penulis pun menyadari bahwa dalam Tesis ini masih terdapat banyak kekurangan, oleh sebab itu penulis memohon saran dan kritik bagi penulis untuk nantinya dapat lebih berkembang lagi. Akhir kata, inilah karya penulis yang dapat penulis persembahkan, semoga dapat bermanfaat bagi rekan-rekan sekalian. Terima kasih.

Jakarta, 2012
Penulis

LEMBAR PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Indonesia, saya yang bertanda tangan dibawah ini:

Nama	: Andre Sugioko
NPM	: 1006787413
Departemen	: Teknik Industri
Fakultas	: Teknik
Jenis Karya	: Tesis

Demi pengembangan ilmu pengetahuan , ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty-Free Right*)** atas karya ilmiah saya yang berjudul :

MODIFIKASI *BEE COLONY ALGORITHM* DENGAN *TABU LIST* PADA PENJADWALAN *JOB SHOP* DENGAN KRITERIA BIAYA KETERLAMBATAN

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Nonekslusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan memublikasikan tugas akhir saya tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Jakarta
Pada tanggal : 2011
Yang menyatakan

(Andre Sugioko)

ABSTRAK

Nama : Andre Sugioko
Program Studi : Teknik Industri
Judul : MODIFIKASI BEE COLONY ALGORITHM DENGAN TABU LIST PADA PENJADWALAN JOB SHOP DENGAN KRITERIA BIAYA KETERLAMBATAN

Penjadwalan *job shop* dengan kriteria biaya keterlambatan merupakan permasalahan yang jarang digunakan dalam penelitian *job shop*. Umumnya penjadwalan *job shop* diselesaikan dengan menggunakan metode metaheuristik, salah satu metode metaheuristik yang populer dibicarakan adalah algoritma *Bee Colony*. Algoritma *Bee Colony* merupakan algoritma yang tidak memiliki metode untuk lepas dari *local optimum*, seperti yang dinyatakan pada penelitian Chong (Chong, et al. 2005), maka penelitian ini akan melakukan modifikasi terhadap algoritma *Bee Colony* dengan menggunakan *tabu list*, untuk meningkatkan perfroma pencarian solusi dan waktu komputasi untuk permasalahan penjadwalan *job shop* dengan kriteria biaya keterlambatan.

Hasil penelitian menunjukan bahwa algoritma *Bee colony-Tabu* memberikan perfroma yang serupa untuk kriteria biaya keterlambatan dan waktu komputasi terhadap algoritma *Tabu Search* dan lebih baik daripada algoritma *Bee Colony* dan *Differentialial Evolution* untuk kriteria biaya keterlambatan. Sedangkan untuk waktu komputasi algoritma *Bee* colony dengan *Tabu List* lebih unggul daripada algoritma *Tabu Search* dan *Bee Colony*, namun waktu komputasi algoritma *Differentialial Evolution* lebih unggul daripada algoritma *Bee colony-Tabu*, *Tabu Search* dan *Bee Colony*.

Kata Kunci:
Penjadwalan, *Job Shop*, Biaya Keterlambatan, Algoritma *Bee Colony*, *Local* Optimum, Algoritma *Tabu Search*, *Tabu List*.

ABSTRACT

Name : Andre Sugioko
Program Studi : Teknik Industri
Judul : MODIFICATION OF BEE COLONY ALGORITHM WITH TABU LIST FOR JOB SHOP SCHEDULING WITH TARDINESS COST

Job shop scheduling with tardiness cost is a problem that rarely exist in paper research. Generally, job shop scheduling solved using metaheuristik method, one of metaheuristik methods popular discussed in many paper are Bee Colony algorithm. Bee Colony Algorithm is an algorithm that does not have a method to escape from local optimum, as stated in the Chong's research (Chong, et al. 2005), because of that this research will make modifications to the Bee Colony algorithm using the taboo list, to improve searching solution and computing time for job shop scheduling problems with late fees criteria.

The results showed that the Bee colony-Tabu algorithm gives perfromance similar to the Tabu Search algorithm and better than Bee Colony algorithm for late fees criteria and computation time, and Differentialial Evolution for the criteria for late fees. As for computational time Bee colony with Tabu List algorithm is superior to Tabu Search algorithm and the Bee Colony, but the computing time algorithm Differentialial Evolution algorithm is superior to Bee Colony-Tabu, Tabu Search and Bee Colony.

Keyword:
Scheduling, Job Shop, Late Fees, Bee Colony Algorithm, Local Optimum, Tabu Search Algorithm, Tabu List.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR RUMUS

DAFTAR LAMPIRAN

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Penjadwalan merupakan bagian yang penting dari suatu sistem produksi suatu perusahaan. Tujuan utama dari penjadwalan adalah untuk menentukan jadwal pekerjaan yang mampu untuk meminimalkan (atau memaksimumkan) satu ukuran (atau beberapa ukuran) perfomansi. Penjadwalan *job shop* merupakan merupakan persoalan yang sering dipakai dalam penelitian-penelitian terutama dalam aplikasi algoritma-algoritma metaheuristik. Beberapa algoritma metaheuristik yang digunakan untuk permasalahan *job shop* antara lain adalah *Scatter Search Method* untuk permasalahan *Fuzzy Job Shop* (Engin et al, 2011); *Simulated Annealing* (Van Laarhoven et al, 1992); *Genetic Algorithms* (Yamada, T and Nakano, R, 1997); *Tabu Search* (Schmidt, K, 2001); *Differential Evolution Algorithm* (Zhang, R, 2011) dan algoritma yang sedang banyak dikembangkan saat ini untuk penjadwalan *job shop* yaitu *Bee Colony* (Chong et al, 2005).

Algoritma *Bee Colony* merupakan salah satu bagian dari algoritma *swarm particle* yang menggunakan kebiasaan *forgaging* dan *waggle dance* dari lebah saat mencari makanan, yang diperkenalkan pertama oleh Karaboga. Dimana keunggulan algoritma *Bee Colony* terdapat pada kemampuannya untuk menyelesaikan permasalahan bersifat kontinu dan memiliki struktur yang sederhana, sedangkan untuk masalah diskrit performa algoritma *bee colony* kurang baik terutama dari algoritma *Tabu Search* untuk permasalahan penjadwalan *job shop*. Pengembangan-pengembangan yang dilakukan terhadap algoritma *Bee Colony* antara lain adalah dengan menggunakan pencarian solusi dengan *profitability rating* (Chong et al, 2005), memasukan algoritma *Bee colony* kedalam *neighbourhood search* (Chong et al, 2006); dan menggunakan *Big Valley Landscape* (Chong, 2007). Namun belum ada yang melakukan modifikasi algoritma *Bee Colony* dengan *tabu list*, pemikiran ini didasari oleh pernyataan dalam penelitian Chong tahun 2005 yang menyatakan bahwa performa algoritma *Bee Colony* dibawah *Tabu Search* dikarenakan pencarian solusi tidak

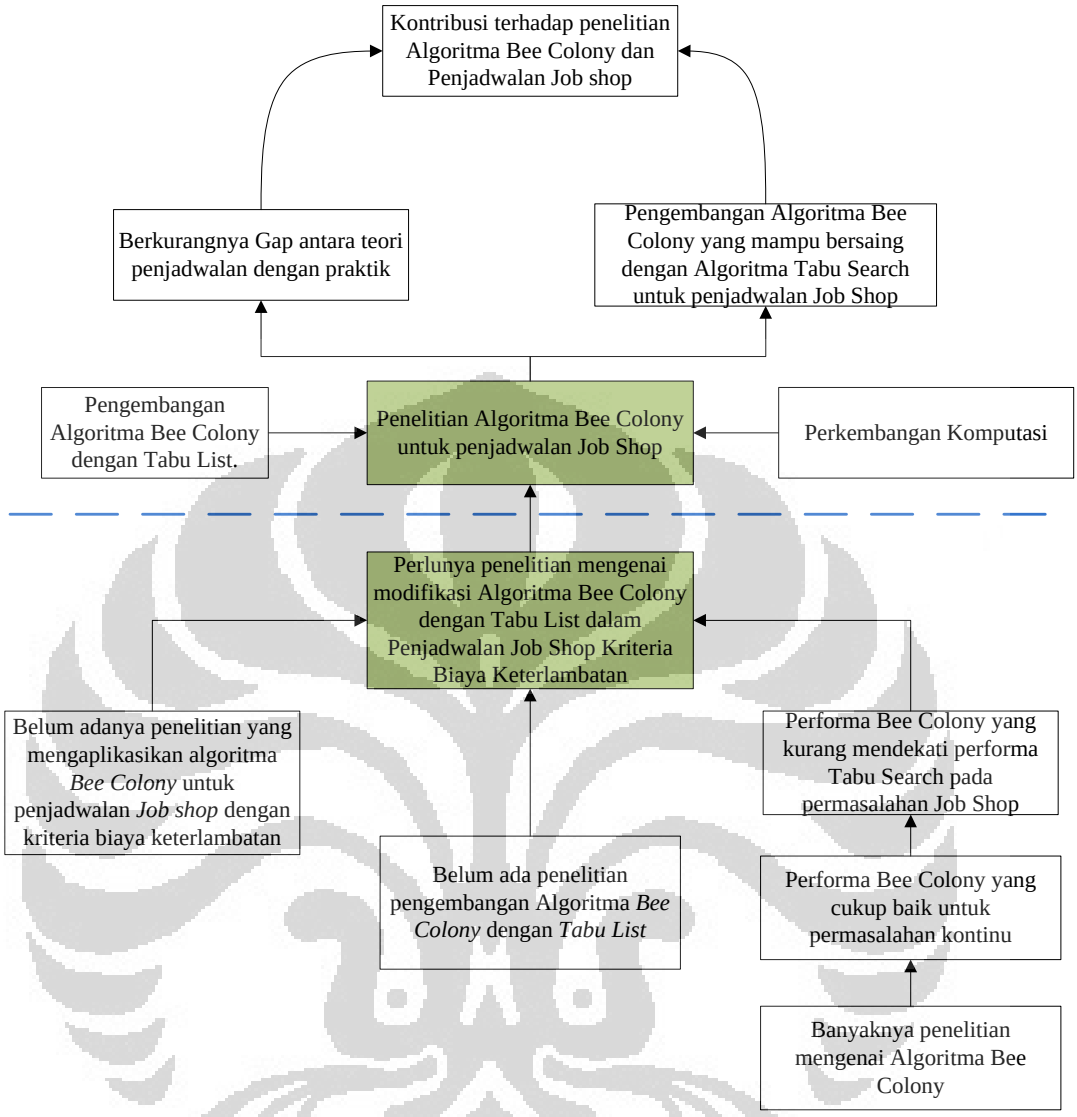
menggunakan metode *swap* dan algoritma *Bee Colony* tidak memiliki metode untuk lepas dari *local optimum* seperti *tabu list* pada *Tabu Search*.

Penjadwalan *job shop* selain bertujuan untuk meminimasi *makespan* terdapat variasi lainnya, antara lain *Availability Constraints* (Engin et al, 2011), *Dependent Setup Times* (Schmidt. K, 2001); dan kendala *due date* (Panwalkar et al. 1981). Namun diantara variasi penjadwalan *job shop* belum ada yang memperhatikan kendala biaya keterlambatan seperti pada penelitian Lina tahun 2008, yang menggunakan algoritma *Differential Evolution* untuk menyelesaikan permasalahan tersebut. Namun algoritma *Differential Evolution* sama seperti algoritma *Bee Colony* yang sama-sama tidak memiliki metode untuk lepas dari *local optimum*. Serta belum adanya penelitian yang mengaplikasikan algoritma *Bee Colony* untuk permasalahan *job shop* yang memperhatikan kendala biaya keterlambatan.

Berdasarkan kedua permasalahan diatas, maka penelitian ini bermaksud untuk memodifikasi algoritma *Bee Colony* dengan *tabu list* untuk permasalahan penjadwalan *job shop* dengan memperhatikan kendala biaya keterlambatan. Penelitian ini akan menggunakan data dari penelitian yang telah dilakukan oleh Astuti (2008) untuk produk *Revo Frame*, dan membandingkan performa modifikasi algoritma *Bee Colony* dengan algoritma yang digunakan dalam penelitian Astuti yaitu *Differential Evolution*.

Dengan demikian, sebuah penelitian mengenai modifikasi algoritma *bee colony* dengan *tabu list* untuk penjadwalan *job shop* dengan biaya keterlambatan, diharapkan akan memberikan kontribusi untuk penelitian kedepannya.

1.2 Diagram Keterkaitan Masalah



Gambar 1.1 Diagram Keterkaitan Masalah

1.3 Perumusan Masalah

Berdasarkan latar belakang masih sedikitnya penelitian yang mengaplikasikan algoritma *bee colony* pada penjadwalan *job shop* dengan kendala biaya keterlambatan. Serta belum adanya penelitian menggunakan *tabu list* dalam algoritma *bee colony*, dimana dengan *tabu list* diharapkan performa algoritma *bee colony* mengalami peningkatan dari segi hasil yang lebih mendekati optimal dan waktu komputasi yang lebih cepat dibandingkan sebelum dilakukan modifikasi.

1.4 Tujuan Penelitian

Tujuan yang ingin dicapai dalam penelitian ini adalah didapatkannya performa modifikasi algoritma *bee colony* dengan *tabu list* yang menghasilkan penjadwalan dengan biaya keterlambatan yang kecil dan waktu komputasi yang singkat.

1.5 Ruang Lingkup Pembahasan

Untuk mendapatkan hasil penelitian yang spesifik dan terarah, maka ruang lingkup permasalahan dari penelitian ini adalah sebagai berikut:

- Data penjadwalan yang digunakan berasal dari penelitian Lina Astuti 2008
- Waktu set-up dan perpindahan semua material sudah termasuk ke dalam waktu proses produksi.
- Satu mesin hanya dapat memproses satu pekerjaan .
- Operasi yang sudah berjalan tidak dapat dihentikan
- Operasi berikutnya tidak dapat diproses sampai operasi sebelumnya selesai dikerjakan.
- Kondisi mesin produksi diasumsikan berjalan dengan normal
- Fungsi tujuan yang ingin diperoleh yaitu meminimumkan biaya keterlambatan.

1.6 Metodologi Penelitian

Berikut adalah langkah-langkah metodologi yang digunakan dalam penelitian ini, sebagaimana tergambarkan pada diagram alir dari metodologi penelitian (gambar 1.2.):

1. Studi Pendahuluan

   Penelitian ini dilakukan melalui studi terhadap algoritma *bee colony* untuk penjadwalan *job shop*.

2. Perumusan Masalah

   Dari berbagai permasalah yang diperoleh dari studi literatur , maka dirumuskanlah masalah yang akan diselesaikan dalam penelitian ini.

3. Tujuan Penelitian

   Dengan inti permasalahan yang ada dan dilakukan studi literatur baik melalui jurnal internasional, laporan penelitian maupun buku teks, maka dirumuskan tujuan dilakukannya penelitian ini.

4. Perencanaan Model (Algoritma)

   Perancangan bahasa progam untuk algoritma *bee colony*, berserta modifikasi algoritma *bee colony* dan algoritma *tabu-search* untuk permasalahan *job shop*.

5. Pengujian dan Analisa Model

   Untuk memastikan model bekerja sesuai dengan tujuan yang diharapkan, maka dilakukan uji verifikasi dan validasi algoritma. Sedangkan pengujian akan dilakukan dua bagian yaitu

   - Pengujian performa algoritma *bee colony-tabu* dengan *tabu search, differential evolution* dan *bee colony* dengan fungsi tujuan minimasi minimasi biaya keterlambatan.

6. Analisa Eksperimen

   Untuk menentukan paramater terbaik dengan menggunakan data yang telah didapat, kemudian akan dilakukan

7. Pengujian Algoritma
8. Analisa performa Algoritma

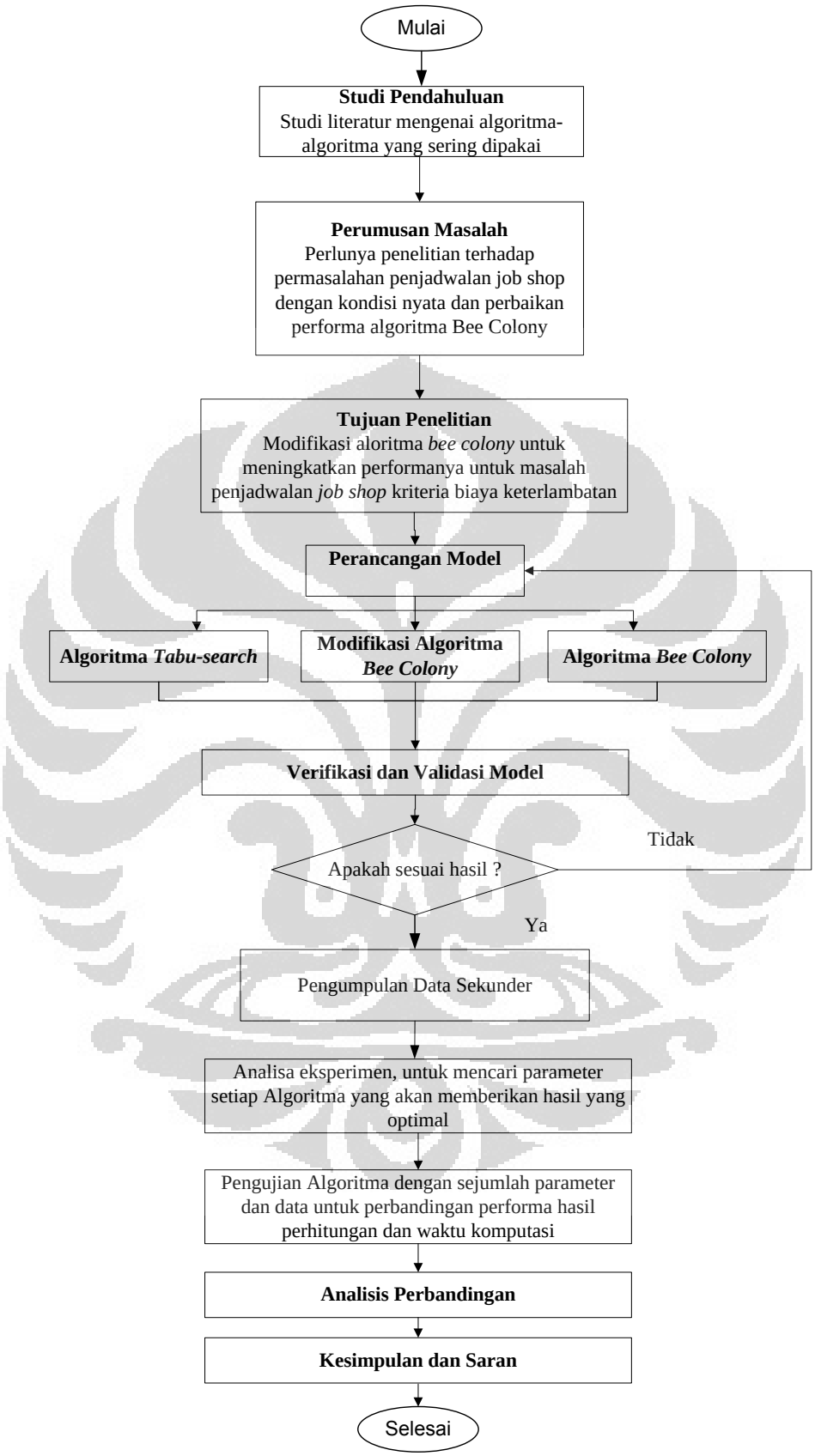
   Menganalisa hasil algoritma.

9. Kesimpulan dan Saran

   Langkah terakhir dalam penelitian ini adalah menyimpulkan hasil-hasil yang dicapai dalam penelitian dan memberikan saran.

1.6.1 Alur Penelitian

Berikut ini adalah alur tahapan penelitian yang digunakan:



Gambar 1.2 Diagram Alur Penelitian

1.7 Sistematika Penulisan

Penelitian tugas akhir ini disusun dalam beberapa bab dengan tujuan memudahkan alur proses berpikir, dengan sistematika sebagai berikut:

Bab pertama merupakan bab pendahuluan, dimana berisi latar belakang masalah, perumusan masalah, tujuan penelitian, ruang lingkup masalah dan asumsi-asumsi, dan sistematika penulisan.Bab kedua merupakan bab yang berisikan teori-teori terkait serta penelitian-penelitian sebelumnya yang berhubungan dengan penelitian ini.

Bab ketiga merupakan bab metodologi, dimana pada bab ini akan dibahas mengenai permasalahan dari data penelitian Lina, model yang dikembangkan, serta pengambilan data sebagai bahan verifikasi dan validasi model yang dikembangkan. Dan dilakukan eksperimen untuk mendapatkan parameter yang memberikan hasil yang optimum

Bab keempat merupakan bab pembahasan yang berisikan hasil pengujian yang didapat dengan algoritma yang telah dikembangkan dan akan digunakan sebagai dasar dalam menganalisa dan menarik kesimpulan.Bab kelima, yaitu bab kesimpulan dan saran. Bab ini berisi kesimpulan dari hasil penelitian yang dilakukan dan saran-saran untuk kelanjutan penelitian.

BAB II
LANDASAN TEORI

Landasan teori merupakan bagian yang berisikan teori-teori yang digunakan untuk menunjukkan bahwa tugas akhir yang dibuat adalah berdasarkan teori yang ada. Teori-teori yang digunakan adalah dasar-dasar mengenai penjadwalan produksi, penjadwalan untuk aliran *Job shop* dan *algoritma bee colony*. Masing-masing teori tersebut akan dijelaskan satu per satu seperti di bawah ini.

2.1. Konsep Penjadwalan

Menurut Morton dan Pentico (1993), "penjadwalan adalah suatu proses pengaturan, pemilihan dan penentuan waktu sumber daya yang berguna untuk menyelesaikan semua aktivitas yang diperlukan untuk memproduksi output yang diinginkan pada waktu yang diharapkan, dimana juga terdapat kendala-kendala diantara aktivitas-aktivitas dan sumber daya yang ada"

Penjadwalan juga didefinisikan sebagai proses pengurutan pembuatan produk secara menyeluruh pada sejumlah mesin tertentu. Penjadwalan juga dipandang sebagai sebagai proses pengalokasian sumber untuk memilih tugas dalam jangka waktu panjang. Penjadwalan merupakan alat ukur yang baik bagi perencana agregat. Pesanan-pesanan aktual pada tahap ini akan ditugaskan pertama kalinya pada sumber daya tertentu (fasilitas, pekerja dan peralatan), kemudian dilakukan pengurutan kerja pada tiap-tiap pusat pemrosesan sehingga dicapai optimalitas utilisasi kapasitas yang ada. (Arman, 2003).

Dimana, tujuan pembuatan penjadwalan menurut Bedworth adalah (Arman, 2003);(1) meningkatkan utilitas atau daya guna dari sumber daya dengan cara mengurangi waktu menganggur dari sumber daya, (2) mengurangi barang setengah jadi, dan (3) mengurangi terjadinya keterlambatan (*tardiness*) baik jumlah pekerjaan atau waktu pekerjaan.

2.1.1. Istilah-Istilah Dalam Penjadwalan

Masalah dalam penjadwalan yang sering muncul adalah terdapat sekumpulan pekerjaan yang menunggu untuk dikerjakan dan hanya tersedia satu sumber daya untuk mengolahnya. Masalah penjadwalan dalam hal ini adalah memutuskan pekerjaan mana yang dikerjakan terlebih dahulu dan mana yang dikerjakan selanjutnya. Penentuan urutan pekerjaan akan mempengaruhi waktu penyelesaian pekerjaan tersebut (*makespan*). Oleh karena itu, perlu dipahami istilah-istilah dan variabel-variabel yang sering digunakan dalam penjadwalan produksi, dimana variabel i merupakan pekerjaan dan variabel j merupakan operasi :

- Waktu Penyelesaian *(C_i)*

  Yaitu waktu yang menunjukkan saat pekerjaan i selesai diproses. Merupakan penjumlahan dari waktu saat pekerjaan i siap dijadwalkan (r_i) ditambah waktu tunggu untuk memulai operasi i setelah operasi ke i-1 selesai (W_i (i-1)) dan waktu proses untuk pekerjaan i (t_i). Berikut ini merupakan formulasi matematikanya :

  $$C_i = r_i + \sum (W_i(i-1) + t_i) \quad \text{(2-1)}$$

  Dimana:

  C_i = Waktu Penyelesaian

  r_i = waktu persiapan pekerjaan ke-i

- Waktu Proses (t_{ij})

  Waktu Proses adalah waktu yang dibutuhkan untuk menyelesaikan operasi ke j dari suatu pekerjaan ke i (termasuk waktu persiapan, penghentian mesin dan waktu pemindahan bahan ke mesin).

- *Makespan* (Total Waktu Produksi) (Ms)

  Makespan adalah waktu yang dibutuhkan untuk menyelesaikan semua pekerjaan yang ada, t_i ada waktu yang dibutuhkan untuk menyelesaikan pekerjaan ke i

  $$Ms = \sum_{i=1}^{n} t_i \quad \text{(2-2)}$$

  Dimana

  n = jumlah pekerjaan yang ada.

t_i = waktu proses pekerjaan ke - i

- *Due Date* (d_i)

  Due date adalah batas waktu penyelesaian dari pekerjaan ke i.

- *Lateness* (L_i)

  Lateness atau keterlambatan adalah penyimpangan waktu penyelesaian suatu pekerjaan ke i hingga batas waktunya .

  $$L_i = C_i - d_i \text{.....(2-3)}$$

  C_i adalah waktu penyelesaian dari pekerjaan ke i

  d_i = *Due date* pekerjaan ke – i

  $L_i < 0$, jika penyelesaian memenuhi batas akhir

  $L_i > 0$, jika penyelesaian melewati batas akhir

- *Tardiness* (T_i)

  Tardiness adalah jumlah keterlambatan penyelesaian suatu pekerjaan hingga batas waktunya. Jika sebuah pekerjaan dapat diselesaikan lebih awal dari batas waktu yang telah ditentukan, maka pekerjaan tersebut dikatakan memiliki keterlambatan negatif dan *zero tardiness*. Namun apabila sebuah pekerjaan memiliki keterlambatan yang positif maka juga memiliki *positif tardiness*.

  $$T_i = \max \{0, L_i\} \text{.....(2-4)}$$

- *Earliness (E_i)*

  Merupakan *lateness* yang bernilai negatif, dimana pekerjaan selesai dikerjakan sebelum *due date* yang ditentukan.

  $$E_i = \min \{ L_i, 0 \} \text{.....(2-5)}$$

  Dimana :

  E_i = *Earliness*

  L_i = *Lateness* pekerjaan ke – i

- *Ready Time (r_i)*, yaitu saat dimana pekerjaan i siap dijadwalkan.

- *Waiting Time (W_i)*

  Yaitu waktu tunggu untuk memulai proses pekerjaan i setelah pekerjaan ke i-1 setelah selesai diproses.

  $$W_i = \sum_{j=1}^{n} W_{ij} \text{ .....(2-6)}$$

- *Flow time (F_i)*

  Yaitu lamanya pekerjaan i berada pada suatu stasiun kerja, rentang waktu antara saat pekerjaan i tiba di stasiun kerja hingga pekerejaan tersebut selesai diproses. Atau dengan kata lain *flow time* merupakan penambahan antara waktu proses dengan waktu menunggu sebelum diproses.

  $$F_i = t_i + W_i \text{.....(2-7)}$$

  $$MFT = \frac{1}{n}\sum_{i=1}^{n} F_i \text{ .....(2-8)}$$

  Dimana :

  F_i = *Flow Time*

  MFT = *Mean Flow time*

  t_i = waktu proses pekerjaan ke - i

  W_i = waktu tunggu pekerjaan ke – i

- *Mean Lateness (L_s)*

  Yaitu rata-rata keterlambatan dari seluruh pekerjaan yang memiliki *due date* masing-masing.

  $$Ls = \frac{1}{n}\sum_{i=1}^{n}(ci - di) \text{ .....(2-9)}$$

  Dimana :

  L_s =*Mean Lateness*

  d_i = *Due date* pekerjaan ke - i

  c_i = *completion time* pekerjaan ke - i

- *Mean Tardiness*

  $$Ts = \frac{1}{n}\sum T_i \text{ .....(2-10)}$$

  Dimana :

  T_s = *Mean Tardiness*

  T_i = *Tardiness* pekerjaan ke – i

- *Number of tardy jobs (N_t)*

  Yaitu jumlah pekerjaan yang mengalami keterlambatan.

  $$N_t = \sum_{i=0}^{n}\delta_i \text{ .....(2-11)}$$

$\delta_i = 1$ jika $T_i \geq 0$

$\delta_i = 0$ jika $T_i = 0$

Dimana :

N_t = *number of tardy* pekerjaan

T_i = *Tardiness* pekerjaan ke – i

2.1.2. Ukuran Kinerja Penjadwalan

Ukuran keberhasilan dari suatu pelaksanaan aktivitas penjadwalan dapat diamati melalui beberapa parameter, seperti berikut : (Arman, 2003)

- Minimasi *makespan* : $C_{max} = \max \{C_i\}$ ……...(2-12)
- Minimasi *Mean Flow Time* : Min ($F = \frac{1}{n}\sum F_i$).....(2-13)
- Minimasi *Mean Tardiness* : $T_s = \frac{1}{n}\sum T_i$ .....(2-14)
- Minimasi *maksimum flow time* : F = max (Fi) …….....….(2-15)
- Minimasi *Mean Lateness* : $L = \frac{1}{n}\sum L_i$ ……..……...(2-16)
- Minimasi maksimum *tardiness* : $T_{max} = \max (T_i)$……..(2-17)

Pada penelitian ini, ukuran kinerja yang digunakan adalah minimasi waktu produksi. Parameter pengukuran kinerja penjadwalan dapat dibagi menjadi beberapa kriteria optimalisasi. Berikut macam – macam kriteria optimalisasi yang digunakan dalam proses penjadwalan adalah: (Michael, 1995)

1. Berkaitan dengan waktu

   Berkaitan dengan waktu, beberapa kriteria optimalisasi yang dapat digunakan adalah :

   - Minimasi mean *flow time*, kriteria ini menunjukkan rata-rata waktu yang dihabiskan setiap komponen di lantai produksi
   - Minimasi *makespan* (waktu proses), *makespan* adalah jumlah waktu yang dibutuhkan untuk menyelesaikan seluruh proses pada semua komponen yang dijadwalkan mulai dari saat pemrosesan komponen pertama sampai dengan komponen terakhir selesai diproses.
   - Penentuan *due date*, *due date* merupakan batas waktu yang ditentukan konsumen agar produk yang dipesannya sudah siap atau selesai.

2. Berkaitan dengan ongkos

   Kriteria ini lebih mengarah ke biaya produksi, seperti : *inventory cost*, *penalty cost* dan lain-lain dan tidak memperhatikan kriteria waktu yang ada sehingga dengan suatu penjadwalan produksi tertentu diharapkan mendapat ongkos yang minimum.

3. Kriteria gabungan

   Beberapa kriteria optimalitas tersebut dapat digabung dan dikombinasikan sehingga menjadi beberapa kriteria yang sesungguhnya (penjadwalan dengan multi kriteria).

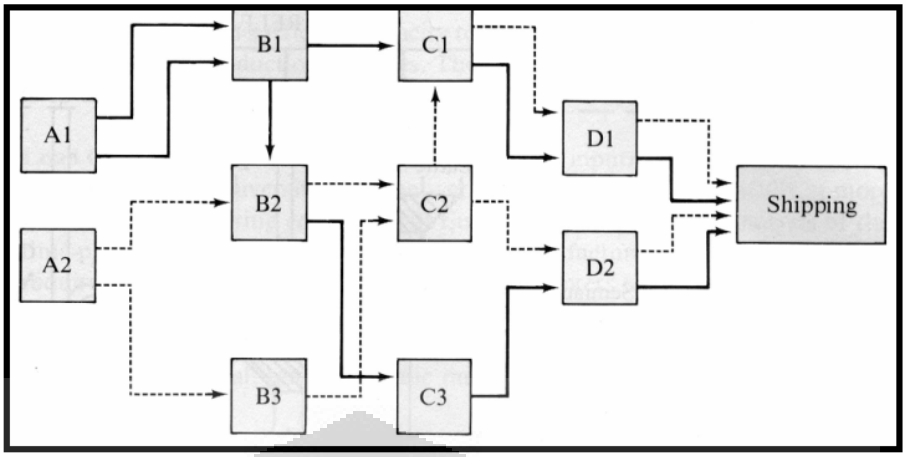
2.2. Penjadwalan *Job Shop*

Job Shop adalah suatu lingkungan manufaktur dimana *job-job* yang datang memiliki rute pengerjaan atau operasi yang seringkali tidak sama. Bentuk sederhana dari model ini mengasumsikan bahwa setiap *job* hanya melewati satu jenis mesin sebanyak satu kali dalam rutenya pada proses tersebut. Namun ada juga model lainnya dimana setiap *job* diperbolehkan untuk melewati mesin sejenis lebih dari satu kali pada rutenya. Model ini disebut juga *job Shop* dengan *recirculation* (pengulangan).

Karakteristik penjadwalan *job Shop* dapat dijabarkan sebagai berikut:

- Ada sejumlah m mesin dan sejumlah n job.
- Setiap job terdiri dari satu rantai urutan yang dapat berbeda satu sama lain.
- Setiap operasi dalam job diproses oleh salah satu mesin yang ada dengan waktu proses yang diasumsikan tetap.
- Setiap proses operasi dapat melewati satu jenis mesin lebih dari satu kali.
- Tidak ada *preemption* (penundaan satu *job* oleh *job* lain).
- Permasalahan penjadwalan untuk model *job Shop* merupakan salah satu permasalahan optimasi kombinatorial yang kompleks sehingga disebut *NP-hard* (*NP* merupakan singkatan dari *nondeterministic polynomial*).

Bentuk permasalahan penjadwalan model *job shop* dapat digambarkan dalam bentuk sepeti di bawah ini:



Gambar 2.1. Contoh Rute Penjadwalan *Job Shop*

2.3. Algoritma Heuristik

Heuristik berasal dari kata Yunani yang berarti menemukan. Metode ini pertama kali digunakan oleh Simon dan Newll untuk mengambarkan pendekatan tertentu untuk memecahkan masalah dan membuat keputusan. Model heuristik ini menggunakan aturan-aturan yang logis dalam memecahkan masalah. Inti dari pendekatan secara heuristik adalah untuk mengaplikasikan rutin secara selektif yang mengurangi bentuk permasalahan. Sebagai contoh, masalah produksi. Bentuk lain dari pengurangan adalah digunakan pada aturan yang relatif sederhana yaitu diterapkan secara berulang sampai semua hasil keputusan telah dibuat.

Algoritma heuristik dalam permasalahan praktis, merupakan salah satu cara untuk mendapatkan solusi yang cukup baik dalam waktu yang cukup cepat. Tetapi model heuristik tidak menjamin hasil yang optimal, tetapi model ini dirancang untuk menghasilkan strategi yang relatif baik dengan mengacu pada pembatasan-pembatasan tertentu. Terdapat kelas dari algortima ini yang menggunakan pencarian secara acak yaitu metode metaheuristik. Yang dapat digunakan untuk rentang permasalahan yang lebih luas, namun dalam performansi tidak bisa dijamin keberhasilannya.

2.3.1. Algoritma Metaheuristik

Algoritma metaheuristik didefinisikan sebagai metode penyelesaian yang mengendalikan interaksi antara improvisasi lokal (*local improvement*) dan strategi

tingkat tinggi (*higher level strategies*) untuk menciptakan proses yang mampu lolos dari jebakan optimum lokal, sekaligus melakukan pencarian penyelesaian yang lebih baik. Karena sifatnya yang stokastik, penggunaan komputer digital mutlak diperlukan. Terdapat beberapa algoritma metaheuristik yang saat ini populer digunakan yaitu :

1. ***Simulated annealing*** (SA) adalah salah satu algoritma untuk untuk optimisasi yang bersifat generik. Berbasiskan probabilitas dan mekanika statistik, algoritma ini dapat digunakan untuk mencari pendekatan terhadap solusi optimum global dari suatu permasalahan. Masalah yang membutuhkan pendekatan SA adalah masalah-masalah optimisasi kombinatorial, di mana ruang pencarian solusi yang ada terlalu besar, sehingga hampir tidak mungkin ditemukan solusi eksak terhadap permasalahan itu. Publikasi tentang pendekatan ini pertama kali dilakukan oleh Kirkpatrick, Gelatt dan Vecchi, diaplikasikan pada desain optimal hardware komputer, dan juga pada salah satu masalah klasik ilmu komputer yaitu *Traveling Salesman Problem.*
2. ***Tabu Search*** adalah metode optimasi yang menggunakan *short-term memory* untuk menjaga agar proses pencarian tidak terjebak pada nilai *optima local*. Metode ini menggunakan *tabu list* untuk menyimpan sekumpulan solusi yang baru saja dievalusi.
3. **Algoritma genetik** adalah teknik pencarian yang di dalam ilmu komputer untuk menemukan penyelesaian perkiraan untuk optimisasi dan masalah pencarian. Algoritma genetik adalah kelas khusus dari algoritma evolusioner dengan menggunakan teknik yang terinspirasi oleh biologi evolusioner seperti warisan, mutasi, seleksi alam dan rekombinasi (atau *crossover*)
4. **Algoritma different evolution** merupakan metode metaheuristik akhir. Metode ini terbilang cukup baru, merupakan versi pengembangan dari Algoritma Genetika. Prinsipnya adalah berdasarkan analogi evolusi biologi, yang terdiri dari proses penginilisasian populasi, proses mutasi, proses penyilangan, dan proses penyeleksian. Keunggulan algoritma ini adalah berstruktur

sederhana, mudah dalam pengimplementasian, cepat dalam mencapai solusi, dan bersifat tangguh (memiliki standar deviasi yang kecil).

5. **Algoritma semut** diperkenalkan oleh Moyson dan Manderick dan secara meluas dikembangkan oleh Marco Dorigo, merupakan teknik probabilistik untuk menyelesaikan masalah komputasi dengan menemukan jalur terbaik melalui grafik. Algoritma ini terinspirasi oleh perilaku semut dalam menemukan jalur dari koloninya menuju makanan.
6. **Algoritma koloni lebah (*Bee Colony Algorithm*)** diperkenalkan oleh Dervis Karaboga dan Bahriye Basturk, Algoritma ini terinspirasi oleh perilaku lebah dalam menemukan lokasi dan posisi dari koloninya menuju makanan.

2.4. Algoritma *Tabu Search* (TS)

Tabu search merupakan suatu metode optimasi matematis yang termasuk ke dalam kelas *local search.* TS memperbaiki performansi *local search* dengan memanfaatkan penggunaan struktur memori. TS diperkenalkan pertama kali oleh Glover (Glover, 1986), dengan ide dasar disampaikan oleh Hansen (Hansen,1986). Banyak eksperimen menujukan bahwa TS saat ini menjadi suatu tejnih optimasi yang dapat diadu hampir semua teknik optimasi yang telah ada (Suyanto,2010).

Proses pencarian bergerak dari satu solusi ke solusi berikutnya, dengan cara memilih solusi terbaik *neighbourhood* solusi sekarang (*current*) yang tidak tergolong solusi terlarang (*tabu*). Ide dasar dari algoritma *tabu search* adalah mencegah proses pencarian dari *local search* agar tidak melakukan pencarian ulang pada ruang solusi yang sudah pernah ditelusuri, dengan memanfaatkan suatu struktur memori yang mencatat sebagian jejak proses pencarian yang telah dilakukan.

Struktur memori fundamental dalam *tabu search* dinamakan *tabu list*. *Tabu list* menyimpan atribut dari sebagian *move* (transisi solusi) yang telah diterapkan pada iterasi-iterasi sebelumnya. *Tabu search* menggunakan *tabu list* untuk menolak solusi-solusi yang memenuhi atribut tertentu guna mencegah

proses pencarian mengalami *cycling* pada daerah solusi yang sama, dan menuntun proses pencarian menelusuri daerah solusi yang belum dikunjungi. Tanpa menggunakan strategi ini, *local search* yang sudah menemukan solusi optimum local dapat terjebak pada daerah solusi optimum local tersebut pada iterasi-iterasi berikutnya. *List* ini mengikuti aturan LIFO dan biasanya sangat pendek (panjangnya biasanya sebesar $O(\sqrt{N})$, dimana *N* adalah jumlah total dari operasi). Stiap saat ada langkah itu akan ditempatkan dalam *tabu list*

Tabu list hanya menyimpan langkah transisi (*move*) yang merupakan lawan atu kebalikan dari langkah yang telah digunakan dalam iterasi sebelumnya untuk bergerak dari satu solusi ke solusi berikutnya. Dengan kata lain *tabu list* berisi langkah-langkah yang membalikan solusi yang baru ke solusi yang lama (Glover, E. Et al,1993).

Pada tiap iterasi, dipilih solusi baru yang merupakan solusi terbaik dalam *neighbourhood* dan tidak tergolong sebagai tabu. Kualitas solusi baru ini tidak harus lebih baik dari kualitas solusi sekarang. Apabila solusi baru ini memiliki nilai fungsi objektif lebih baik dibandingkan solusi terbaik yang telah dicapai sebelumnya, maka solusi baru ini dicatat sebagai solusi terbaik yang baru. Apabila terdapat *move* yang dinilai dapat menghasilkan solusi yang dinilai dapat menghasilkan solusi yang baik namun *move* tersebut berstatus tabu, maka *move* tersebut dapat digunakan utnuk membentuk solusi berikutnya (status tabunya dibatalkan). Hal ini merupakan kondisi khusus pada *tabu list* yang dikenal dengan kriteria aspirasi atau kondisi aspirasi.

Terdapat 3 (tiga) strategi utama yang digunakan dalam TS, yaitu (Pham,2000):

1. Strategi pelanggaran (*the forbidding strategy*) untuk mengontrol apa saja yang boleh masuk ke *tabu list*.
2. Strategi pembebasan (*the freeing strategy*) untuk memutuskan apa saja yang boleh dikeluarkan dari *tabu list* dan kapan pengeluaran dilakukan.
3. Strategi jangka pendek (*the short term strategy*) yang mengatur interaksi antara strategi pelanggran dan strategi pembebasan untuk membangkitkan dan menseleksi solusi-solusi percobaan.

Tabu search memiliki lima parameter utama yang harus ditentukan, yaitu (Hillier and Lieberman,2005) prosedur *local search*, struktur ketetanggaan , kondisi tabu, kondisi aspirasi, dan kriteria pengehentian. Algoritma TS bisa dihentikan berdasarkan kriteria tertentu, misalnya sejumlah iterasi yang ditentukan user, sejumlah waktu, atau sejumlah iterasi berurutan yang tidak menghasilkan nilai fungsi objektif terbaik, dan sebagainya. TS memiliki lima unsur dasar yaitu:

1. Langkah utama untuk memanfaatkan memori dalam TS adalah mengklasifikasikan suatu subhimpunan dalam suatu ketetanggaan sebagai larangan atau tabu.
2. Suatu ketetanggaan dibangun untuk mengidentifikasi solusi-solusi tetangga yang dapat dicapai dari solusi saat ini.
3. Kalsifikasi bergantung pada sejarah pencariakn dan khusunya pasa kebaruan (*recency*) atau frejuensi bahwa langkah atau komponen solusi tertentu, yang disebut atribut, telah berpartisipasi pada pembangkitan solusi-solusi sebelumnya.
4. Suatu *tabu list* mencatat langkah-langkah terlarang atau *tabu moves*.
5. Batasan-batasan tabu bisa diberikan pengecualian. Ketika suatu langkah tabu memberikan suatu solusi yang lebih baik dibandingkan langkah-langkah sebelumnya, maka langkah tersebut dapat digunakan utnuk membentuk solusi berikutnya (status tabunya dibatalkan) atau yang disebut *criteria aspirasi* atau kondisi aspirasi.

2.5. Algoritma *Bee Colony*

Algoritma lebah ini merupakan algoritma yang terinspirasi dari kebiasaan eksplorasi lebah (*forgaging*) untuk mencari solusi optimal. Proses ekplorasi (*forgaging*) dimulai dari mengirimkan lebah pengintai untuk mencari bunga yang berperluang memiliki madu yang banyak. Lebah pengintai tersebut bergerak secara acak dari satu tangkai bunga ke tangkai bunga yang lainnya. Pada saat musim panen, koloni tetap melakukan ekplorasinya, dan tetap mempertahankan populasi dari lebah pengintai. (Seeley TD et al, 1996).

Ketika lebah pengintai kembali ke sarang dan menemukan bunga dengan kadar gula/madu yang dianggap cukup tinggi daripada yang diharapkan, akan mengambil nektarnya sebagai sampel lalu melakukan tarian untuk memberikan lokasi bunga tersebut yang disebut dengan ”*waggle dance*”. (Von Frisch K, 1976). Informasi ini membantu koloni untuk mengirimkan lebah yang lainnya ke bunga tersebut. Dan tarian ini juga digunakan untuk mengevaluasi keuntungan dari bunga yang lainnya, yang berdasarkan energi yang diperlukan, dan jumlah hasil yang mereka bisa dapatkan. Setelah melakukan tarian, lebah pengintai tersebut akan kembali ke bunga yang telah ditemukan dan diikuti oleh beberapa lebah lainnya, hal ini diperuntukan untuk mencari bunga berkualitas yang lainnya. Sehingga dengan ini koloni mampu untuk mengumpulkan makanan dengan cepat dan efisien

Proses algoritma Bee Colony Optimalisasi secara umum dibagi dalam beberapa tahap, yaitu: (Pham D.T., et al, 2006)

1. Tentukan jumlah *list* solusi terbaik, jumlah lebah dan jumlah iterasi.
2. Melakukan sebanyak jumlah lebah pencarian terhadap aera solusi yang telah ditentukan.
3. Tiap calon solusi akan diuji perfomansinya dengan menggunakan *fitness test* atau memilih solusi terbaik.
4. Solusi yang memiliki nilai tinggi akan dipilih untuk dilakukan *neighbourhood search*, dengan sebanyak jumlah lebah.
5. Membandingkan solusi baru yang telah didapat terhadap solusi yang ada pada *list* solusi terbaik. Apabila solusi baru memiliki nilai terbaik, maka solusi tersebut dapat menggantikan solusi pada *list* solusi terbaik.
6. Dilakukan berulang hingga kriteria berhenti tercapai. Dan dipilih yang memiliki nilai tertinggi.

BAB 3

METODE PENELITIAN

Bab ini menjelaskan langkah-langkah yang dijalankan dalam penelitian ini. Langkah-langkah tersebut adalah pengumpulan data, formulasi model permasalahan, pengembangan algoritma, verifikasi dan validasi serta pengujian dan evaluasi algoritma. Dimana dalam penelitian ini akan dilakukan dua pengujian dimana yang pertama pengujian untuk mencari nilai *makespan* yang bertujuan untuk membuktikan pernyataan penelitian Chong, dan pengujian modifikasi algoritma *bee colony* untuk penjadwalan *job shop* dengan kriteria biaya keterlambatan.

3.1. Pengumpulan Data

Data yang digunakan untuk pengujian dan evaluasi algoritma diperoleh dari penelitian sebelumnya (Astuti, L 2008). Adapun objek yang diteliti adalah penjadwalan produksi komponen *Revo Frame*. *Revo Frame* adalah komponen utama untuk produk *Hydraulic Excavator*.

Data yang digunakan adalah data Proses pengerjaan untuk paket pesanan pada periode 24 Januari sampai Februari 2008. Total *Revo Frame* yang harus diproduksi pada periode tersebut adalah 90 buah, yang dapat dilihat pada tabel 3.1

Tabel 3.1 Data Pesanan Periode Januari – Februari 2008

Jenis Revo Frame	Januari 2008	Februari 2008	Total
PC 100F-6		10	10
PC 200-7	9	6	15
PC 300-7	10	12	22
PC 300-8	3	6	9
PC 300LC-8		21	21
PC 400LC-7		13	13

3.1.1. Permasalahan

Dalam penelitian Lina permasalahan yang ada pada PT.X adalah sebagai berikut, kebijakan untuk pesanan diterima kurang lebih 3 bulan sebelum *due date*-

nya. Keadaan ini merupakan kebijakan perusahaan sehingga PPIC dapat menyusun MPS, MRP dan rencana pemesanan barang dahulu . Apabila PPIC merasa bahwa permintaan dapat dipenuhi sesuai *due date*-nya, maka PPIC akan menyusun *Baan Process Planned, MRP Order Lead Time Simulation*, yang merupakan jadwal kapan material harus diantar, kapan fabrikasi harus mulai produksi, kapan *assembly* harus mulai produksi, dan lain-lain. Dalam *Baan Process* juga disertai dengan *lead time* setiap operasi.

Serta, data pesanan yang akan dibahas hanya untuk *stage* fabrikasi dan hanya pada komponen *Revo Frame*. Komponen ini diambil menjadi bahan penelitian ini karena komponen inilah yang harus pertama kali tiba di *stage assembly* untuk membuat produk jadi yang disebut *Hydraulic Excavator*. Jadi, ketika PPIC selesai membuat *Baan Process*, maka akan disusun *Production Order* (PO) yang berisi *due date* setiap *job* pada *stage* fabrikasi ke *stage assembly*. Setiap paket pesanan berbeda jenis, jumlah pesanannya, dan *due datenya*.

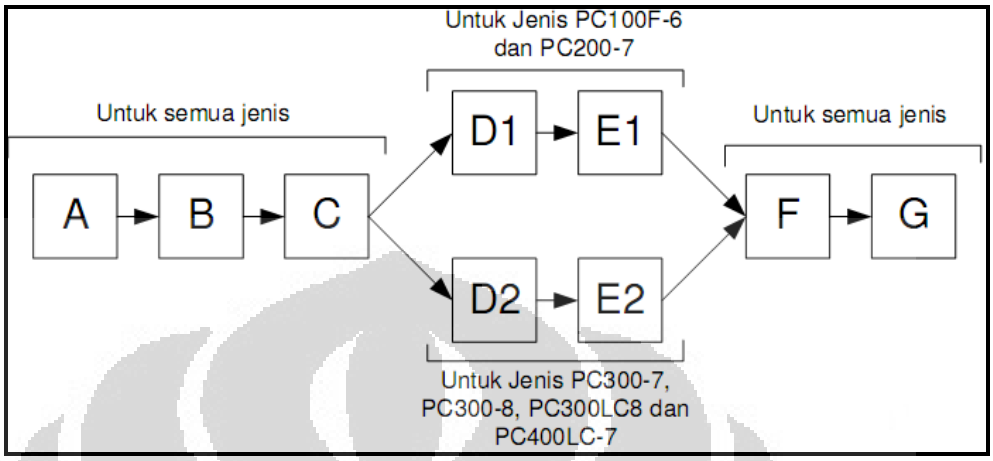
3.1.2. Rute dan Waktu Operasi

Rute proses operasi *Revo Frame* merupakan rute *job -shop* karena tidak semua barang melewati rute yang sama. Pengerjaan *Revo Frame* terdiri dari 7 proses yang dimulai dengan *Straightening Press* (STP) dan diakhiri dengan proses painting (pengecatan). Setelah komponen selesai dicat, maka komponen tersebut siap untuk dikirim ke bagian perakitan. Tabel 3.2. merupakan urutan proses produksi yang harus dilalui untuk memproduksi R*evo Frame*, jumlah mesin atau peralatan untuk tiap proses, dan alokasinya.

Tabel 3.2 Jumlah dan Alokasi Mesin setiap Rute Operasi

Kode Proses	Nama Proses	Jumlah Mesin	Alokasi
A	Straightening Press	1	Semua Jenis
B	Machining	1	Semua Jenis
C	Washing	1	Semua Jenis
D	Tack Welding	2	1 mesin untuk PC100 dan PC200
			1 mesin untuk semua jenis PC300 dan PC400
E	Semi Automated Welding	2	1 mesin untuk PC100 dan PC200
			1 mesin untuk semua jenis PC300 dan PC400
F	Finishing	1	Semua Jenis
G	Painting	1	Semua Jenis

Untuk lebih jelas mengenai urutan rute operasi *Revo Frame*, dapat dilihat pada gambar 3.1. dan untuk waktu proses operasi *Revo Frame* dapat dilihat pada tabel 3.3



Gambar 3.1 Rute Operasi *Revo Frame*

Tabel 3.3. Waktu Operasi *Revo Frame*

Jenis Revo Frame	Waktu Proses (Menit)								
	A	B	C	D1	D2	E1	E2	F	G
PC 100F-6	50	140	15	150	0	120	0	90	155
PC 200-7	60	140	15	210	0	240	0	150	155
PC 300-7	120	140	15	0	210	0	300	160	180
PC 300-8	120	140	15	0	210	0	300	160	180
PC 300LC-8	120	140	15	0	210	0	300	160	180
PC 400LC-7	120	150	15	0	210	0	340	150	200

3.1.3. Jadwal Produksi PT X Periode Januari - Februari 2008

Dalam membuat jadwal produksi, jumlah pesanan untuk setiap tipe dipecah lagi menjadi satuan *job*. Sebagai contoh, jika *Revo Frame* PC100-F6 dipesan sebanyak 10 unit, dikarenakan setiap mesin hanya bisa mengerjakan satu operasi sehingga susunan mesin disebut seri (tidak ada yang paralel), maka pesanan akan diproduksi tidak sekaligus 10 unit, tetapi dipecah menjadi 10 *job*. Jadi, ada 90 *job* yang akan dijadwalkan oleh PT KI pada periode Januari – Februari 2008. *Job-job* tersebut akan melewati 9 mesin (rute proses). Selanjutnya 90 pesanan tersebut akan dibuat urutan pekerjaannya.

Tabel 3.4. Penjadwalan PT X Lengkap (Periode Januari – Februari 2008)

Nomor	Kode	RUTE Proses (Menit)										
Urut	Komponen	A	B	C	D1	D2	E1	E2	F	G	Due Date	Penalti
1	PC300-7 KI84-001	120	140	15	0	210	0	300	160	180	1760	1.87
2	PC300-7 KI84-002	120	140	15	0	210	0	300	160	180	1760	1.87
3	PC300-7 KI84-003	120	140	15	0	210	0	300	160	180	1760	1.87
4	PC300-7 KI84-004	120	140	15	0	210	0	300	160	180	1760	1.87
5	PC300-7 KI84-005	120	140	15	0	210	0	300	160	180	2640	1.87
6	PC300-7 KI84-006	120	140	15	0	210	0	300	160	180	2640	1.87
7	PC300-7 KI84-007	120	140	15	0	210	0	300	160	180	3520	1.87
8	PC300-7 KI84-008	120	140	15	0	210	0	300	160	180	3520	1.87
9	PC300-7 KI84-009	120	140	15	0	210	0	300	160	180	3520	1.87
10	PC300-7 KI84-010	120	140	15	0	210	0	300	160	180	3520	1.87
11	PC300-8 KI84-001	120	140	15	0	210	0	300	160	180	3520	1.95
12	PC300-8 KI84-002	120	140	15	0	210	0	300	160	180	3520	1.95
13	PC200-7 KI84-001	60	140	15	210	0	240	0	150	155	4400	1.14
14	PC200-7 KI84-002	60	140	15	210	0	240	0	150	155	4400	1.14
15	PC200-7 KI84-003	60	140	15	210	0	240	0	150	155	4400	1.14
16	PC300-8 KI84-003	120	140	15	0	210	0	300	160	180	4400	1.95
17	PC200-7 KI84-004	60	140	15	210	0	240	0	150	155	5280	1.14
18	PC200-7 KI84-005	60	140	15	210	0	240	0	150	155	5280	1.14
19	PC200-7 KI84-006	60	140	15	210	0	240	0	150	155	5280	1.14
20	PC200-7 KI84-007	60	140	15	210	0	240	0	150	155	5280	1.14
21	PC200-7 KI84-008	60	140	15	210	0	240	0	150	155	5280	1.14
22	PC200-7 KI84-009	60	140	15	210	0	240	0	150	155	5280	1.14
23	PC200-7 KI84-010	60	140	15	210	0	240	0	150	155	6160	1.14
24	PC200-7 KI84-011	60	140	15	210	0	240	0	150	155	6160	1.14
25	PC200-7 KI84-012	60	140	15	210	0	240	0	150	155	6160	1.14
26	PC200-7 KI84-013	60	140	15	210	0	240	0	150	155	6160	1.14
27	PC200-7 KI84-014	60	140	15	210	0	240	0	150	155	7040	1.14
28	PC200-7 KI84-015	60	140	15	210	0	240	0	150	155	7040	1.14
29	PC300-8 KI84-004	120	140	15	0	210	0	300	160	180	7920	1.95
30	PC300-8 KI84-005	120	140	15	0	210	0	300	160	180	7920	1.95
31	PC300LC-8 KI84-001	120	140	15	0	210	0	300	160	180	7920	2.04
32	PC400LC-7 KI84-001	120	150	15	0	240	0	340	150	200	7920	2.39
33	PC300LC-8 KI84-002	120	140	15	0	210	0	300	160	180	8800	2.04
34	PC300LC-8 KI84-003	120	140	15	0	210	0	300	160	180	8800	2.04
35	PC300LC-8 KI84-004	120	140	15	0	210	0	300	160	180	8800	2.04
36	PC400LC-7 KI84-002	120	150	15	0	240	0	340	150	200	8800	2.39
37	PC100F-6 KI84-001	50	140	15	150	0	120	0	90	155	9680	1.00
38	PC300LC-8 KI84-005	120	140	15	0	210	0	300	160	180	9680	2.04
39	PC300LC-8 KI84-006	120	140	15	0	210	0	300	160	180	9680	2.04
40	PC300LC-8 KI84-007	120	140	15	0	210	0	300	160	180	9680	2.04
41	PC100F-6 KI84-002	50	140	15	150	0	120	0	90	155	10560	1.00
42	PC100F-6 KI84-003	50	140	15	150	0	120	0	90	155	10560	1.00
43	PC100F-6 KI84-004	50	140	15	150	0	120	0	90	155	10560	1.00
44	PC100F-6 KI84-005	50	140	15	150	0	120	0	90	155	11440	1.00
45	PC400LC-7 KI84-003	120	150	15	0	240	0	340	150	200	11440	2.39
46	PC400LC-7 KI84-004	120	150	15	0	240	0	340	150	200	11440	2.39

Tabel 3.4. Penjadwalan PT X Lengkap (Periode Januari – Februari 2008)
Lanjutan

Nomor	Kode	RUTE Proses (Menit)										
Urut	Komponen	A	B	C	D1	D2	E1	E2	F	G	Due Date	Penalti
46	PC400LC-7 KI84-004	120	150	15	0	240	0	340	150	200	11440	2.39
47	PC400LC-7 KI84-005	120	150	15	0	240	0	340	150	200	11440	2.39
48	PC400LC-7 KI84-006	120	150	15	0	240	0	340	150	200	11440	2.39
49	PC300-7 KI84-011	120	140	15	0	210	0	300	160	180	12320	1.87
50	PC300-7 KI84-012	120	140	15	0	210	0	300	160	180	12320	1.87
51	PC400LC-7 KI84-007	120	150	15	0	240	0	340	150	200	12320	2.39
52	PC400LC-7 KI84-008	120	150	15	0	240	0	340	150	200	12320	2.39
53	PC400LC-7 KI84-009	120	150	15	0	240	0	340	150	200	12320	2.39
54	PC400LC-7 KI84-010	120	150	15	0	240	0	340	150	200	12320	2.39
55	PC300-7 KI84-013	120	140	15	0	210	0	300	160	180	13200	1.87
56	PC300-7 KI84-014	120	140	15	0	210	0	300	160	180	13200	1.87
57	PC300-7 KI84-015	120	140	15	0	210	0	300	160	180	13200	1.87
58	PC300-7 KI84-016	120	140	15	0	210	0	300	160	180	13200	1.87
59	PC300-7 KI84-017	120	140	15	0	210	0	300	160	180	13200	1.87
60	PC100F-6 KI84-006	50	140	15	150	0	120	0	90	155	14080	1.00
61	PC300-7 KI84-018	120	140	15	0	210	0	300	160	180	14080	1.87
62	PC300-7 KI84-019	120	140	15	0	210	0	300	160	180	14080	1.87
63	PC100F-6 KI84-007	50	140	15	150	0	120	0	90	155	14960	1.00
64	PC100F-6 KI84-008	50	140	15	150	0	120	0	90	155	14960	1.00
65	PC100F-6 KI84-009	50	140	15	150	0	120	0	90	155	15840	1.00
66	PC100F-6 KI84-010	50	140	15	150	0	120	0	90	155	15840	1.00
67	PC300LC-8 KI84-008	120	140	15	0	210	0	300	160	180	15840	2.04
68	PC400LC-7 KI84-011	120	150	15	0	240	0	340	150	200	15840	2.39
69	PC400LC-7 KI84-012	120	150	15	0	240	0	340	150	200	15840	2.39
70	PC300LC-8 KI84-009	120	140	15	0	210	0	300	160	180	16720	2.04
71	PC300LC-8 KI84-010	120	140	15	0	210	0	300	160	180	16720	2.04
72	PC400LC-7 KI84-013	120	150	15	0	240	0	340	150	200	16720	2.39
73	PC300-8 KI84-006	120	140	15	0	210	0	300	160	180	17600	1.95
74	PC300-8 KI84-007	120	140	15	0	210	0	300	160	180	17600	1.95
75	PC300-8 KI84-008	120	140	15	0	210	0	300	160	180	17600	1.95
76	PC300-8 KI84-009	120	140	15	0	210	0	300	160	180	17600	1.95
77	PC300LC-8 KI84-011	120	140	15	0	210	0	300	160	180	17600	2.04
78	PC300LC-8 KI84-012	120	140	15	0	210	0	300	160	180	17600	2.04
79	PC300LC-8 KI84-013	120	140	15	0	210	0	300	160	180	17600	2.04
80	PC300LC-8 KI84-014	120	140	15	0	210	0	300	160	180	17600	2.04
81	PC300LC-8 KI84-015	120	140	15	0	210	0	300	160	180	17600	2.04
82	PC300LC-8 KI84-016	120	140	15	0	210	0	300	160	180	17600	2.04
83	PC300-7 KI84-020	120	140	15	0	210	0	300	160	180	18480	1.87
84	PC300-7 KI84-021	120	140	15	0	210	0	300	160	180	18480	1.87
85	PC300-7 KI84-022	120	140	15	0	210	0	300	160	180	18480	1.87
86	PC300LC-8 KI84-017	120	140	15	0	210	0	300	160	180	18480	2.04
87	PC300LC-8 KI84-018	120	140	15	0	210	0	300	160	180	18480	2.04
88	PC300LC-8 KI84-019	120	140	15	0	210	0	300	160	180	18480	2.04
89	PC300LC-8 KI84-020	120	140	15	0	210	0	300	160	180	18480	2.04
90	PC300LC-8 KI84-021	120	140	15	0	210	0	300	160	180	18480	2.04

3.2. Formulasi Model Permasalahan

Dari data dan permasalahan yang telah diidentifikasi di awal, maka dapat diformulasikan sebuah model permasalahan dalam bentuk model matematis. Adapun model matematis tersebut bisa dijelaskan sebagai berikut.

Keterangan Notasi:

t = waktu mulai operasi

t_{ij} = waktu mulai operasi j pada *job* i

τ_{ij} = waktu proses operasi j pada *job* i

t_{aj} = waktu mulai operasi j pada *job* a

τ_{aj} = waktu proses operasi j pada *job* a

t_{bj} = waktu mulai operasi j pada *job* b

τ_{bj} = waktu proses operasi j pada *job* b

C_i = Total Biaya keterlambatan untuk *Job* i

c_i = Biaya keterlambatan untuk *job* i setiap menit.

y_{abj} = apabila *job* a datang dahulu maka y_{abj} bernilai 0,

apabila *job* b datang dahulu maka y_{abj} bernilai 1

Fungsi Tujuan:

Meminimalkan C_{total}

$C_{total} = C_1 + C_2 + ...+C_i$ ……...(3-1)

Batasan-batasan:

Kendala *Job* yang berurutan

$t_{i(j+1)} - t_{ij} \geq \tau_{ij}$ ……...(3-2)

Kendala satu mesin tidak dapat memproses dua *job* secara bersamaan

$t_{aj} \geq t_{bj} + \tau_{bj}$ Atau $t_{aj} - t_{bj} \geq \tau_{bj}$ untuk *job b* datang pertama $\forall b, \forall a, \forall j$

atau

$t_{bj} \geq t_{aj} + \tau_{aj}$ Atau $t_b - t_{aj} \geq \tau_{aj}$ untuk *job a* datang pertama $\forall a, \forall b, \forall j$

Kedua bentuk diatas bukan bentuk linier, sehingga harus diubah menjadi bentuk linier seperti pada bagian berikut:

$M(y_{abj}) + t_{aj} - t_{bj} \geq \tau_{bj}$ ……...(3-3)

$M(1-y_{abj})+t_b-t_{aj} \geq \tau_{aj}$ ……...(3-4)

Dimana, M : Bilangan sangat besar.

y_{abj} { 1 (b setelah a) ; 0 (a setelah b) }

Biaya Keterlambatan C_i

$C_i = c_i \times T_i(t_{i9} + \tau_{i9} - due\ date_i)$ ……...(3-5)

Dimana, T_i : Keterlambatan *job ke-i*

T_i { 1 ($t_{i9} + \tau_{i9} > due\ date_i$) ; 0 ($t_{i9} + \tau_{i9} \leq due\ date_i$) }

Variabel keputusan:

$t_{ij} \geq 0$ ……...(3-6)

$y_{abj} = 1,0$

3.3. Pengembangan Algoritma

Algoritma yang dikembangkan pada penelitian ini terdiri dari tiga macam, yaitu algoritma *Tabu Search, Bee Colony* dan *Bee colony dengan tabu list (Bee Colony-tabu list)*. Dimana algoritma ini diolah dengan menggunakan bahasa pemrograman Matlab. Bahasa pemrograman ini dipilih karena banyak memberi kemudahan untuk mengimplementasikan algoritma-algoritma yang banyak menggunakan operasi matriks. Kemudian ketiga algoritma akan dibandingkan dengan algoritma yang digunakan pada penelitian terdahulu yaitu algoritma *Different Evolution*. Diagram alir untuk algoritma *Tabu Search, Bee Colony* dan *Bee Colony-tabu list* dapat dilihat pada lampiran 1.

3.3.1. Langkah-langkah Umun Penyusunan Algoritma *Bee Colony*

Algoritma *bee colony* yang digunakan dalam penelitian ini merupakan algoritma *bee colony* yang umum dipakai, dimana langkah-langkah yang dimaksud adalah :

1. Inisialisasi solusi awal,
   1.1. Pada awal perhitungan sebelum generasi (iterasi) dimulai, dilakukan input parameter-parameter, yaitu ukuran jumlah populasi lebah, jumlah lebah pengintai, dan panjang *list* solusi yang akan digunakan, serta kriteria berhenti yaitu jumlah iterasi yang dipakai.
   1.2. inisialisai solusi awal menggunakan solusi yang didapat secara acak.
2. Tahap *Forgaging* 1
   2.1. Solusi awal, akan dijadikan acuan sejumlah *n* lebah untuk dilakukan *neighbourhood search*, sehingga didapatkan sejumlah *n* solusi alternatif
3. Tahap *Waggle Dance*
   3.1. Solusi alternatif yang telah didapat,kemudian dilakukan pemilihan solusi sejumlah panjang *list* solusi yang telah ditentukan. Kriteria yang digunakan memilih hasil yang terbaik.
   3.2. Dan dilakukan pemilihan secara acak untuk solusi-solusi yang terbaik untuk dijadikan acuan pencarian *neighbourhood search* oleh *n* lebah.
4. Tahap *Forgaging* 2
   4.1. Solusi yang didapat oleh *n* lebah akan dibandingkan kembali dengan solusi yang ada dalam *list*, solusi baru apabila memiliki nilai yang lebih baik akan menggantikan solusi lama.
5. Tahap *looping*
   5.1. Pada tahap ini pengulangan proses dilakukan hanya pada tahap *waggle dance* dan tahap *forgaging*, hingga kriteria berhenti.
6. Kiteria berhenti / terminasi
   6.1. Pada penelitian ini, kriteria terminasi yang digunakan adalah jumlah iterasi . Proses pembentukan iterasi baru akan terus berulang sampai jumlah iterasi yang telah ditentukan tercapai.

3.3.2. Modifikasi Algoritma *Bee Colony*

Modifikasi yang dilakukan pada algoritma *bee colony* terletak pada *list* solusinya, dimana *list* solusi *bee colony* dibantu dengan *tabu list*. Alasan menggunakan *tabu list* dalam modifikasi algoritma *bee colony* adalah karena pada algoritma *tabu search*, *tabu list* merupakan bagian terpenting dari algoritma *tabu*

search yang berfungsi untuk mencegah didapatnya solusi yang mirip dimana dengan *tabu list* kemungkinan lepas dari *local optimum* cukup besar, serta dapat mencari solusi-solusi yang berpotensial menghasilkan solusi optimal. Berdasarkan keunggulan *tabu list* tersebut, maka penelitian ini menggunakan keunggulan *tabu list* untuk dimasukan kedalam algoritma *bee colony*. Berikut langkah-langkah penyusunan algoritma *bee colony tabu list*

1. Inisialisasi solusi awal,
   1.1. Pada awal perhitungan sebelum generasi (iterasi) dimulai, dilakukan input parameter-parameter, yaitu ukuran jumlah populasi lebah, jumlah lebah pengintai, dan panjang *list* solusi yang akan digunakan, serta kriteria berhenti yaitu jumlah iterasi yang dipakai.
   1.2. inisialisai solusi awal menggunakan solusi yang didapat secara acak.
2. Tahap *Forgaging* 1
   2.1. Solusi awal, akan dijadikan acuan sejumlah *n* lebah untuk dilakukan *neighbourhood search*, sehingga didapatka sejumlah *n* solusi alternatif
3. Tahap *Waggle Dance*
   3.1. Solusi alternatif yang telah didapat,kemudian dilakukan pemilihan solusi sejumlah panjang *list* solusi yang telah ditentukan. Kriteria yang digunakan memilih hasil yang terbaik.
   3.2. Dan dilakukan pemilihan secara acak untuk solusi-solusi yang terbaik untuk dijadikan acuan pencarian *neighbourhood search* oleh *n* lebah.
   3.3. Solusi terbaik akan dimasukan ke *list* solusi
   3.4. Solusi terbaik langkah-langkahnya akan ditabukan untuk mencegah terjebaknya pada local optimum.
4. Tahap *Forgaging* 2
   4.1. Solusi yang didapat oleh *n* lebah akan dibandingkan kembali dengan solusi yang ada dalam *list*, solusi baru apabila memiliki nilai yang lebih baik akan menggantikan solusi lama.
5. Tahap *looping*
   5.1. Pada tahap ini pengulangan proses dilakukan hanya pada tahap *waggle dance* dan tahap *forgaging*, hingga kriteria berhenti.

6. Kiteria berhenti / terminasi

   6.1. Pada penelitian ini, kriteria terminasi yang digunakan adalah jumlah iterasi . Proses pembentukan iterasi baru akan terus berulang sampai jumlah iterasi yang telah ditentukan tercapai.

3.3.3. Langkah-langkah Umun Penyusunan Algoritma *Tabu Search*

Langkah-langkah yang digunakan untuk algoritma *tabu search*, juga menggunakan langkah-langkah yang umum digunakan dalam banyak penelitian mengenai *tabu search*, dimana langkah-langkah yang dimaksud adalah:

1. Inisialisasi solusi awal,

   1.1. Pada awal perhitungan sebelum generasi (iterasi) dimulai, dilakukan input parameter-parameter, yaitu ukuran jumlah iterasi, panjang *tabu list*, dan jumlah solusi tetangga.

   1.2. inisialisai solusi awal menggunakan solusi yang didapat secara acak.

2. Tahap *neighbourhood search*

   2.1. Solusi awal, akan dijadikan acuan untuk dilakukan *neighbourhood search*, sehingga didapatkan sejumlah *n* solusi tetangga.

3. Tahap Seleksi Solusi

   3.1. Solusi alternatif yang telah didapat,kemudian dilakukan pemilihan solusi. Kriteria yang digunakan memilih hasil yang terbaik.

   3.2. Dan dilakukan pemilihan secara acak untuk solusi-solusi yang terbaik untuk dijadikan acuan pencarian *neighbourhood search* kembali.

4. Tahap Pemasukan *Tabu List*

   4.1. Solusi terbaik langkah-langkahnya akan disimpan dan ditabukan.

5. Tahap looping

   5.1. Pada tahap ini pengulangan proses dilakukan hanya pada tahap ke-2 hingga ke-4, hingga kriteria berhenti.

6. Kiteria berhenti / terminasi

   6.1. Pada penelitian ini, kriteria terminasi yang digunakan adalah jumlah iterasi . Proses pembentukan iterasi baru akan terus berulang sampai jumlah iterasi yang telah ditentukan tercapai.

3.4. Verifikasi dan Validasi Program

Pada verifikasi program dilakukan saat melakukan kodifikasi program menggunakan MATLAB, saat akan mengkompilasi kode, dan saat menjalankan program dengan persoalan berskala kecil dengan parameter sembarang. MATLAB menyediakan fasilitas *Evaluate MATLAB Expression*, dimana MATLAB akan menginformasikan kepada *programer* ketika terjadi kesalahan pada proses kodifikasi, pada saat verifikasi dilakukan.

Pada validasi program, akan dilakukan pengujian dengan menggunakan persoalan berskala kecil dengan parameter sembarang. Hal ini bertujuan untuk memeriksa apakah program memberikan hasil perhitungan yang diharapkan atau tidak. Maka, dengan menggunakan persoalan pada tabel 3.5, dimana gambar 3.2 merupakan hasil keluaran untuk algoritma *Tabu Search* dan gambar 3.3 untuk algoritma *Bee Colony*. Hasil yang diberikan akan dibandingkan dengan penelitian sebelumnya yang menggunakan persoalan yang sama untuk validasi program. Dimana hasil yang diberikan kedua algoritma memberikan hasil yang sama dengan penelitian sebelumnya dengan nilai *makespan* berkisar antara 680 menit hingga 770 menit , maka dengan ini program algoritma TS dan Bee sudah tervalidasi.

Tabel 3.5 Data *dummy* untuk Validasi

	Waktu Proses (menit)								
JOB	A	B	C	D1	D2	E1	E2	F	G
1	80	60	30	0	50	0	60	60	50
2	70	40	40	40	0	50	0	80	50
3	80	50	70	0	70	0	80	50	80
4	60	40	50	0	90	0	70	50	70
5	70	90	50	30	0	40	0	60	60

```
Solusi =

     4     2     5     3     1

>> CalculateTimeProcess(Solusi,waktu_proses)

ans =

    60   100   150   150   240   150   310   430   500
   130   170   210   250   240   300   310   380   430
   200   290   340   370   340   410   340   490   560
   280   340   410   410   480   410   560   610   690
   360   420   450   450   530   450   620   680   740
```

Gambar 3.2. Hasil Perhitungan Data *Dummy* dengan Algoritma *Tabu Search*.

```
Solusi =

     3     5     2     1     4

>> waktu_proses=xlsread('Data_percobaan.xlsx')

waktu_proses =

    80    60    30     0    50     0    60    60    50
    70    40    40    40     0    50     0    80    50
    80    50    70     0    70     0    80    50    80
    60    40    50     0    90     0    70    50    70
    70    90    50    30     0    40     0    60    60

>> CalculateTimeProcess(Solusi,waktu_proses)

ans =

    80   130   200   200   270   200   350   400   480
   150   240   290   320   290   360   350   460   540
   220   280   330   370   330   420   350   540   590
   300   360   390   390   440   420   500   600   650
   360   400   450   450   540   450   610   660   730
```

Gambar 3.3. Hasil Perhitungan Data *Dummy* dengan Algoritma *Bee Colony*

```
>> CalculateTimeProcess(Solusi,waktu_proses)

ans =

    80   130   200   200   270   200   350   400   480
   150   240   290   320   290   360   350   460   540
   220   280   330   370   330   420   350   540   590
   300   360   390   390   440   420   500   600   650
   360   400   450   450   540   450   610   660   730
```

Gambar 3.4. Hasil Perhitungan Data *Dummy* dengan Algoritma *Bee Colony-tabu list*

3.5. Penentuan Nilai Parameter

3.5.1. Penentuan Nilai Parameter Untuk Biaya Keterlambatan

Untuk menentukan nilai parameter-parameter algoritma-algoritma yang digunakan untuk pengolahan data, terlebih dahulu dilakukan Design of Experiments (DOE) terhadap parameter-parameter untuk melihat kombinasi terbaik dari parameter –parameter. Untuk *tabu search* parameter yang diujikan adalah panjang *tabu list*, jumlah iterasi, dan jumlah solusi tetangga. Sedangkan untuk *bee colony* adalah. Jumlah populasi (iterasi), jumlah lebah (solusi tetangga) dan panjang *list* solusi.

Skenario yang akan diujikan untuk faktor-faktor terdiri atas 3 (tiga) level setiap faktor, yaitu yaitu level rendah, sedang, dan tinggi untuk algoritma *tabu search*. Sedangkan skenario eksperimen untuk *bee colony*, dapat dilihat pada tabel 3.7.

Tabel 3.6. Skenario Eksperimen Algoritma *Tabu Search*

Faktor	Level 1 (Rendah)	Level 2 (Sedang)	Level 3 (Tinggi)
Jumlah solusi tetangga	30	50	100
Panjang *tabu list*	7	13	90
Jumlah iterasi	100	1500	3000

Tabel 3.7. Skenario Eksperimen Algoritma *Bee Colony*

Jumlah Iterasi	100	1500	3000
Jumlah Lebah	10	50	90
Panjang List	5	20	

Nilai faktor untuk *tabu search* berdasarkan penelitian terdahulu (Astuti, 2008) dimana performa algoritma menjadi stabil setelah menyentuh iterasi ke-2000, dalam skenario terdapat iterasi sebanyak 1500 kali untuk hasil performa yang sebelum stabil dan 3000 kali untuk hasil performa sesudah stabil, dan panjang *tabu list* 7 dan 90 berdasarkan saran Glover untuk permasalahan penjadwalan, dan 13 berdasarkan Rossi (Sheibatolhamdy, 2011)dimana *tabu list* berjumlah setengah dari jumlah bilangan primer dari seluruh operasi.

Sedangkan untuk nilai faktor Algoritma *bee colony* untuk jumlah iterasi 100, 1500, dan 3000 diambil dari jumlah iterasi *tabu search*. Untuk jumlah lebah dan panjang list yang dipakai, merupakan nilai yang cukup sering dipakai dalam penelitian Chong terutama untuk penjadwalan *job shop*.

Pada eksperimen *tabu search* didapatkan bahwa, panjang *tabu list*, dan jumlah iterasi berserta kombinasi dari kedua faktor memberikan efek yang signifikan terhadap biaya keterlambatan yang dihasilkan. Dengan demikian berdasarkan pengamatan akan hasil eksperimen, kombinasi parameter yang memberikan hasil yang mendekati optimal adalah : Tetangga Level 2 : 50; Tabu list Level 2: 7; Dan Jumlah Iterasi Level 2 : 3000.

Pada eksperimen *bee colony* disimpulkan bahwa, jumlah panjang *list* solusi , dan interaksnya dengan jumlah lebah dan atau jumlah iterasi memberikan efek yang signifikan terhadap biaya keterlambatan yang dihasilkan. Dengan demikian kombinasi yang dipakai untuk algoritma *bee colony* dipilih Jumlah lebah : 50; dan jumlah iterasi : 3000, dengan panjang list solusi : 20.

Sehingga parameter-parameter yang diajukan untuk algoritma *tabu* search dan *bee colony* untuk pengolahan data dapat dilihat pada tabel 3.8

Tabel 3.8 Parameter-Parameter Algoritma *Tabu* Search , *Bee Colony* dan *Bee Colony-tabu list* Untuk Pengolahan Data

Parameter	*Tabu Search*	*Bee Colony*	*Bee Colony-tabu list*
Jumlah solusi tetangga	50	50	90
Panjang *tabu list*	7	-	13
Jumlah iterasi	3000	3000	1500
Panjang List	-	20	13

Parameter untuk algoritma *bee colony-tabu list*, didapat dengan melakukan eksperimen dengan menggunakan skenario pada tabel 3.9, yang didapat skenario tersebut (tabel 3.8) memberikan nilai biaya keterlambatan yang kecil. Dimana, faktor ini didapat dari gabungan skenario untuk *tabu search* yaitu *tabu* list-nya dan *bee colony* untuk jumlah lebah dan panjang *list*, sedangkan untuk jumlah iterasi yang nilainya cukup kecil dikarenakan hasil eksperimen sebelumnya nilai biaya keterlambatan sudah cukup stabil pada iterasi ke-1500, sehingga penambahan jumlah iterasi hanya akan menambah waktu komputasi.

Tabel 3.9 Skenario Eksperimen untuk Algoritma *Bee Colony-tabu list*

Tabu list	7	13	
Jumlah Iterasi	1500	3000	
Jumlah Lebah	50	90	100
Panjang *List*	7	13	

3.6. Pengujian dan Evaluasi Performa Algoritma

3.6.1. Pengujian dan Evaluasi Performa Algoritma Untuk Nilai Biaya Keterlambatan

Setelah pengembangan ketiga algoritma selesai, dan didapatkan parameter terbaik untuk permasalahan penjadwalan *job shop Revo Frame*, maka untuk melihat seberapa baiknya performa algoritma *bee colony-tabu list* selanjutnya akan dilakukan pengujian dan evaluasi performa algorima dengan parameter yang mirip untuk ketiga algoritma yang dapat dilihat pada tabel 3.10 untuk jumlah iterasi 100 (seratus) dan tabel 3.11 untuk jumlah iterasi 1000 (seribu), dan soal yang akan dipakai adalah soal penjadwalan yang ada pada tabel 3.4.

Tabel 3.10 Parameter untuk Pengujian dan Evaluasi Performa Algoritma yang Dikembangkan Dengan Jumlah Iterasi 100

Parameter	*Tabu Search*	*Bee Colony*	*Bee Colony-tabu list*	*Differential Evolution*
Jumlah solusi tetangga	10	10	10	-
Panjang *tabu list*	7	-	7	-
Jumlah iterasi	100	100	100	100
Panjang List	-	7	7	-
Ukuran Populasi	-	-	-	10
Operator Mutasi	-	-	-	0,6
Operator Pindah Silang	-	-	-	0,5

Tabel 3.11 Parameter untuk Pengujian dan Evaluasi Performa Algoritma yang Dikembangkan Dengan Jumlah Iterasi 1000

Parameter	*Tabu Search*	*Bee Colony*	*Bee Colony-tabu list*	*Differential Evolution*
Jumlah solusi tetangga	10	10	10	-
Panjang *tabu list*	7	-	7	-
Jumlah iterasi	1000	1000	1000	1000
Panjang List	-	7	7	-
Ukuran Populasi	-	-	-	10
Operator Mutasi	-	-	-	0.6
Operator Pindah Silang	-	-	-	0.5

Tujuan diberikan parameter pada tabel 3.10 dan tabel 3.11 , berdasarkan beberapa pengujian secara *trial and error* pada parameter ini ditemukan variasi hasil untuk ketiga algoritma yang dikembangkan. Sehingga diharapkan parameter pada tabel 3.10 dan tabel 3.11 akan memberikan perbedaan performa antara ketiga algoritma baik dari hasil biaya keterlambatan maupun waktu komputasi.

Pengujian akan sebanyak 2 set *trial*, dimana setiap set terdiri atas 2 kali *run*. Perbandingan akan dilakukan sebanyak tiga kali, yaitu dengan menggunakan parameter pada tabel 3.10 dan tabel 3.11 dan menggunakan parameter terbaik (tabel 3.12) yang setiap algoritma yang telah didapatkan ringkasan dapat dilihat pada tabel 3.13. Program dijalankan dijalankan dengan spesifikasi komputer, yaitu Intel(R) Core(TM)2 Quad CPU Q9550 @2.83Ghz, 4GB of RAM, Sistem Operasi: Microsoft Windows XP.

Pengujian untuk parameter terbaik (tabel 3.12) selain menggunakan data tabel 3.4, juga akan menggunakan data pada lampiran 3. Tujuan menggunakan data-data tersebut (tabel 3.4 dan lampiran 3), untuk melihat konsistensi performa algoritma baik untuk data yang nornal, maupun data yang telah dimodifikasi.

Modifikasi data yang akan digunakan dalam pengujian performa pertama adalah menaikan beberapa waktu proses operasi *job*, kedua menurunkan waktu proses operasi *job*, kedua hal ini bertujuan untuk melihat performa algoritma dalam kondisi dimana mesin sudah menjadi tua sehingga waktu penyelesaian menjadi lebih lama, dan untuk kondisi dimana mesin telah diganti dengan yang baru sehingga waktu proses operasi ada yang menjadi lebih cepat.

Selain modifikasi waktu, pengujian performa juga melakukan modifikasi jumlah *job* yaitu dari 90 menjadi 60 *job* dan 30 *job*, yang bertujuan untuk melihat apakah performa algoritma untuk data yang sedikit akan mengalami kenaikan performa atau tidak.

Tabel 3.12. Parameter Terbaik Setiap Algoritma untuk Perbandingan

Parameter	*Tabu Search*	*Bee Colony*	*Bee Colony-tabu list*	*Differential Evolution*
Jumlah solusi tetangga	50	50	90	-
Panjang *tabu list*	7	-	13	-
Jumlah iterasi	3000	3000	1500	2000
Panjang List	-	20	13	-
Ukuran Populasi	-	-	-	100
Operator Mutasi	-	-	-	0.6
Operator Pindah Silang	-	-	-	0.5

3.7. Hasil Pengujian Performa Ketiga algoritma dan Algoritma *Differential Evolution*

Berikut merupakan hasil pengujian ketiga algoritma dan algoritma *Differential evolution* dengan permasalahan tabel 3.4 dengan parameter pada tabel 3.10 dan tabel 3.11, serta dengan menggunakan parameter terbaik keempat algoritma. Untuk hasil pengujian performa dengan data modifikasi.

Tabel 3.13 Hasil Performa Algoritma *Tabu Search, Bee Colony, Bee Colony-tabu list,* dan *Differential Evolution* dengan Parameter serupa Iterasi : 100

	Trial 1		Trial 2		
	1	2	1	2	Average
Tabu Search					
Biaya keterlambatan	68402.4	79383.5	85491.85	71013.9	**76072.91**
Makespan (min)	20845	20845	20925	20845	20865
Waktu Komputasi (sec)	1.8	1.75	1.83	1.8	1.795
Bee Colony					
Biaya keterlambatan	122176.2	142568.25	133526.8	133448	132929.8125
Makespan (min)	20845	20925	20845	20915	20882.5
Waktu Komputasi (sec)	1.8	1.81	1.81	1.75	1.7925
Bee Colony-tabu list					
Biaya keterlambatan	108317.6	113060	126920.85	108455.85	114188.575
Makespan (min)	20845	20845	20845	20915	20862.5
Waktu Komputasi (sec)	1.78	1.85	1.8	1.8	1.8075
Differential Evolution					
Biaya keterlambatan	99496	104530	117380	97241	104661.75
Makespan (min)	20845	20925	20845	20915	20882.5
Waktu Komputasi (sec)	0.5	0.4	0.4	0.4	0.425

Tabel 3.14 Hasil Performa Algoritma *Tabu Search, Bee Colony, Bee Colony-tabu list,* dan *Differential Evolution* dengan Parameter serupa Iterasi : 1000

	Trial 1		Trial 2		
	1	2	1	2	Average
Tabu Search					
Biaya keterlambatan	33220.25	33136.9	34485.53	33354	**33549.17**
Makespan (min)	20845	20845	20845	20845	20845
Waktu Komputasi (sec)	9.7	9.5	9.51	9.48	9.5475
Bee Colony					
Biaya keterlambatan	67534.85	74709.05	69475.85	66351.85	69517.9
Makespan (min)	20845	20845	20845	20845	20845
Waktu Komputasi (sec)	9.7	9.7	9.74	9.6	9.685
Bee Colony-tabu list					
Biaya keterlambatan	62446.75	64537.75	69077.6	70136.3	66549.6
Makespan (min)	20845	20845	20845	20845	20845
Waktu Komputasi (sec)	9.73	9.6	9.7	9.7	9.6825
Differential Evolution					
Biaya keterlambatan	97758	65181	91754	56721	77853.5
Makespan (min)	20845	20845	20845	20845	20845
Waktu Komputasi (sec)	4	3.97	3.9	3.95	3.955

Tabel 3.15 Hasil Performa Algoritma *Tabu Search, Bee Colony, Bee Colony-tabu list,* dan *Differential Evolution* dengan Data Tabel 3.4

	Trial 1		Trial 2		
	1	2	1	2	Average
Tabu Search					
Biaya keterlambatan	28280.05	28352.05	28352.05	28368.25	28338.1
Makespan (min)	20845	20845	20845	20845	20845
Waktu Komputasi (sec)	119.3	119.9	118.6	118.85	119.1625
Bee Colony					
Biaya keterlambatan	28368.25	28280.05	28368.25	28368.25	28346.2
Makespan (min)	20845	20845	20845	20845	20845
Waktu Komputasi (sec)	120.8	120.3	119.63	119.46	120.0475
Bee Colony-tabu list					
Biaya keterlambatan	28280.05	28302.85	28377.6	28280.05	**28310.14**
Makespan (min)	20845	20845	20845	20845	20845
Waktu Komputasi (sec)	107.91	108.43	107.4	108.04	107.945
Differential Evolution					
Biaya keterlambatan	29895	28378	28434	28378	28771.25
Makespan (min)	20845	20845	20845	20845	20845
Waktu Komputasi (sec)	74.04	73.078	73.98	73.89	73.747

Tabel 3.16 . Hasil Performa Algoritma dengan Parameter Terbaik dan dengan Data Waktu Pekerjaan yang Ditingkatkan (Tabel L3.1)

	Trial 1		Trial 2		
	1	2	1	2	Average
Tabu Search					
Biaya keterlambatan	30639.5	30639.5	30639.5	30639.5	**30639.5**
Makespan (min)	20905	20905	20905	20905	20905
Waktu Komputasi (sec)	118.42	119.24	119.08	118.79	118.883
Bee Colony					
Biaya keterlambatan	30639.95	30770.4	30639.95	30639.5	**30672.5**
Makespan (min)	20905	20905	20905	20905	20905
Waktu Komputasi (sec)	119.3	119.5	119.5	119.3	119.4
Bee Colony-tabu list					
Biaya keterlambatan	30639.5	30639.5	30639.5	30639.5	**30639.5**
Makespan (min)	20905	20905	20905	20905	20905
Waktu Komputasi (sec)	107.75	107.42	107.98	107.82	107.743
Differential Evolution					
Biaya keterlambatan	30949	30787	30770	30674	30795
Makespan (min)	20905	20905	20905	20905	20905
Waktu Komputasi (sec)	74.12	73.95	73.83	73.92	73.955

Tabel 3.17. Hasil Performa Algoritma dengan Parameter Terbaik dan dengan Data Waktu Pekerjaan yang Diturunkan (Tabel L3.2)

	Trial 1		Trial 2		
	1	2	1	2	Average
Tabu Search					
Biaya keterlambatan	26870.7	26973.55	26973.55	26870.7	**26922.13**
Makespan (min)	20815	20815	20815	20815	20815
Waktu Komputasi (sec)	118.51	118.61	119.422	118.23	118.693
Bee Colony					
Biaya keterlambatan	26925	26950.4	26880.05	26880.05	**26908.88**
Makespan (min)	20815	20815	20815	20815	20815
Waktu Komputasi (sec)	119.6	119.5	120.03	120.99	120.03
Bee Colony-tabu list					
Biaya keterlambatan	26880.05	26973.55	26880.05	26870.7	**26901.09**
Makespan (min)	20815	20815	20815	20815	20815
Waktu Komputasi (sec)	109.1	107.67	108.59	108.05	108.3525
Differential Evolution					
Biaya keterlambatan	27887	26896	26896	28515	27548.5
Makespan (min)	20815	20815	20815	20815	20815
Waktu Komputasi (sec)	74.4	74.34	74.5	74.4	74.41

Tabel 3.18. Hasil Performa Algoritma dengan Parameter Terbaik dan dengan Data Jumlah *Job* 60 (Tabel L3.3)

	Trial 1		Trial 2		
	1	2	1	2	Average
Tabu Search					
Biaya keterlambatan	0	0	0	0	0
Makespan (min)	14265	14265	14265	14265	14265
Waktu Komputasi (sec)	84.04	83.61	84	83.76	83.8525
Bee Colony					
Biaya keterlambatan	0	0	0	0	0
Makespan (min)	14265	14275	14265	14265	14267.5
Waktu Komputasi (sec)	84.56	85.34	84.64	85.7	85.06
Bee Colony-tabu list					
Biaya keterlambatan	0	0	0	0	0
Makespan (min)	14265	14265	14265	14265	14265
Waktu Komputasi (sec)	76.41	78.2	77.8	76.3	77.1775
Differential Evolution					
Biaya keterlambatan	0	0	0	0	0
Makespan (min)	14265	14265	14265	14265	14265
Waktu Komputasi (sec)	74.5	74.39	74.38	74.41	74.42

Tabel 3.19. Hasil Performa Algoritma dengan Parameter Terbaik dan dengan Data Jumlah *Job* 30 (Tabel L3.4)

	Trial 1		Trial 2		
	1	2	1	2	Average
Tabu Search					
Biaya keterlambatan	0	0	0	0	0
Makespan (min)	7725	7505	7505	7505	7560
Waktu Komputasi (sec)	49.87	49.52	49.71	49.71	49.7025
Bee Colony					
Biaya keterlambatan	0	0	0	0	0
Makespan (min)	7505	7585	7515	7575	7545
Waktu Komputasi (sec)	50.62	50.36	50.43	50.51	50.48
Bee Colony-tabu list					
Biaya keterlambatan	0	0	0	0	0
Makespan (min)	7505	7575	7505	7575	7540
Waktu Komputasi (sec)	45.69	45.63	45.74	45.7	45.69
Differential Evolution					
Biaya keterlambatan	0	0	0	0	0
Makespan (min)	7595	7545	7575	7725	7610
Waktu Komputasi (sec)	74.1	74.012	74	73.98	74.023

BAB 4
PEMBAHASAN

4.1. Analisi perbandingan Performa Algoritma

Berdasarkan pengujian yang dilakukan untuk masing-masing algoritma, maka dapat dilakukan perbandingan untuk menganalisis performa tiap algoritma. Dari hasil pengujian terhadap data permasalahan, hasil tiap algoritma direkapitulasi yang tersaji pada tabel 4.1. Analisis performa dilakukan dengan mengevaluasi pencapaian biaya keterlambatan terendah dan waktu komputasi terkecil.

Tabel 4.1. Rekapitulasi Hasil Performa Algoritma

	Parameter Serupa		Parameter Terbaik				
	Iterasi : 100	Iterasi : 1000	90 Data	90 Data peningkatan Waktu Proses	90 Data penguranggan Waktu Proses	60 Data	30 Data
Tabu Search							
Biaya keterlambatan	**76072,91**	**33549,17**	28338	**30639,5**	26922,1	0	0
Makespan (min)	20865	20845	20845	20905	20815	14265	7560
Waktu Komputasi (sec)	1,795	9,5475	119,2	118,88	118,7	83,85	49,7
Bee Colony							
Biaya keterlambatan	132929,8	69517,9	28346	30672,45	26908,8	0	0
Makespan (min)	20882,5	20845	20845	20905	20815	14267	7545
Waktu Komputasi (sec)	1,7925	9,685	120	119,4	120,03	85,06	50,48
Bee Colony-tabu list							
Biaya keterlambatan	114188,6	66549,6	**28310**	**30639,5**	**26901,1**	0	0
Makespan (min)	20862,5	20845	20845	20905	20815	14265	7540
Waktu Komputasi (sec)	1,807	9,6825	107,9	107,74	108,35	77,17	45,69
Differential Evolution							
Biaya keterlambatan	104661,8	77853,5	28771	30795	27548,5	0	0
Makespan (min)	20882,5	20845	20845	20905	20815	14265	7610
Waktu Komputasi (sec)	**0,425**	**3,955**	73,75	73,96	74,41	74,42	74,03

Untuk pengujian dengan parameter serupa, algoritma *tabu search* untuk kriteria biaya keterlambatan memberikan hasil yang lebih unggul daripada algoritma lainnya, namun untuk waktu komputasi algoritma *Differential evolution* lebih unggul. Sedangkan algoritma *bee colony-tabu list* memiliki performa yang sedikit lebih unggul daripada *tabu search, bee colony* dan *Differential evolution* untuk pengujian 90 data dengan menggunakan parameter terbaik. Pada tabel 4.1 dilihat untuk algoritma yang menggunakan proses pencarian *neighbourhood search* akan memiliki waktu komputasi yang berbanding lurus dengan jumlah permasalahan (*job),* hal ini dikarenakan semakin sedikitnya *job* akan mempermudah proses pencarian solusi dengan menggunakan pencarian *neighbourhood search.*

4.2 Analisa Perbandingan Hasil Penelitian Sebelumnya

4.2.1. Perbandingan dengan penelitian Chong.

Penelitian Chong pada tahun 2005 yang menggunakan algoritma *bee colony* untuk penjadwalan *job shop*, menggunakan algoritma *bee colony* modifikasi Chong perbedaan dapat dilihat pada tabel dibawah ini

Tabel 4.2 Perbandingan Pendekatan Penggunaan Algoritma *Bee Colony*

	Chong	Penelitian Saat Ini
Pembangkitan Solusi Awal	Metode Forgaging terstruktur	Acak
Pencarian Solusi	Mencari solusi dari nol	Mencari solusi dengan *metode neighbourhood search (swap)*
Waggle Dance	Seleksi solusi	Seleksi solusi
Pemilihan solusi untuk iterasi berikut	*Rating*	Acak dari list terbaik
Permasalahan Penjadwalan yang dipakai	Generate	Data Lapangan

Modifikasi yang dilakukan dalam penelitian ini berdasarkan pernyataan dalam penelitian Chong (Chong, 2005). Dan dengan melakukan pengujian algoritma, maka didapatkan hasil yang dapat dibandingkan dengan hasil penelitian Chong, yang dapat dilihat pada tabel 4.3.

Tabel 4.3 Perbandingan Antara Hasil Penelitian Chong dengan Temuan dalam Penelitian Saat ini

	Chong, 2005	Temuan dalam Penelitian Saat ini
Hasil	Performa *Bee Colony* dibawah *Tabu Search*	Performa *Bee Colony* sedikit dibawah *Tabu Search* baik dengan paramater serupa dan terbaik
Pernyataan 1	*Heuristics under perform tabu search primarily due to: solutions are always constructed from scratch, instead of the more efficient operation swaps in tabu search*	Penggunaan *neighbourhood search* (seperti *swap)* dalam algoritma *bee colony* memberikan performa yang hampir serupa dengan *tabu search*
Pernyataan 2	*Heuristics under perform tabu search primarily due to: no clear scheme to escape from being locked into local minimums*	Penggunaan *tabu list* dalam algoritma *bee colony-tabu list* memberikan performa lebih baik daripada *bee colony*

Berdasarkan penelitian Chong, algoritma selain *tabu search* memiliki performa dibawah *tabu search* (*ant colony dan bee* colony) untuk penjadwalan *job shop* dikarenakan dalam proses pencarian solusi alternatif menggunakan metode yang mencari solusi dari titik awal (membuat solusi) bukan mencari solusi tetangga. Penelitian ini menggunakan pencarian solusi tetangga, sehingga performa algoritma *bee colony* mendekati performa *tabu search*. Sedangkan untuk *bee colony-tabu list* menunjukan performa diatas *tabu search* . Hal ini menunjukan bahwa, pencarian solusi dengan mencari solusi tetangga untuk permasalahan *job shop* dalam penelitian ini akan memberikan performa yang lebih efektif daripada mencari solusi dari titik awal, dan penggunaan *tabu list* memberikan efek yang cukup positif terhadap peningkatan performa algoritma *bee colony*.

4.2.2 Perbedaan dengan penelitian Lina Astuti

Perbandingan juga dilakukan untuk penelitian Astuti (2008). Yang dikarenakan data dari penelitian Astuti digunakan sebagai acuan untuk verifikasi, validasi dan perbandingan performa. Dimana perbandingan pendekatan yang dilakukan dalam penelitian ini dengan Astuti Lina adalah.

Tabel 4.4. Perbandingan Pendekatan Antara Penelitian Astusi Lina dengan Penelitian Saat ini

	Astuti Lina, 2008	Penelitian Ini
Algoritma	*Differential Evolution*	*Bee Colony*
Pembangkitan Solusi Awal	Acak	Acak
Pencarian Solusi Alternatif	Metode mutasi dan pindah silang yang terstruktur	Metode *Neighbourhood search*

Berdasarkan tabel 4.1 menunjukkan bahwa performa algoritma yang menggunakan pencarian terstruktur akan memberikan waktu komputasi yang tetap dan dikarenakan algoritma *Differential evolution* tidak memiliki metode untuk lepas dari *local optimum*, maka solusi yang dihasilkan dibawah solusi yang dihasilkan oleh algoritma *tabu search* dan *bee colony-tabu list* yang memiliki *tabu list* untuk lepas dari *local optimum*.

Keunggulan penelitian sebelumnya terletak pada waktu komputasi yang singkat untuk permasalahan dengan jumlah *job* 90. Namun waktu komputasi untuk jumlah *job* kurang dari 90, akan memberikan waktu yang serupa dengan waktu komputasi untuk permasalahan dengan jumlah *job* 90. Hal ini dikarenakan pencarian solusi alternatif yang menggunakan metode pindah silang dan mutasi yang terstruktur, yang memungkinkan waktu untuk mendapat solusi alternatif relatif tetap untuk setiap iterasinya.

BAB 5
KESIMPULAN DAN SARAN

5.1. Kesimpulan

Tujuan dari penelitian ini adalah untuk mendapatkan algoritma *bee colony* yang memberikan hasil total biaya keterlambatan yang minimal dan waktu komputasi yang singkat. Hasil penelitian menunjukan bahwa algoritma *bee colony* dengan *tabu list* memberikan hasil biaya keterlambatan yang lebih baik daripada algoritma *bee colony* dan algoritma *differential evolution,* dan seimbang dengan algoritma *tabu search*. Untuk waktu komputasi ditunjukan bahwa waktu komputasi algoritma *bee colony-tabu* lebih unggul daripada algoritma *bee colony* dan algoritma *tabu search*, namun kalah cepat daripada algoritma *differential evolution.* Sedangkan untuk jumlah *job* yang sedikit sekitar 30-60 *job* waktu komputasi algoritma *bee colony-tabu* lebih unggul daripada algoritma lainnya termasuk algoritma *differential evolution*. Sehingga didapatkan kesimpulan bahwa, pencarian solusi dengan menggunakan metode *neighbourhood search* (*swap)* akan memberikan performa yang lebih unggul daripada metode pencarian solusi yang bersifat *constructed from scratch* , dan dengan memberikan *tabu list* pada algoritma *bee colony* akan memberikan efek yang positif terhadap peningkatan performa algoritma *bee colony* untuk penjadwalan *job shop* dengan kriteria biaya keterlambatan.

5.2. Saran

Berdasarkan penelitian yang telah dilakukan didapatkan beberapa saran yang mungkin berguna untuk penelitian berikutnya.

- Penelitian menggunakan berbagai data untuk mengetahui apakah kesimpulan diatas juga berlaku.
- Pengaruh penggunaan *tabu list* untuk algoritma metaheuristik lainnya yang mungkin akan memberikan hasil yang berbeda, seperti algoritma *ant colony*, *particle swarm, cukcoo search*, atau *firefly algorithm*.
- Perbandingan performa algoritma dengan menggunakan fungsi tujuan yang berbeda, seperti membandingkan performa algoritma untuk *job shop* dengan *due date* dan *job shop* dengan kendala kapasitas mesin.
- Perluasan permasalahan dengan mempertimbangkan multi fungsi tujuan

DAFTAR REFERENSI

1. Astuti, Lina. 2008. "Optimasi Penjadwalan Job Shop Dengan Metode Algoritma Differential Evolution untuk Meminimumkan Total Biaya Keterlambatan Penyelesaian Pesanan di PT X". *Skripsi*. UNIVERSITAS INDONESIA. Fakultas Teknik. Departemen Teknik Industri. Depok.
2. Bedworth. David D. dan Bailey. James E., 1982, *Integrated Production Control System* , John Wiley and Sons, New York, Hal. 311-314.
3. Cheng, Runwei & Gen, Mitsuo, 1997, *Genetic Algorithms and Engineering Design*. New York, USA : John Wiley & Sons.
4. Chong, C. S., Sivakumar, A. I., Malcolm Low, Y. H., Gay, K. L. (2006). A bee colony optimization algorithm to job shop scheduling. *Proceedings of the 38th conference on Winter simulation* . pages 1954-1961. California.
5. Chong, C. S., M. Y. H. Low, A. I. Sivakumar, and K. L. Gay. (2007), Using a bee colony Algorithm for neighborhood search in job shop scheduling problems. *Proceedings of 21st European Conference on Modeling and Simulation* (ECMS).
6. Chong, C. S., M. Y. H. Low, A. I. Sivakumar, and K. L. Gay. (2008). Bee Colony Optimization Algorithm with Big Valley Landscape Exploitation For Job shop scheduling problems. Proceedings *of the 2008 Winter simulation conference* , Pages 2050-2058
7. Engin, O, et al. (2011). A Scatter Search Method for Fuzzy Job Shop Scheduling Problem with Availability Constraints. *Proceedings of the World Congress on Engineering 2011 Vol II*. WCE 2011, July 6 - 8, 2011, London, U.K.
8. Glover,F. (1993). TABU SEARCH FUNDAMENTALS AND USE. US West Chair in Systems Science.
9. Goodman. E., Hedetniemi. S. T., 1977, *Introduction to the Design and Analysis of Algorithms*, McGraw-Hill.
10. Hillier, F.S. and Lieberman, G.J. 2005. *Introduction to Operations Research*. New York, NY: McGraw-Hill. 8th Ed
11. Karaboga, D, (2005). An idea based on honey bee swarm for numerical optimization. *Technical Report* TR06. Erciyes University, Engineering Faculty. Computer Engineering Department.
12. Karaboga, D., Akay,B. (2009). A Comparative Study of Artificial Bee Colony Algorithm. *Applied Mathematics and Computation*. 214, 108-132. Elsevier. Netherlands.
13. Karaboga D., Basturk B, (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *J Glob Optim* (2007) 39:459–471. Springer Science+Business Media B.V.
14. Karaboga D., Basturk B. (2008). On the performance of artificial bee colony (ABC) algorithm, *Applied Soft Computing* 8 , volume 2008: 687–697, ScienceDirect.
15. Morton T.E and Pentico D.W. (1993). *Heuristic Scheduling Systems*. New York: Jhon Wilet & Sons.

16. Nakrani. S. and Tovey. C., (2004), "On honey bees and dynamic server allocation in internet hosting centers," *Adaptive Behavior, vol. 12*, no. 3-4, pp. 223–240

17. Nasution, Arman H., (2003), *Perencanaan dan Pengendalian Produksi* , Edisi Pertama. Surabaya: Guna Widya.

18. Panwalkar.S. S, et al. (1981). Common Due Date Assignment to Minimize Total Penalty for The One Machine Scheduling Problem. *Operation Research.* Vol 30, No. 2, March-April 1982.

19. Pham DT, Ghanbarzadeh A, Koc E, Otri S, Rahim S and Zaidi M. (2006). *The Bees Algorithm – A Novel Tool for Complex Optimisation Problems*. Intelligent Systems Laboratory, Manufacturing Engineering Centre, Cardiff University, UK,

20. Seeley, T.D., S. Kühnholz, and A. Weidenmüller. 1996. The honey bee's tremble dance stimulates additional bees to function as nectar receivers. Behavioral Ecology and Sociobiology 39: 419-427

21. Schmidt, K. (May 18, 2001). *Using Tabu Search to Solve the Job Shop Scheduling Problem with Sequence Dependent Setup Times*.

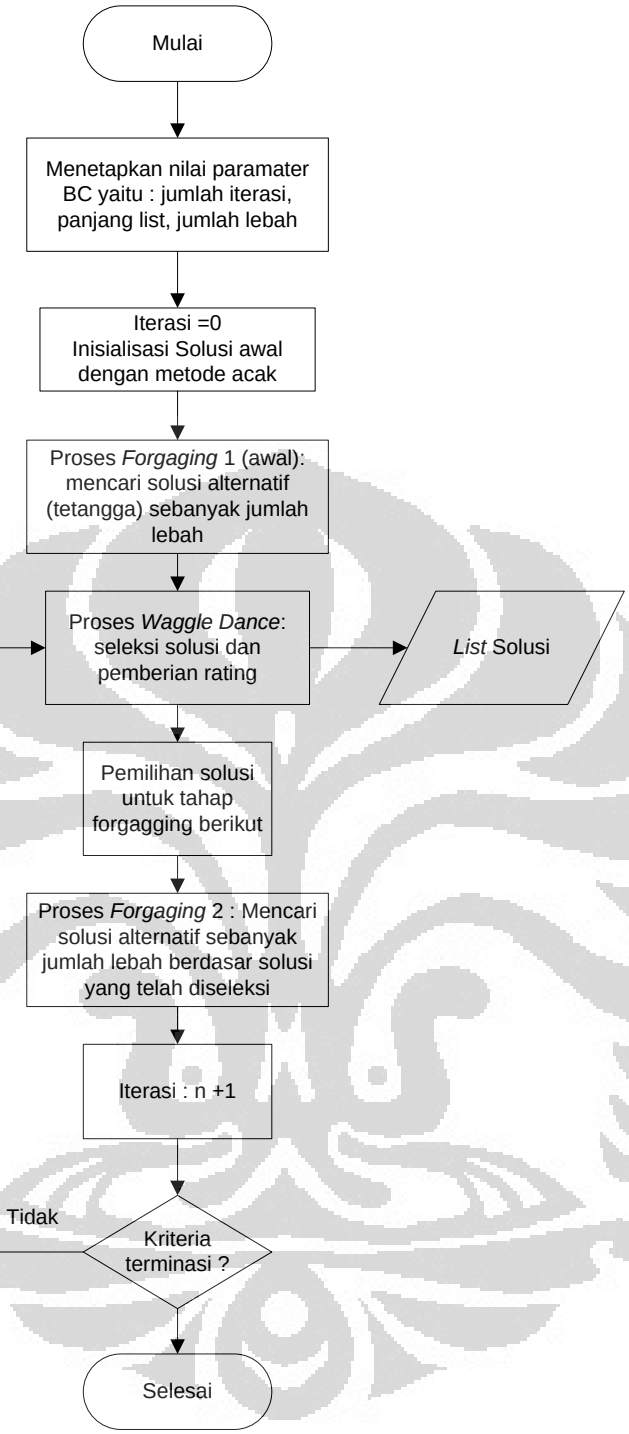
22. Teodorovic. D., (2008), "Swarm intelligence systems for transportation engineering: Principles and applications," *Transportation Research Part C: Emerging Technologies, vol. 16*, no. 6, pp. 651–657.

23. Van Laarhoven, Peter J.M, Emile, and Lenstra, JK. (1992). Job Shop Scheduling by Simulated Annealing. *Operations Research*, Vol. 40, No. 1 (Jan. - Feb., 1992), pp. 113-125

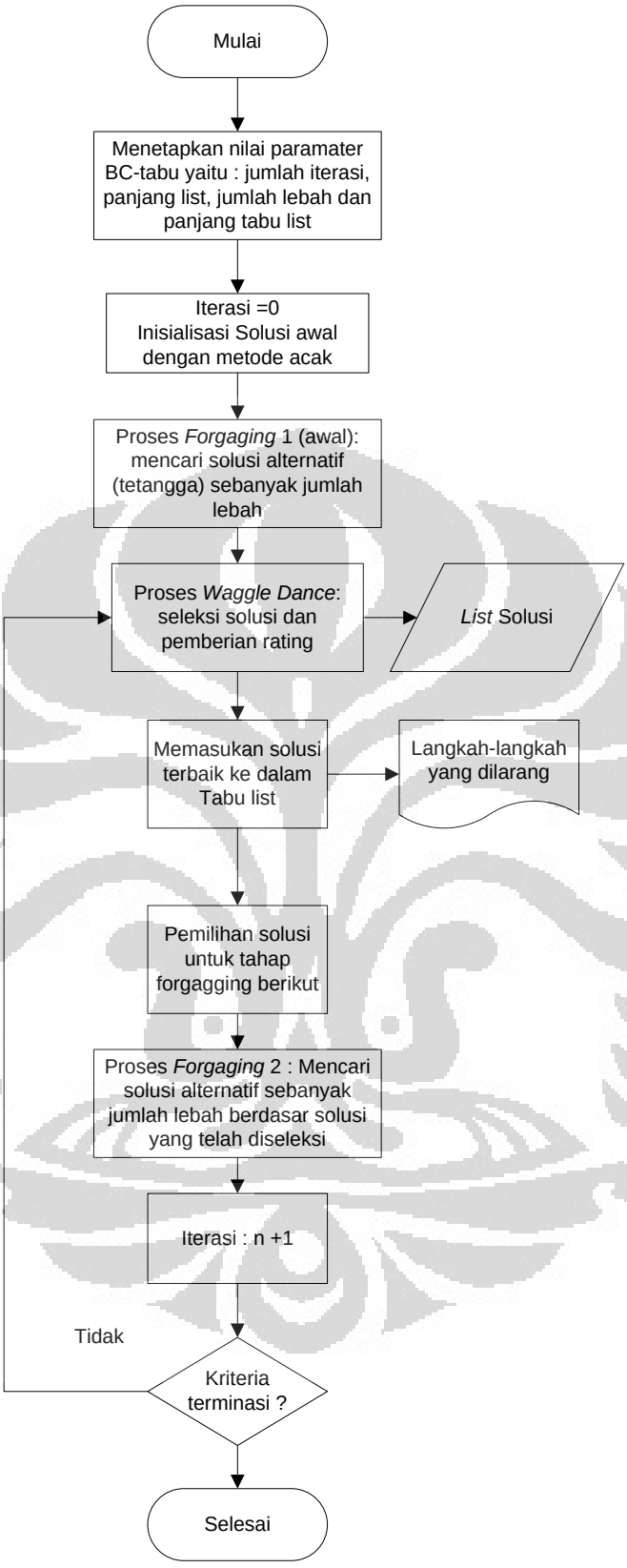
24. Von Frisch. K., (1974), "Decoding the language of the bee," *Science, vol. 185*, no. 4152, pp. 663–668.

25. Yamada, T and Nakano, R. (1997). Genetic Algorithms for Job-Shop Scheduling Problems. *Proceedings of Modern Heuristic for Decision Support*, pp. 67{81,UNICOM seminar,18-19 March 1997,London.

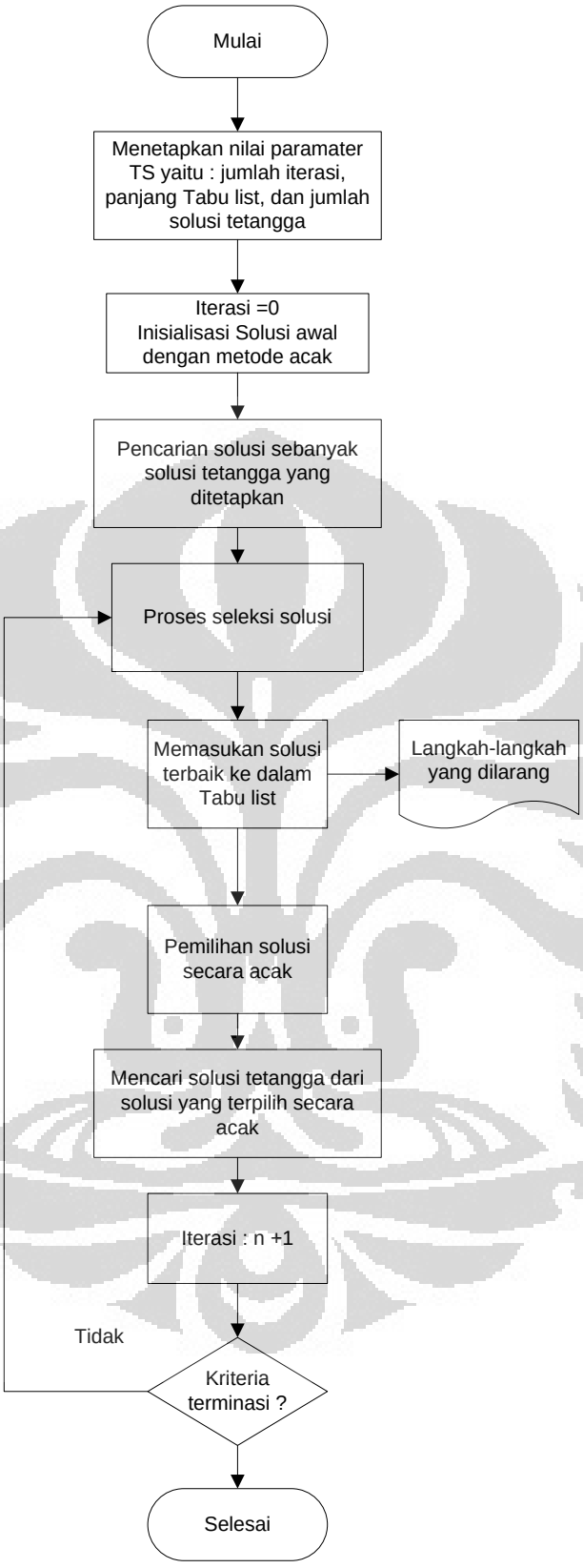
26. Zhang, R. (2011). A Differential Evolution Algorithm for Job Shop Scheduling Problems Involving Due Date Determination Decisions. *International Journal of Digital Content Technology and its Applications*. Volume 5, Number 7, July 2011



Gambar L1.1. Diagram Alir Pengerjaan Algoritma *Bee Colony*



Gambar L1.2. Diagram Alir Pengerjaan Algoritma *Bee Colony-Tabu*



Gambar L1.3. Diagram Alir Pengerjaan Algoritma *Tabu Search*

Tabel L2.1. Hasil Skenario Algoritma *Tabu Search*

Solusi Tetangga	*Tabu List*	Iterasi	Biaya Penalti 1	*Biaya Penalti 2*	*Makespan 1*	*Makespan 2*	Waktu Komputasi 1	Waktu Komputasi 2
30	7	100	35119.95	39144.15	20845	20845	3.3	3.21
30	7	1500	28377.6	28368.25	20845	20845	36.83	36.95
30	7	3000	28312.2	28368.25	20845	20845	72.77	73.15
30	13	100	39341.55	37121.75	20845	20845	3.37	3.2
30	13	1500	28415	28403.6	20845	20845	36.97	37.36
30	13	3000	28415	28366.95	20845	20845	73.27	73.538
30	90	100	35801.3	32307.65	20845	20845	3.4	3.51
30	90	1500	28471.8	28597.55	20845	20845	39.36	39.16
30	90	3000	28398.5	28395.45	20845	20845	78.67	78.34
50	7	100	31585.2	31739.6	20845	20845	4.89	4.86
50	7	1500	28433.7	28352	20845	20845	60.12	59.72
50	**7**	**3000**	**28280.05**	**28345.5**	**20845**	**20845**	**118.38**	**118.88**
50	13	100	29401.8	30319.75	20845	20845	4.87	5
50	13	1500	28345.5	28374.85	20845	20845	60.364	60.2
50	13	3000	**28280.05**	28371.05	20845	20845	118.86	119.75
50	90	100	33677.75	38152.35	20845	20845	5.1	5.1
50	90	1500	28377.6	28368.25	20845	20845	64.191	65.04
50	90	3000	28352.05	28368	20845	20845	126.7	127.7
100	7	100	28927.4	28746.3	20845	20845	8.716	8.694
100	7	1500	28377.6	28352.05	20845	20845	117.53	117.54
100	7	3000	28345.5	**28280.05**	20845	20845	232.8	233.81
100	13	100	29001	28979.45	20845	20845	8.75	8.78
100	13	1500	28433.7	28433.7	20845	20845	117.72	117.7
100	13	3000	28345.5	28352.05	20845	20845	235.9	233.8
100	90	100	30408.9	30764.05	20845	20845	9.35	9.21
100	90	1500	28377.6	28436.5	20845	20845	126.8	126.02
100	90	3000	28377.6	28345.5	20845	20845	253.5	264.3

Tabel L2.2. Hasil Skenario Algoritma *Bee Colony*

Jumlah Lebah	Panjang *List*	Jumlah Iterasi	Biaya Penalti 1	Biaya Penalti 2	*Makespan 1*	*Makespan 2*	Waktu Komputasi 1	Waktu Komputasi 2
10	5	100	145025	125859	20915	20885	1.92	1.81
10	5	1500	69213.2	68534.75	20845	20845	13.9	14.41
10	5	3000	61835.8	58104.05	20845	20845	26.97	27.74
10	20	100	157060	107343.2	20845	20845	1.76	1.79
10	20	1500	67331	62176.85	20845	20845	13.98	14.51
10	20	3000	59747	62950.2	20845	20845	27.1	27.11
50	5	100	39981.7	33569.85	20845	20845	4.83	4.95
50	5	1500	28280.5	28377.6	20845	20845	60.613	60.554
50	5	3000	28377.6	28377.6	20845	20845	119.36	119.54
50	20	100	40045.7	34313.5	20845	20845	5	4.93
50	20	1500	28405.7	28376.85	20845	20845	60.4	60.5
50	**20**	**3000**	**28268.3**	**28280.05**	**20845**	**20845**	**120.02**	**120.3**
90	5	100	30538.7	29658.4	20845	20845	8.03	7.92
90	5	1500	**28280.1**	28352.05	20845	20845	106.77	106.5
90	5	3000	28345.5	28345.5	20845	20845	212.01	211.95
90	20	100	29395.1	29922.7	20845	20845	8	7.99
90	20	1500	28280.1	28377.6	20845	20845	107.2198	106.73
90	20	3000	28345.5	28377.6	20845	20845	212.21	212.83

Tabel L2.3. Hasil Skenario Algoritma *Bee Colony-Tabu List*

Jumlah Lebah	Panjang *List*	*Tabu List*	Jumlah Iterasi	Biaya Penalti 1	Biaya Penalti 2	*Makespan 1*	*Makespan 2*	Waktu Komputasi 1	Waktu Komputasi 2
50	7	7	1500	28398.25	28377.6	20845	20845	61.1	60.8
50	7	7	3000	28371.45	**28280.1**	20845	20845	120.7	120.115
50	7	13	1500	28301.65	28368.3	20845	20845	60.97	60.81
50	7	13	3000	28368.25	28368.3	20845	20845	121.49	120.64
50	13	7	1500	28371.05	28352.1	20845	20845	60.88	60.9
50	13	7	3000	28374.85	28268.3	20845	20845	121.1	120.7
50	13	13	1500	28368.25	28265.5	20845	20845	61.21	61.2
50	13	13	3000	28302.85	28368.3	20845	20845	121.4	121.1
90	7	7	1500	**28280.05**	28433.7	20845	20845	107.94	107.34
90	7	7	3000	28345.5	28352.1	20845	20845	214.34	214.3
90	7	13	1500	28352.05	28345.5	20845	20845	107.9	108.8
90	7	13	3000	**28280.05**	28352.1	20845	20845	215.1	215.7
90	13	7	1500	28377.6	28377.6	20845	20845	107.4	108.13
90	13	7	3000	28345.5	28345.5	20845	20845	213.7	214.21
90	**13**	**13**	**1500**	**28280.05**	**28302.9**	**20845**	**20845**	**107.91**	**108.43**
90	13	13	3000	28352.05	28352.1	20845	20845	215.03	216.22

Hasil ANOVA untuk Eksperimen Algoritma Tabu Search

Hasil ANOVA untuk Algoritma Tabu Search dengan Respon Biaya Keterlambatan

Analysis of Variance for Biaya, using Adjusted SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Tetangga	2	50460110	50460110	25230055	25.03	0.000
Tabu-L	**2**	**1778883**	**1778883**	**889442**	**0.88**	**0.425**
Iterasi	2	235537372	235537372	117768686	116.85	0.000
Tetangga*Tabu-L	4	18393132	18393132	4598283	4.56	0.006
Tetangga*Iterasi	4	97593653	97593653	24398413	24.21	0.000
Tabu-L*Iterasi	**4**	**2889271**	**2889271**	**722318**	**0.72**	**0.588**
Tetangga*Tabu-L*Iterasi	8	38147086	38147086	4768386	4.73	0.001
Error	27	27213305	27213305	1007900		
Total	53	472012813				

Hasil ANOVA untuk Algoritma Tabu Search dengan Respon Makespan

Analysis of Variance for Makespan, using Adjusted SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Tetangga	2	0.0000000	0.0000000	0.0000000	**	
Tabu-L	2	0.0000000	0.0000000	0.0000000	**	
Iterasi	2	0.0000000	0.0000000	0.0000000	**	
Tetangga*Tabu-L	4	0.0000000	0.0000000	0.0000000	**	
Tetangga*Iterasi	4	0.0000000	0.0000000	0.0000000	**	
Tabu-L*Iterasi	4	0.0000000	0.0000000	0.0000000	**	
Tetangga*Tabu-L*Iterasi	8	0.0000000	0.0000000	0.0000000	**	
Error	27	0.0000000	0.0000000	0.0000000		
Total	53	0.0000000				

Hasil ANOVA untuk Algoritma Tabu Search dengan Respon Waktu Komputasi

Analysis of Variance for Waktu, using Adjusted SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Tetangga	2	69543	69543	34771	14872.75	0.000
Tabu-L	2	447	447	223	95.59	0.000
Iterasi	2	178006	178006	89003	38069.08	0.000
Tetangga*Tabu-L	4	174	174	44	18.63	0.000
Tetangga*Iterasi	4	41869	41869	10467	4477.13	0.000
Tabu-L*Iterasi	4	318	318	79	33.99	0.000
Tetangga*Tabu-L*Iterasi	8	151	151	19	8.08	0.000
Error	27	63	63	2		
Total	53	290571				

Hasil ANOVA untuk Eksperimen Algoritma Bee Colony

Hasil ANOVA untuk Algoritma Bee Colony dengan Respon Biaya Keterlambatan

Analysis of Variance for Biaya, using Adjusted SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Lebah	2	26082221260	26082221260	13041110630	158.32	0.000
LIst	**1**	**4039966**	**4039966**	**4039966**	**0.05**	**0.827**
Iterasi	2	5751293561	5751293561	2875646781	34.91	0.000
Lebah*LIst	**2**	**7987037**	**7987037**	**3993518**	**0.05**	**0.953**
Lebah*Iterasi	4	7628130159	7628130159	1907032540	23.15	0.000
LIst*Iterasi	**2**	**5547626**	**5547626**	**2773813**	**0.03**	**0.967**
Lebah*LIst*Iterasi	**4**	**12171998**	**12171998**	**3042999**	**0.04**	**0.997**
Error	18	1482690078	1482690078	82371671		
Total	35	40974081686				

Hasil ANOVA untuk Algoritma Bee Colony dengan Respon Makespan

Analysis of Variance for MAkespan, using Adjusted SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Lebah	2	672.22	672.22	336.11	13.44	0.000
LIst	1	336.11	336.11	336.11	13.44	0.002
Iterasi	2	672.22	672.22	336.11	13.44	0.000
Lebah*LIst	2	672.22	672.22	336.11	13.44	0.000
Lebah*Iterasi	4	1344.44	1344.44	336.11	13.44	0.000
LIst*Iterasi	2	672.22	672.22	336.11	13.44	0.000
Lebah*LIst*Iterasi	4	1344.44	1344.44	336.11	13.44	0.000
Error	18	450.00	450.00	25.00		
Total	35	6163.89				

Hasil ANOVA untuk Algoritma Bee Colony dengan Respon Waktu Komputasi

Analysis of Variance for Waktu, using Adjusted SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Lebah	2	53691.2	53691.2	26845.6	482355.87	0.000
LIst	1	0.2	0.2	0.2	3.38	0.083
Iterasi	2	79171.0	79171.0	39585.5	711264.46	0.000
Lebah*LIst	2	0.2	0.2	0.1	2.19	0.141
Lebah*Iterasi	4	32001.0	32001.0	8000.3	143746.90	0.000
LIst*Iterasi	2	0.2	0.2	0.1	1.57	0.236
Lebah*LIst*Iterasi	4	0.4	0.4	0.1	1.83	0.167
Error	18	1.0	1.0	0.1		
Total	35	164865.2				

Hasil ANOVA untuk Eksperimen Algoritma Bee Colony-Tabu

Hasil ANOVA untuk Algoritma Bee Colony-Tabu dengan Respon Biaya Keterlambatan

Analysis of Variance for Biaya, using Adjusted SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Lebah	1	29	29	29	0.01	0.909
LIst	1	908	908	908	0.42	0.526
Tabu-L	1	3263	3263	3263	1.51	0.237
Iterasi	1	489	489	489	0.23	0.641
Lebah*LIst	1	750	750	750	0.35	0.564
Lebah*Tabu-L	1	786	786	786	0.36	0.555
Lebah*Iterasi	1	179	179	179	0.08	0.777
LIst*Tabu-L	1	435	435	435	0.20	0.660
LIst*Iterasi	1	739	739	739	0.34	0.567
Tabu-L*Iterasi	1	6173	6173	6173	2.86	0.110
Lebah*LIst*Tabu-L	1	43	43	43	0.02	0.889
Lebah*LIst*Iterasi	1	477	477	477	0.22	0.645
Lebah*Tabu-L*Iterasi	1	926	926	926	0.43	0.522
LIst*Tabu-L*Iterasi	1	811	811	811	0.37	0.549
Lebah*LIst*Tabu-L*Iterasi	1	2969	2969	2969	1.37	0.258
Error	16	34587	34587	2162		
Total	31	53564				

Hasil ANOVA untuk Algoritma Bee Colony-Tabu dengan Respon Makespan

Analysis of Variance for Makespan, using Adjusted SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Lebah	1	0.0000000	0.0000000	0.0000000	**	
LIst	1	0.0000000	0.0000000	0.0000000	**	
Tabu-L	1	0.0000000	0.0000000	0.0000000	**	
Iterasi	1	0.0000000	0.0000000	0.0000000	**	
Lebah*LIst	1	0.0000000	0.0000000	0.0000000	**	
Lebah*Tabu-L	1	0.0000000	0.0000000	0.0000000	**	
Lebah*Iterasi	1	0.0000000	0.0000000	0.0000000	**	
LIst*Tabu-L	1	0.0000000	0.0000000	0.0000000	**	
LIst*Iterasi	1	0.0000000	0.0000000	0.0000000	**	
Tabu-L*Iterasi	1	0.0000000	0.0000000	0.0000000	**	
Lebah*LIst*Tabu-L	1	0.0000000	0.0000000	0.0000000	**	
Lebah*LIst*Iterasi	1	0.0000000	0.0000000	0.0000000	**	
Lebah*Tabu-L*Iterasi	1	0.0000000	0.0000000	0.0000000	**	
LIst*Tabu-L*Iterasi	1	0.0000000	0.0000000	0.0000000	**	
Lebah*LIst*Tabu-L*Iterasi	1	0.0000000	0.0000000	0.0000000	**	
Error	16	0.0000000	0.0000000	0.0000000		
Total	31	0.0000000				

Hasil ANOVA untuk Algoritma Bee Colony-Tabu dengan Waktu Komputasi

Analysis of Variance for Waktu, using Adjusted SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Lebah	1	39715.1	39715.1	39715.1	233535.78	0.000
LIst	1	0.1	0.1	0.1	0.40	0.536
Tabu-L	1	3.3	3.3	3.3	19.32	0.000
Iterasi	1	55621.5	55621.5	55621.5	327069.84	0.000
Lebah*LIst	1	0.2	0.2	0.2	0.93	0.348
Lebah*Tabu-L	1	0.8	0.8	0.8	4.98	0.040
Lebah*Iterasi	1	4403.3	4403.3	4403.3	25892.75	0.000
LIst*Tabu-L	1	0.0	0.0	0.0	0.09	0.766
LIst*Iterasi	1	0.0	0.0	0.0	0.08	0.776
Tabu-L*Iterasi	1	0.7	0.7	0.7	4.19	0.057
Lebah*LIst*Tabu-L	1	0.0	0.0	0.0	0.03	0.854
Lebah*LIst*Iterasi	1	0.0	0.0	0.0	0.19	0.669
Lebah*Tabu-L*Iterasi	1	0.1	0.1	0.1	0.57	0.460
LIst*Tabu-L*Iterasi	1	0.0	0.0	0.0	0.03	0.858
Lebah*LIst*Tabu-L*Iterasi	1	0.3	0.3	0.3	1.83	0.195
Error	16	2.7	2.7	0.2		
Total	31	99748.3				

Kesimpulan

Algoritma *Tabu Search*

Pada hasil analisa regresi didapatkan bahwa, panjang *tabu list* dan interaksi antara panjang *tabu list* dan jumlah iterasi mempengaruhi hasil biaya keterlambatan. Namun untuk waktu komputasi dan *makespan* tidak ada variable yang memberikan pengaruh yang signifikan.

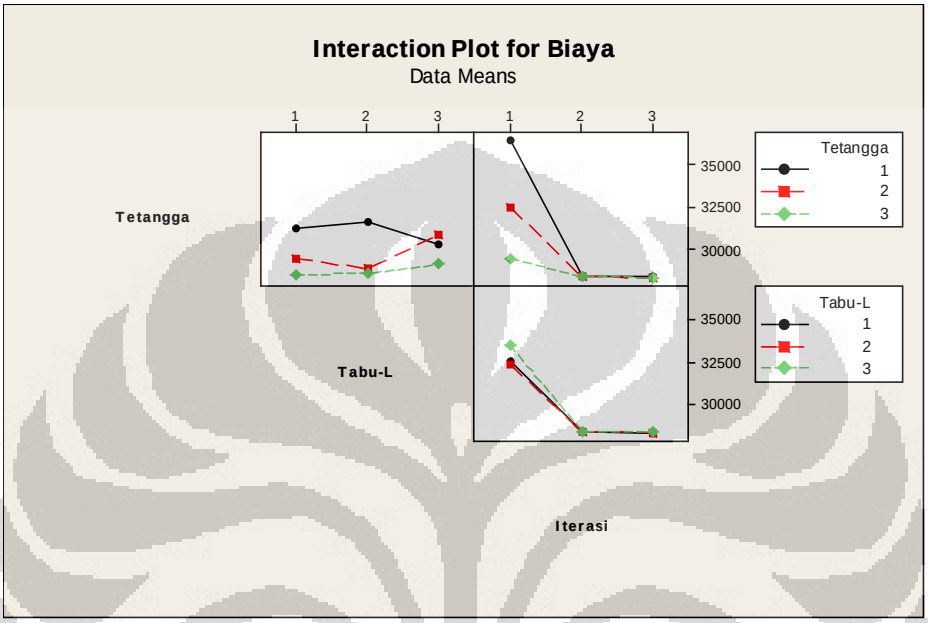
Algoritma *Bee Colony*

Hasil regresi menunjukan bahwa untuk mendapatkan panjang *list* solusi, interaksi antara panjang *list*-jumlah lebah, panjang *list*-jumlah iterasi, dan panjang *list*-jumlah lebah-jumlah iterasi akan memberikan pengaruh yang cukup signifikan terhadap fungsi tujuan (biaya keterlambatan) dan waktu komputasi.

Algoritma *Bee Colony-Tabu List*

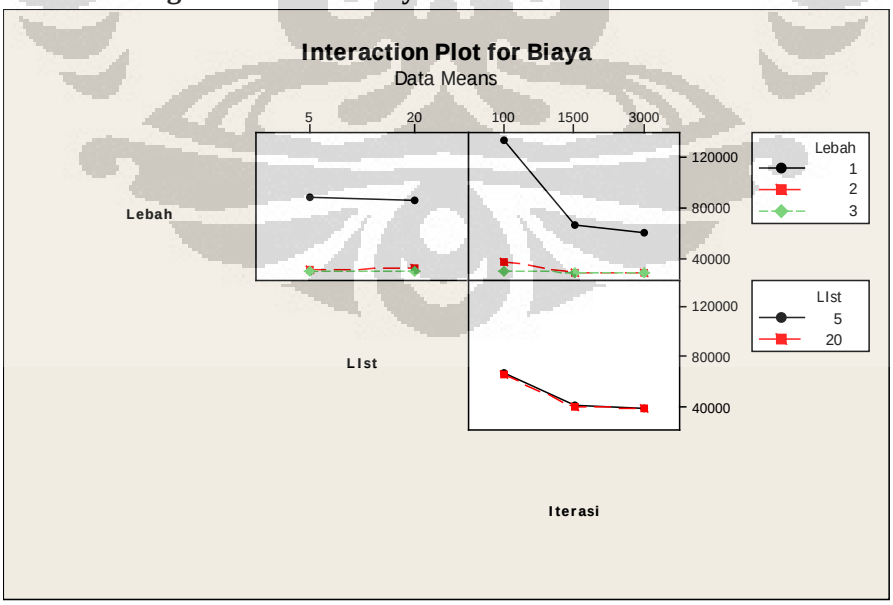
Hasil regresi menunjukan bahwa seluruh variable berserta interaksinya, memiliki pengaruh yang siginifikan terhadap fungsi tujuan (biaya keterlambatan), dan juga untuk waktu komputasi (kecuali : jumlah lebah, jumlah iterasi, panjang *tabu list*, jumlah lebah- panjang *tabu list*, jumlah lebah- jumlah iterasi, dan , jumlah iterasi- panjang *tabu list*), hal ini menunjukan modifikasi dengan menggunakan *tabu list* memberikan peningkatan performa yang siginifikan pada algoritma *bee colony*.

Interaction Plot Untuk Algoritma Tabu Search



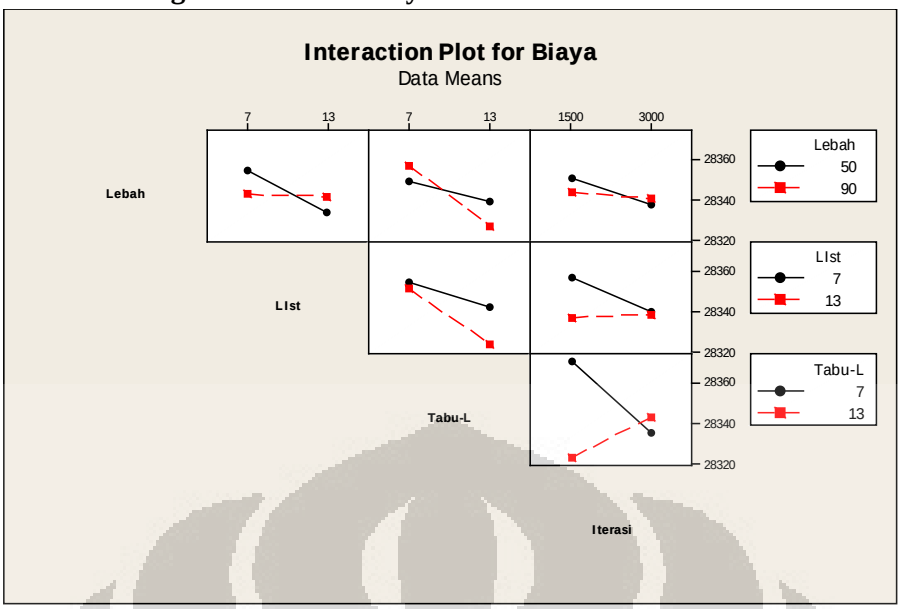
Tetangga Level 2 : 50; Tabu list Level 2: 7; Dan Jumlah Iterasi Level 2 : 1500
Akan memberikan perfroma yang lebih cepat stabil untuk respon biaya keterlambatan.

Interaction Plot Untuk Algoritma Bee Colony



Pada Jumlah lebah : 90; dan jumlah iterasi : 3000, akan menghasilkan hasil mendekati optimum lebih cepat dan stabil.

Interaction Plot Untuk Algoritma Bee Colony-Tabu



Pada jumlah lebah : 90; jumlah iterasi : 1500; jumlah list solusi : 13; dan tabu list : 13, memberikan hasil yang lebih mendekati optimal.

Tabel L3.1 Data Percobaan Untuk Waktu Pekerjaan yang Ditingkatkan

	A	B	C	D1	D2	E1	E2	F	G	Due Date	Bobot
1	150	140	15	0	240	0	300	180	180	1760	187
2	150	140	15	0	240	0	300	180	180	1760	187
3	150	140	15	0	240	0	300	180	180	1760	187
4	150	140	15	0	240	0	300	180	180	1760	187
5	150	140	15	0	240	0	300	180	180	2640	187
6	150	140	15	0	240	0	300	180	180	2640	187
7	150	140	15	0	240	0	300	180	180	3520	187
8	150	140	15	0	240	0	300	180	180	3520	187
9	150	140	15	0	240	0	300	180	180	3520	187
10	150	140	15	0	240	0	300	180	180	3520	187
11	150	140	15	0	240	0	300	180	180	3520	195
12	150	140	15	0	240	0	300	180	180	3520	195
13	60	140	15	210	0	240	0	150	155	4400	114
14	60	140	15	210	0	240	0	150	155	4400	114
15	60	140	15	210	0	240	0	150	155	4400	114
16	120	140	15	0	210	0	300	160	180	4400	195
17	60	140	15	210	0	240	0	150	155	5280	114
18	60	140	15	210	0	240	0	150	155	5280	114
19	60	140	15	210	0	240	0	150	155	5280	114
20	60	140	15	210	0	240	0	150	155	5280	114
21	60	140	15	210	0	240	0	150	155	5280	114
22	60	140	15	210	0	240	0	150	155	5280	114
23	60	140	15	210	0	240	0	150	155	6160	114
24	60	140	15	210	0	240	0	150	155	6160	114
25	60	140	15	210	0	240	0	150	155	6160	114
26	60	140	15	210	0	240	0	150	155	6160	114
27	60	140	15	210	0	240	0	150	155	7040	114
28	60	140	15	210	0	240	0	150	155	7040	114
29	120	140	15	0	210	0	300	160	180	7920	195
30	120	140	15	0	210	0	300	160	180	7920	195
31	120	140	15	0	210	0	300	160	180	7920	204
32	120	150	15	0	240	0	340	150	200	7920	239
33	120	140	15	0	210	0	300	160	180	8800	204
34	120	140	15	0	210	0	300	160	180	8800	204
35	120	140	15	0	210	0	300	160	180	8800	204
36	120	150	15	0	240	0	340	150	200	8800	239
37	50	140	15	150	0	120	0	90	155	9680	100
38	120	140	15	0	210	0	300	160	180	9680	204
39	120	140	15	0	210	0	300	160	180	9680	204
40	120	140	15	0	210	0	300	160	180	9680	204
41	50	140	15	150	0	120	0	90	155	10560	100
42	50	140	15	150	0	120	0	90	155	10560	100
43	50	140	15	150	0	120	0	90	155	10560	100
44	50	140	15	150	0	120	0	90	155	11440	100
45	120	150	15	0	240	0	340	150	200	11440	239

Tabel L3.1 Data Percobaan Untuk Waktu Pekerjaan yang Ditingkatkan (lanjutan)

	A	B	C	D1	D2	E1	E2	F	G	Due Date	Bobot
46	120	150	15	0	240	0	340	150	200	11440	239
47	120	150	15	0	240	0	340	150	200	11440	239
48	120	150	15	0	240	0	340	150	200	11440	239
49	120	140	15	0	210	0	300	160	180	12320	187
50	120	140	15	0	210	0	300	160	180	12320	187
51	120	150	15	0	240	0	340	150	200	12320	239
52	120	150	15	0	240	0	340	150	200	12320	239
53	120	150	15	0	240	0	340	150	200	12320	239
54	120	150	15	0	240	0	340	150	200	12320	239
55	120	140	15	0	210	0	300	160	180	13200	187
56	120	140	15	0	210	0	300	160	180	13200	187
57	120	140	15	0	210	0	300	160	180	13200	187
58	120	140	15	0	210	0	300	160	180	13200	187
59	120	140	15	0	210	0	300	160	180	13200	187
60	50	140	15	150	0	120	0	90	155	14080	100
61	120	140	15	0	210	0	300	160	180	14080	187
62	120	140	15	0	210	0	300	160	180	14080	187
63	50	140	15	150	0	120	0	90	155	14960	100
64	50	140	15	150	0	120	0	90	155	14960	100
65	50	140	15	150	0	120	0	90	155	15840	100
66	50	140	15	150	0	120	0	90	155	15840	100
67	200	140	15	0	210	0	300	160	180	15840	204
68	200	150	15	0	240	0	340	150	200	15840	239
69	200	150	15	0	240	0	340	150	200	15840	239
70	200	140	15	0	210	0	300	160	180	16720	204
71	200	140	15	0	210	0	300	160	180	16720	204
72	200	150	15	0	240	0	340	150	200	16720	239
73	200	140	15	0	210	0	300	160	180	17600	195
74	200	140	15	0	210	0	300	160	180	17600	195
75	200	140	15	0	210	0	300	160	180	17600	195
76	200	140	15	0	210	0	300	160	180	17600	195
77	200	140	15	0	210	0	300	160	180	17600	204
78	200	140	15	0	210	0	300	160	180	17600	204
79	200	140	15	0	210	0	300	160	180	17600	204
80	200	140	15	0	210	0	300	160	180	17600	204
81	200	140	15	0	210	0	300	160	180	17600	204
82	200	140	15	0	210	0	300	160	180	17600	204
83	200	140	15	0	210	0	300	160	180	18480	187
84	200	140	15	0	210	0	300	160	180	18480	187
85	200	140	15	0	210	0	300	160	180	18480	187
86	200	140	15	0	210	0	300	160	180	18480	204
87	200	140	15	0	210	0	300	160	180	18480	204
88	200	140	15	0	210	0	300	160	180	18480	204
89	200	140	15	0	210	0	300	160	180	18480	204
90	200	140	15	0	210	0	300	160	180	18480	204

Tabel L3.2 Data Percobaan Untuk Waktu Pekerjaan yang Diturunkan

	A	B	C	D1	D2	E1	E2	F	G	Due Date	Bobot
1	100	140	15	0	200	0	300	120	180	1760	187
2	100	140	15	0	200	0	300	120	180	1760	187
3	100	140	15	0	200	0	300	120	180	1760	187
4	100	140	15	0	200	0	300	120	180	1760	187
5	100	140	15	0	200	0	300	120	180	2640	187
6	100	140	15	0	200	0	300	120	180	2640	187
7	100	140	15	0	200	0	300	120	180	3520	187
8	100	140	15	0	200	0	300	120	180	3520	187
9	100	140	15	0	200	0	300	120	180	3520	187
10	100	140	15	0	200	0	300	120	180	3520	187
11	100	140	15	0	200	0	300	120	180	3520	195
12	100	140	15	0	200	0	300	120	180	3520	195
13	60	140	15	210	0	240	0	150	155	4400	114
14	60	140	15	210	0	240	0	150	155	4400	114
15	60	140	15	210	0	240	0	150	155	4400	114
16	120	140	15	0	210	0	300	160	180	4400	195
17	60	140	15	210	0	240	0	150	155	5280	114
18	60	140	15	210	0	240	0	150	155	5280	114
19	60	140	15	210	0	240	0	150	155	5280	114
20	60	140	15	210	0	240	0	150	155	5280	114
21	60	140	15	210	0	240	0	150	155	5280	114
22	60	140	15	210	0	240	0	150	155	5280	114
23	60	140	15	210	0	240	0	150	155	6160	114
24	60	140	15	210	0	240	0	150	155	6160	114
25	60	140	15	210	0	240	0	150	155	6160	114
26	60	140	15	210	0	240	0	150	155	6160	114
27	60	140	15	210	0	240	0	150	155	7040	114
28	60	140	15	210	0	240	0	150	155	7040	114
29	120	140	15	0	210	0	300	160	180	7920	195
30	120	140	15	0	210	0	300	160	180	7920	195
31	120	140	15	0	210	0	300	160	180	7920	204
32	120	150	15	0	240	0	340	150	200	7920	239
33	120	140	15	0	210	0	300	160	180	8800	204
34	120	140	15	0	210	0	300	160	180	8800	204
35	120	140	15	0	210	0	300	160	180	8800	204
36	120	150	15	0	240	0	340	150	200	8800	239
37	50	140	15	150	0	120	0	90	155	9680	100
38	120	140	15	0	210	0	300	160	180	9680	204
39	120	140	15	0	210	0	300	160	180	9680	204
40	120	140	15	0	210	0	300	160	180	9680	204
41	50	140	15	150	0	120	0	90	155	10560	100
42	50	140	15	150	0	120	0	90	155	10560	100
43	50	140	15	150	0	120	0	90	155	10560	100
44	50	140	15	150	0	120	0	90	155	11440	100
45	120	150	15	0	240	0	340	150	200	11440	239

Tabel L3.2 Data Percobaan Untuk Waktu Pekerjaan yang Diturunkan (lanjutan)

	A	B	C	D1	D2	E1	E2	F	G	Due Date	Bobot
46	120	150	15	0	240	0	340	150	200	11440	239
47	120	150	15	0	240	0	340	150	200	11440	239
48	120	150	15	0	240	0	340	150	200	11440	239
49	120	140	15	0	210	0	300	160	180	12320	187
50	120	140	15	0	210	0	300	160	180	12320	187
51	120	150	15	0	240	0	340	150	200	12320	239
52	120	150	15	0	240	0	340	150	200	12320	239
53	120	150	15	0	240	0	340	150	200	12320	239
54	120	150	15	0	240	0	340	150	200	12320	239
55	120	140	15	0	210	0	300	160	180	13200	187
56	120	140	15	0	210	0	300	160	180	13200	187
57	120	140	15	0	210	0	300	160	180	13200	187
58	120	140	15	0	210	0	300	160	180	13200	187
59	120	140	15	0	210	0	300	160	180	13200	187
60	50	140	15	150	0	120	0	90	155	14080	100
61	120	140	15	0	210	0	300	160	180	14080	187
62	120	140	15	0	210	0	300	160	180	14080	187
63	50	140	15	150	0	120	0	90	155	14960	100
64	50	140	15	150	0	120	0	90	155	14960	100
65	50	140	15	150	0	120	0	90	155	15840	100
66	50	140	15	150	0	120	0	90	155	15840	100
67	100	140	15	0	210	0	300	160	180	15840	204
68	100	150	15	0	240	0	340	150	200	15840	239
69	100	150	15	0	240	0	340	150	200	15840	239
70	100	140	15	0	210	0	300	160	180	16720	204
71	100	140	15	0	210	0	300	160	180	16720	204
72	100	150	15	0	240	0	340	150	200	16720	239
73	100	140	15	0	210	0	300	160	180	17600	195
74	100	140	15	0	210	0	300	160	180	17600	195
75	100	140	15	0	210	0	300	160	180	17600	195
76	100	140	15	0	210	0	300	160	180	17600	195
77	100	140	15	0	210	0	300	160	180	17600	204
78	100	140	15	0	210	0	300	160	180	17600	204
79	100	140	15	0	210	0	300	160	180	17600	204
80	100	140	15	0	210	0	300	160	180	17600	204
81	100	140	15	0	210	0	300	160	180	17600	204
82	100	140	15	0	210	0	300	160	180	17600	204
83	100	140	15	0	210	0	300	160	180	18480	187
84	100	140	15	0	210	0	300	160	180	18480	187
85	100	140	15	0	210	0	300	160	180	18480	187
86	100	140	15	0	210	0	300	160	180	18480	204
87	100	140	15	0	210	0	300	160	180	18480	204
88	100	140	15	0	210	0	300	160	180	18480	204
89	100	140	15	0	210	0	300	160	180	18480	204
90	100	140	15	0	210	0	300	160	180	18480	204

Tabel L3.3 Data Percobaan Untuk Jumlah *Job* 60

	A	B	C	D1	D2	E1	E2	F	G	Due Date	Bobot
1	120	140	15	0	210	0	300	160	180	1760	187
2	120	140	15	0	210	0	300	160	180	1760	187
3	120	140	15	0	210	0	300	160	180	1760	187
4	120	140	15	0	210	0	300	160	180	2640	187
5	120	140	15	0	210	0	300	160	180	3520	187
6	120	140	15	0	210	0	300	160	180	3520	187
7	120	140	15	0	210	0	300	160	180	3520	187
8	120	140	15	0	210	0	300	160	180	3520	195
9	60	140	15	210	0	240	0	150	155	4400	114
10	60	140	15	210	0	240	0	150	155	4400	114
11	120	140	15	0	210	0	300	160	180	4400	195
12	60	140	15	210	0	240	0	150	155	5280	114
13	60	140	15	210	0	240	0	150	155	5280	114
14	60	140	15	210	0	240	0	150	155	5280	114
15	60	140	15	210	0	240	0	150	155	5280	114
16	60	140	15	210	0	240	0	150	155	6160	114
17	60	140	15	210	0	240	0	150	155	6160	114
18	60	140	15	210	0	240	0	150	155	6160	114
19	60	140	15	210	0	240	0	150	155	7040	114
20	120	140	15	0	210	0	300	160	180	7920	195
21	120	140	15	0	210	0	300	160	180	7920	204
22	120	150	15	0	240	0	340	150	200	7920	239
23	120	140	15	0	210	0	300	160	180	8800	204
24	120	140	15	0	210	0	300	160	180	8800	204
25	50	140	15	150	0	120	0	90	155	9680	100
26	120	140	15	0	210	0	300	160	180	9680	204
27	120	140	15	0	210	0	300	160	180	9680	204
28	50	140	15	150	0	120	0	90	155	10560	100
29	50	140	15	150	0	120	0	90	155	10560	100
30	50	140	15	150	0	120	0	90	155	11440	100

	A	B	C	D1	D2	E1	E2	F	G	Due Date	Bobot
31	120	150	15	0	240	0	340	150	200	11440	239
32	120	150	15	0	240	0	340	150	200	11440	239
33	120	140	15	0	210	0	300	160	180	12320	187
34	120	140	15	0	210	0	300	160	180	12320	187
35	120	150	15	0	240	0	340	150	200	12320	239
36	120	150	15	0	240	0	340	150	200	12320	239
37	120	140	15	0	210	0	300	160	180	13200	187
38	120	140	15	0	210	0	300	160	180	13200	187
39	120	140	15	0	210	0	300	160	180	13200	187
40	120	140	15	0	210	0	300	160	180	13200	187
41	120	140	15	0	210	0	300	160	180	14080	187
42	120	140	15	0	210	0	300	160	180	14080	187
43	50	140	15	150	0	120	0	90	155	14960	100
44	50	140	15	150	0	120	0	90	155	15840	100
45	120	140	15	0	210	0	300	160	180	15840	204
46	120	150	15	0	240	0	340	150	200	15840	239
47	120	140	15	0	210	0	300	160	180	16720	204
48	120	140	15	0	210	0	300	160	180	16720	204
49	120	140	15	0	210	0	300	160	180	17600	195
50	120	140	15	0	210	0	300	160	180	17600	195
51	120	140	15	0	210	0	300	160	180	17600	195
52	120	140	15	0	210	0	300	160	180	17600	204
53	120	140	15	0	210	0	300	160	180	17600	204
54	120	140	15	0	210	0	300	160	180	17600	204
55	120	140	15	0	210	0	300	160	180	17600	204
56	120	140	15	0	210	0	300	160	180	18480	187
57	120	140	15	0	210	0	300	160	180	18480	187
58	120	140	15	0	210	0	300	160	180	18480	204
59	120	140	15	0	210	0	300	160	180	18480	204
60	120	140	15	0	210	0	300	160	180	18480	204

Tabel L3.4 Data Percobaan Untuk Jumlah *Job* 30

	A	B	C	D1	D2	E1	E2	F	G	Due Date	Bobot
1	120	140	15	0	210	0	300	160	180	1760	187
2	120	140	15	0	210	0	300	160	180	1760	187
3	120	140	15	0	210	0	300	160	180	3520	187
4	120	140	15	0	210	0	300	160	180	3520	187
5	60	140	15	210	0	240	0	150	155	4400	114
6	120	140	15	0	210	0	300	160	180	4400	195
7	60	140	15	210	0	240	0	150	155	5280	114
8	60	140	15	210	0	240	0	150	155	5280	114
9	60	140	15	210	0	240	0	150	155	6160	114
10	60	140	15	210	0	240	0	150	155	7040	114
11	120	140	15	0	210	0	300	160	180	7920	204
12	120	140	15	0	210	0	300	160	180	8800	204
13	50	140	15	150	0	120	0	90	155	9680	100
14	120	140	15	0	210	0	300	160	180	9680	204
15	50	140	15	150	0	120	0	90	155	10560	100
16	120	150	15	0	240	0	340	150	200	11440	239
17	120	140	15	0	210	0	300	160	180	12320	187
18	120	150	15	0	240	0	340	150	200	12320	239
19	120	140	15	0	210	0	300	160	180	13200	187
20	120	140	15	0	210	0	300	160	180	13200	187
21	120	140	15	0	210	0	300	160	180	14080	187
22	50	140	15	150	0	120	0	90	155	14960	100
23	120	140	15	0	210	0	300	160	180	15840	204
24	120	140	15	0	210	0	300	160	180	16720	204
25	120	140	15	0	210	0	300	160	180	17600	195
26	120	140	15	0	210	0	300	160	180	17600	195
27	120	140	15	0	210	0	300	160	180	17600	204
28	120	140	15	0	210	0	300	160	180	17600	204
29	120	140	15	0	210	0	300	160	180	18480	187
30	120	140	15	0	210	0	300	160	180	18480	204

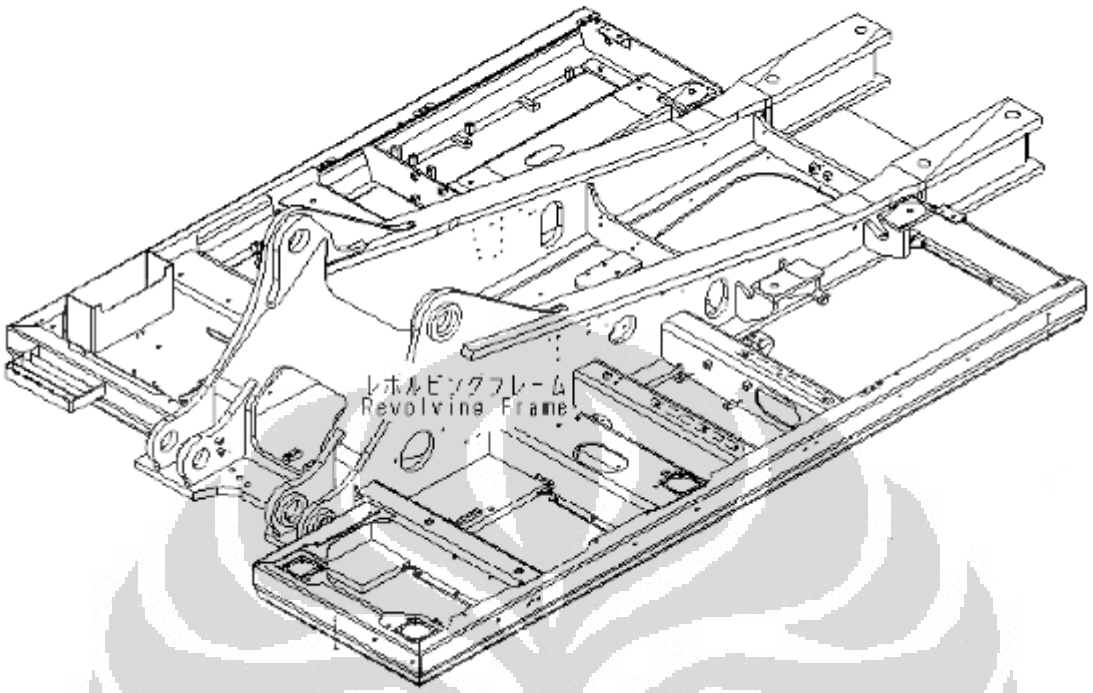
```
%Bee Colony untuk kasus Job Shop dengan tabu list
%untuk merekam waktu komputasi yang dibutuhkan
clc
clear all;
tic;
%input data
% baca_data=xlsread('Data_percobaan.xlsx');
% penalti=baca_data(:,11);

baca_data=xlsread('Data.xlsx');
penalti=baca_data(:,11)/100;
waktu_proses=baca_data(:,1:9);
due_date=baca_data(:,10);
JumlahJob = length(waktu_proses(:,1));

% tabu list
PanjangTabu = 10;
TabuList = ones(PanjangTabu, 3);

% Parameter Populasi lebah dan jumlah lebah
JumlahPopulasiLebah = 1000;
JumlahLebah = 10;
ListTerbaik = zeros(5,JumlahJob);

%total penalti diset paling besar
pnalti=99999999999999;

%Generate initial solution
SolusiAwal = randperm(JumlahJob);
[DbTime WaktuSelesaiPerJob WaktuSelesaiSolusiAwal DaftarKeterlambatan
DaftarPenalti TotalPenalti] =
HitungWaktuProses(SolusiAwal,waktu_proses,due_date,penalti);

%Catat kondisi awal
SolusiTerbaik = SolusiAwal; %solusi global
TotalPenaltiSolusiTerbaik = TotalPenalti; % Total penalti solusi global
SolusiSaatIni = SolusiAwal; %solusi iterasi
TotalPenaltiSolusiSaatIni = TotalPenalti; % Total penalti solusi iterasi

% Lebah Solusi per Populasi
LebahSolusiPerPopulasi = zeros(JumlahLebah, JumlahJob);
TabelPenaltiLebahSolusiPerPopulasi = zeros(1, JumlahLebah);
TabelOperasiLocalSearch = zeros(JumlahLebah, 3);

LebahSolusiTerbaik = zeros(1, JumlahJob);
TotalPenaltiLebahSolusiTerbaik = 0;
```

```
LebahSolusiTetanggaTabuTerbaik = zeros(1, JumlahJob);
TotalPenaltiLebahSolusiTetanggaTabuTerbaik = 0;

h = waitbar(0,'Silahkan Ditunggu, sedang dalam proses Running...');
% Mulai iterasi TS
for i = 1 : JumlahPopulasiLebah

   %generate lebah sejumlah n
   for j = 1 : JumlahLebah

%        Pilihan = randi(3);
      if i>1
         idx = randperm(5);
         n = idx(1);
         SolusiSaatIni(1,:) = ListTerbaik(n,:);
      end
      pil=randperm(3);
      Pilihan=pil(1);
      switch (Pilihan)
         case 1 % 1-insert
            [LebahSolusiPerPopulasi(j, :) Job_1 Job_2] = PerformInsert(SolusiSaatIni);
            [DbTime WaktuSelesaiPerJob WaktuSelesaiJobTerakhir DaftarKeterlambatan
DaftarPenalti TotalPenalti] = HitungWaktuProses(LebahSolusiPerPopulasi(j,
:),waktu_proses,due_date,penalti);
            TabelPenaltiLebahSolusiPerPopulasi(j)=TotalPenalti;

         case 2 % 1-swap
            [LebahSolusiPerPopulasi(j, :)  Job_1 Job_2] = PerformSwap(SolusiSaatIni);
            [DbTime WaktuSelesaiPerJob WaktuSelesaiJobTerakhir DaftarKeterlambatan
DaftarPenalti TotalPenalti] = HitungWaktuProses(LebahSolusiPerPopulasi(j,
:),waktu_proses,due_date,penalti);
            TabelPenaltiLebahSolusiPerPopulasi(j)=TotalPenalti;

         case 3 % 2-opt
            [LebahSolusiPerPopulasi(j, :)  Job_1 Job_2] = Perform2Opt(SolusiSaatIni);
            [DbTime WaktuSelesaiPerJob WaktuSelesaiJobTerakhir DaftarKeterlambatan
DaftarPenalti TotalPenalti] = HitungWaktuProses(LebahSolusiPerPopulasi(j,
:),waktu_proses,due_date,penalti);
            TabelPenaltiLebahSolusiPerPopulasi(j)=TotalPenalti;

      end
      TabelOperasiLocalSearch(j, :) = [Pilihan  Job_1 Job_2];
   end

   %bedakan antara yg tabu maupun yg tidak tabu
```

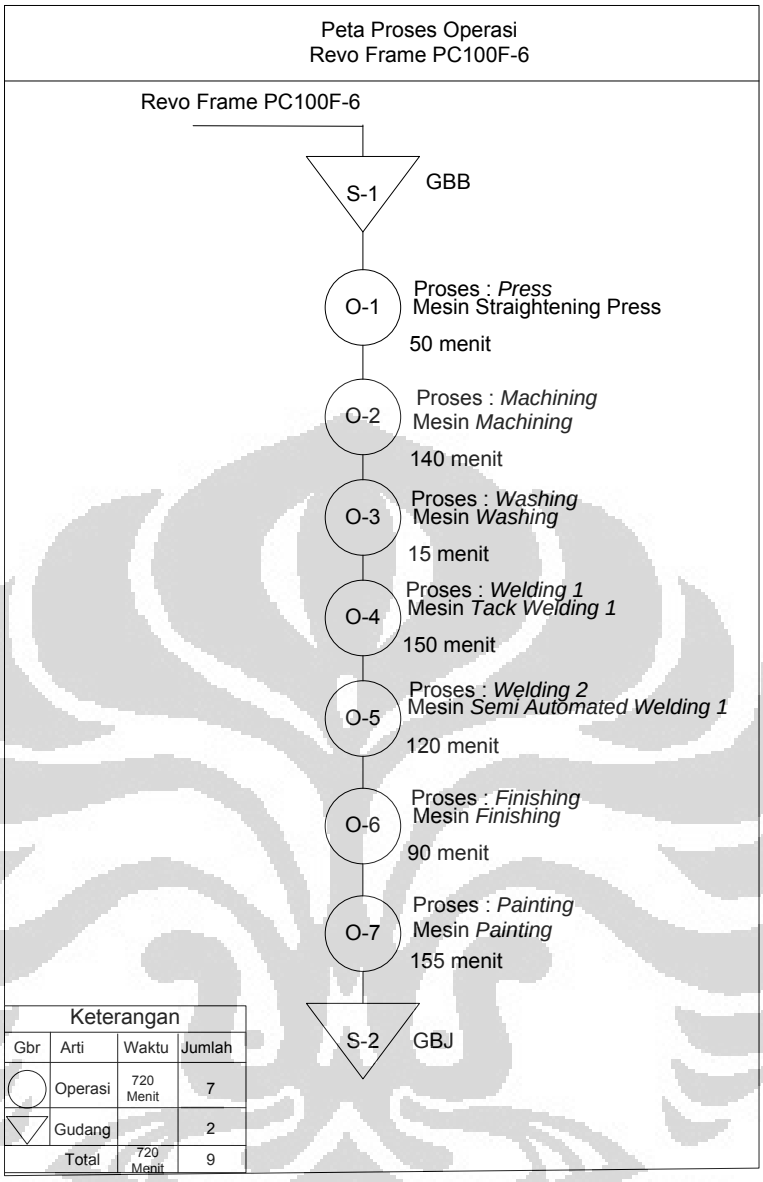
```
   ApakahTabu = zeros(1, JumlahLebah);
   for j = 1 : JumlahLebah
      for k = 1 : PanjangTabu
         if TabelOperasiLocalSearch(j, :) == TabuList(k, :)
            ApakahTabu(j) = 1;
         end
      end
   end

   %cari solusi tetangga terbaik yg tabu maupun yg tidak tabu
   IndeksTabuTerbaik = 1;
   IndeksTidakTabuTerbaik = 1;
   TotalPenaltiLebahSolusiTerbaik = pnalti;
   TotalPenaltiLebahSolusiTetanggaTabuTerbaik = pnalti;
   for j = 1 : JumlahLebah
      if  ApakahTabu(j) == 0 % apabila tidak tabu
         if TabelPenaltiLebahSolusiPerPopulasi(j) < TotalPenaltiLebahSolusiTerbaik
            LebahSolusiTerbaik = LebahSolusiPerPopulasi(j, :);
            TotalPenaltiLebahSolusiTerbaik = TabelPenaltiLebahSolusiPerPopulasi(j);
            IndeksTidakTabuTerbaik = j;
         end
      else
         if TabelPenaltiLebahSolusiPerPopulasi(j) <
TotalPenaltiLebahSolusiTetanggaTabuTerbaik
            LebahSolusiTetanggaTabuTerbaik = LebahSolusiPerPopulasi(j, :);
            TotalPenaltiLebahSolusiTetanggaTabuTerbaik =
TabelPenaltiLebahSolusiPerPopulasi(j);
            IndeksTabuTerbaik = j;
         end
      end
   end

   % uji yg tabu dengan global, kalau tidak masukkan yg tidak tabu
   if TotalPenaltiLebahSolusiTetanggaTabuTerbaik < TotalPenaltiSolusiTerbaik
      SolusiTerbaik = LebahSolusiTetanggaTabuTerbaik; %solusi global
      TotalPenaltiSolusiTerbaik = TotalPenaltiLebahSolusiTetanggaTabuTerbaik; % Penalti
solusi global
      SolusiSaatIni = LebahSolusiTetanggaTabuTerbaik; %solusi iterasi
      TotalPenaltiSolusiSaatIni = TotalPenaltiLebahSolusiTetanggaTabuTerbaik; % Penalti
solusi iterasi
      %update tabu list
      IndeksTabuList = mod(i, PanjangTabu) + 1;
      TabuList(IndeksTabuList, :) = TabelOperasiLocalSearch(IndeksTabuTerbaik, :);
   else %update solusi saat ini dari solusi tidak tabu
      SolusiSaatIni = LebahSolusiTerbaik; %solusi iterasi
      TotalPenaltiSolusiSaatIni = TotalPenaltiLebahSolusiTerbaik; % Penalti solusi iterasi
```

```
        if TotalPenaltiLebahSolusiTerbaik < TotalPenaltiSolusiTerbaik
            SolusiTerbaik = LebahSolusiTerbaik; %solusi global
            TotalPenaltiSolusiTerbaik = TotalPenaltiLebahSolusiTerbaik; % Penalti solusi
global
        end
        %update tabu list
        IndeksTabuList = mod(i, PanjangTabu) + 1;
        TabuList(IndeksTabuList, :) = TabelOperasiLocalSearch(IndeksTidakTabuTerbaik, :);
    end
    [UrutanPenalti NoUrut]=sort(TabelPenaltiLebahSolusiPerPopulasi);
    ListTerbaik(1,:)=LebahSolusiTerbaik;
    ListTerbaik(2,:)=LebahSolusiPerPopulasi(NoUrut(1), :);
    ListTerbaik(3,:)=LebahSolusiPerPopulasi(NoUrut(2), :);
    ListTerbaik(4,:)=LebahSolusiPerPopulasi(NoUrut(3), :);
    ListTerbaik(5,:)=LebahSolusiPerPopulasi(NoUrut(4), :);
    waitbar(i / JumlahPopulasiLebah);
end
close(h);
waktu_running = toc;
[Waktu_Semua_Job_Per_Mesin Waktu_Semua_Job_Selesai TotalTime
DaftarKeterlambatan DaftarPenalti TotalPenalti] =
HitungWaktuProses(SolusiTerbaik,waktu_proses,due_date,penalti);
disp(['Solusi Job Shop Terbaik : ',num2str(SolusiTerbaik)]);
[kolom]=find(DaftarKeterlambatan~=0);
JumlahKeterlambatan=length(kolom);
disp(['Jumlah Keterlambatan : ',num2str(JumlahKeterlambatan)]);
DaftarKeterlambatan
DaftarPenalti
Waktu_Semua_Job_Per_Mesin
Waktu_Semua_Job_Selesai
due_date
disp(['Total Penalti : ',num2str(TotalPenalti)]);
disp(['Waktu Total Semua Job Selesai : ',num2str(TotalTime)]);
disp(['Waktu Running Program : ',num2str(waktu_running)]);
```

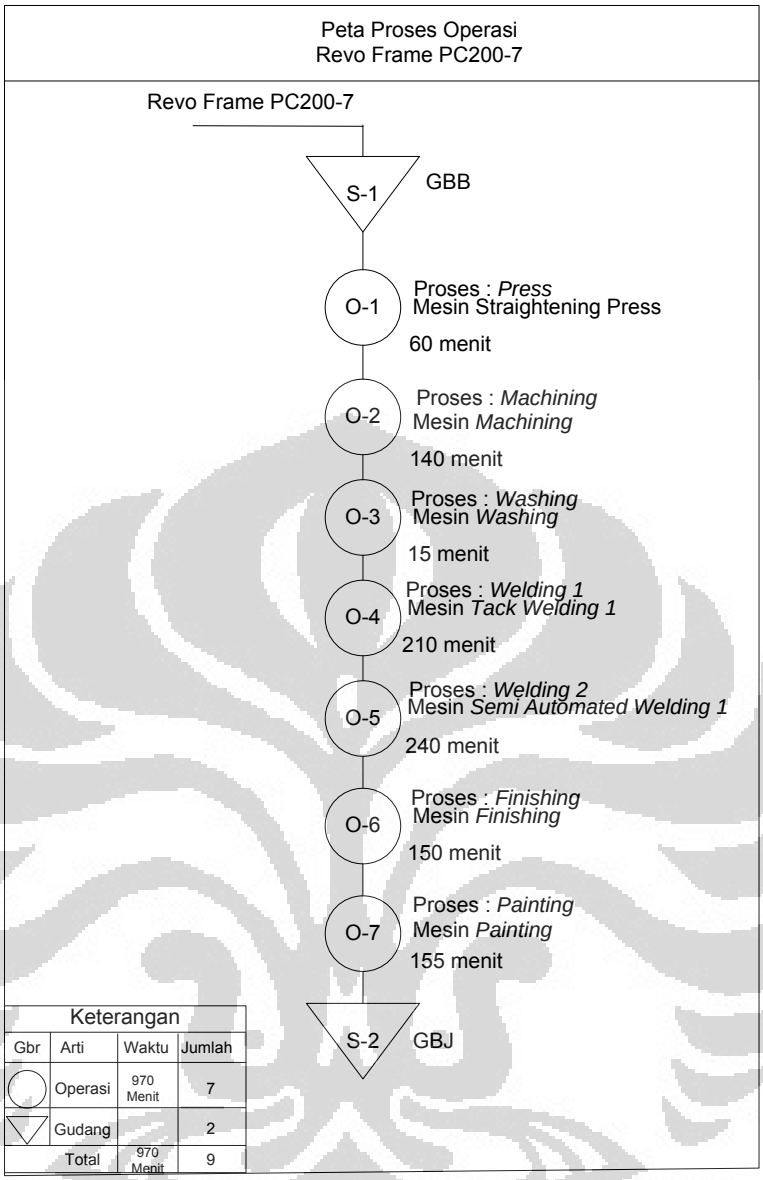
PC400LC-7 S/N 50001-UP (Overseas Version)
00010467



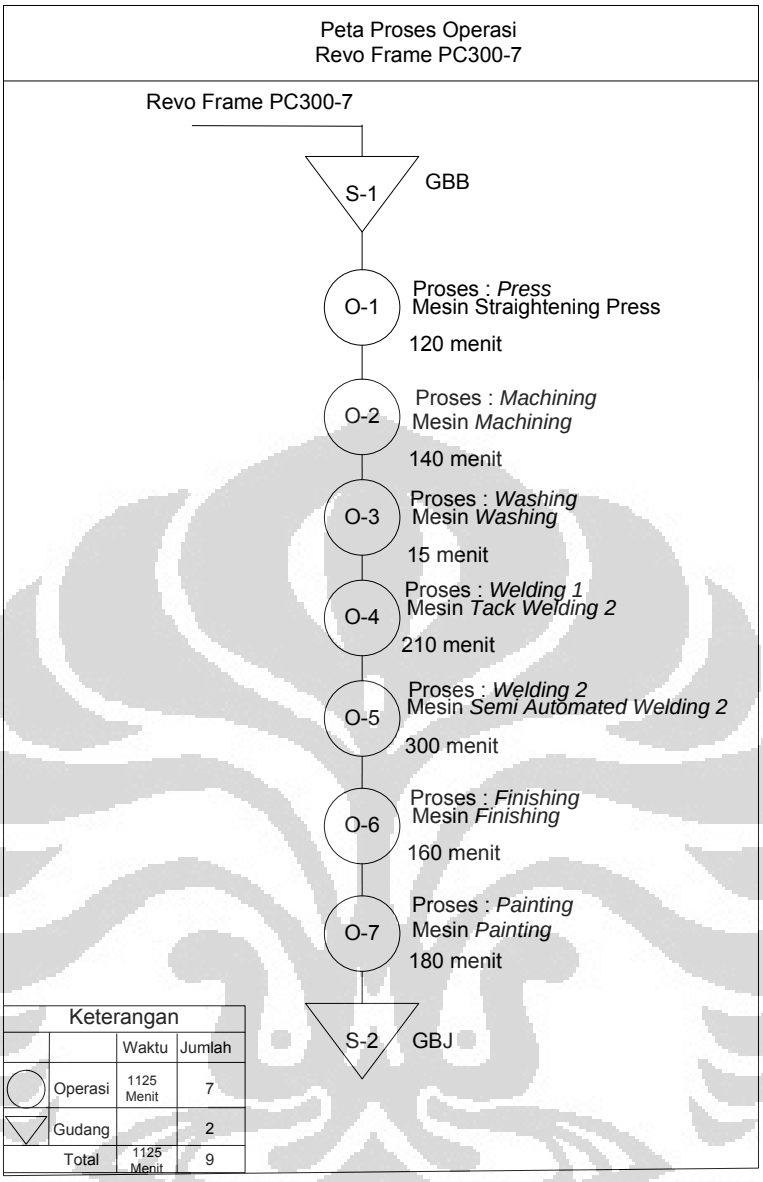
Gambar L5.1 Gambaran Produk *Revo Frame*



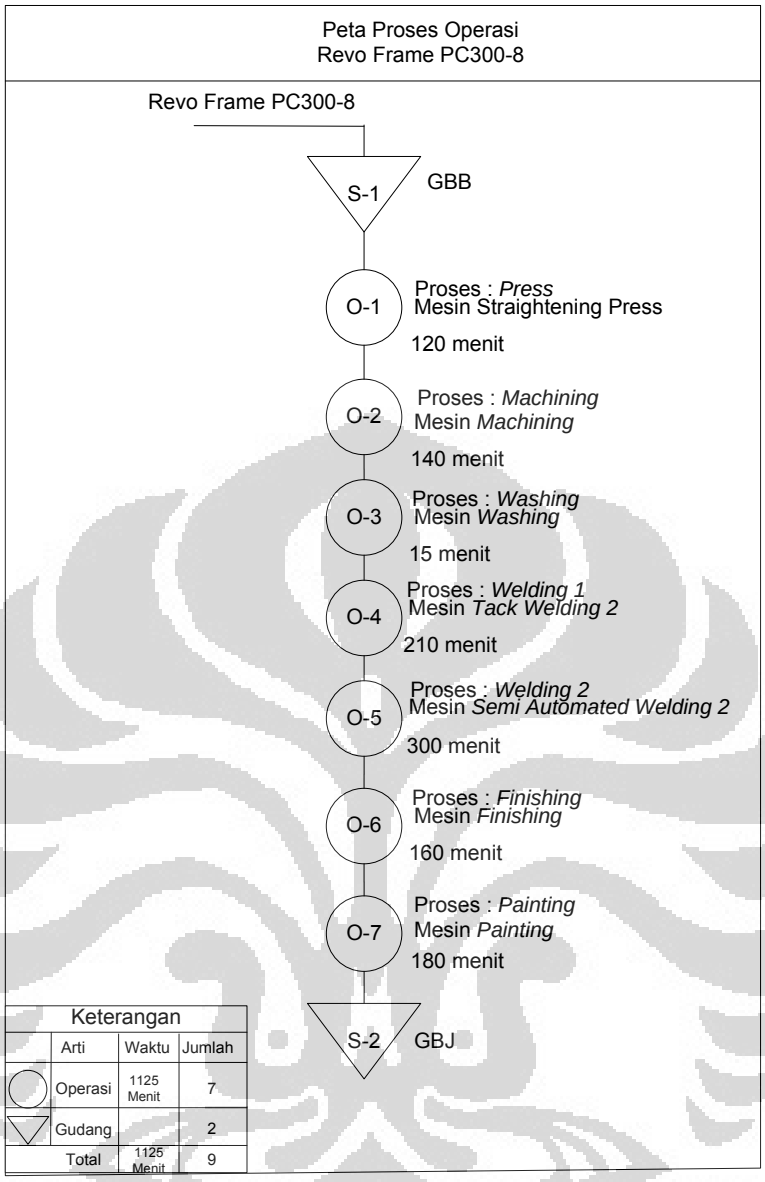
Gambar L5.2 Peta Proses Operasi *Revo Frame* PC 100 F-6



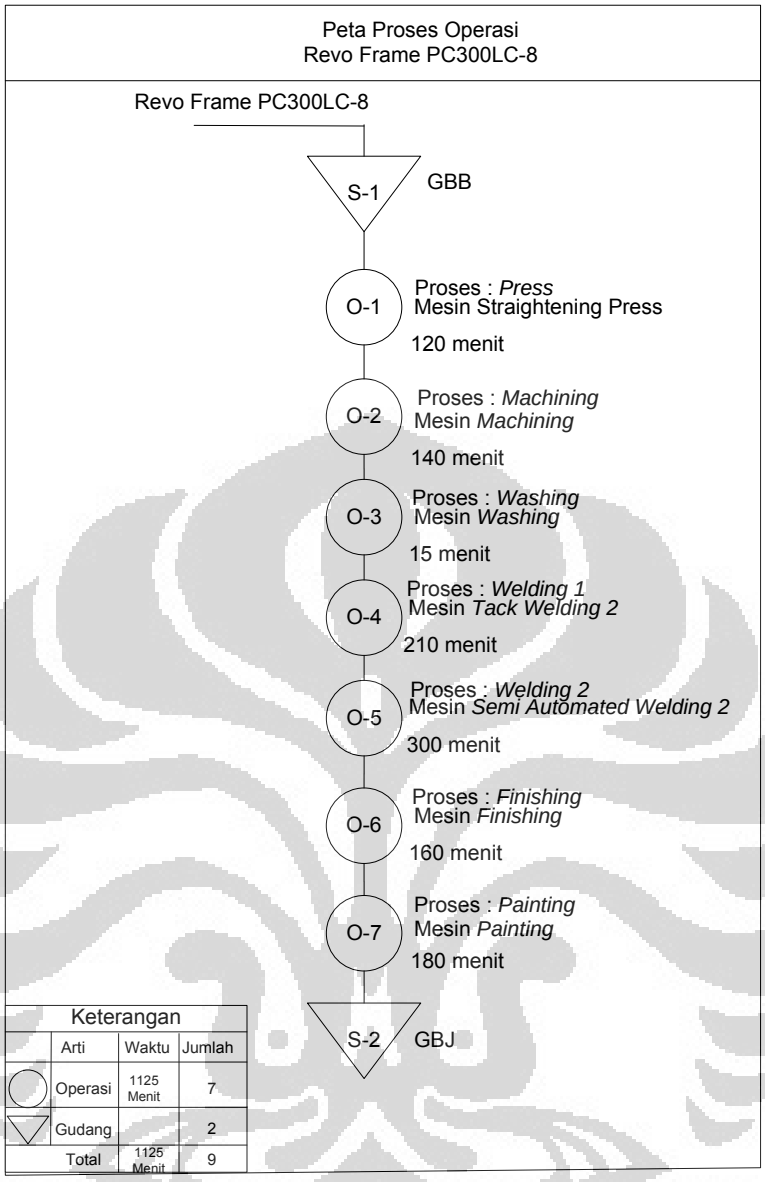
Gambar L5.3 Peta Proses Operasi *Revo Frame* PC 200-7



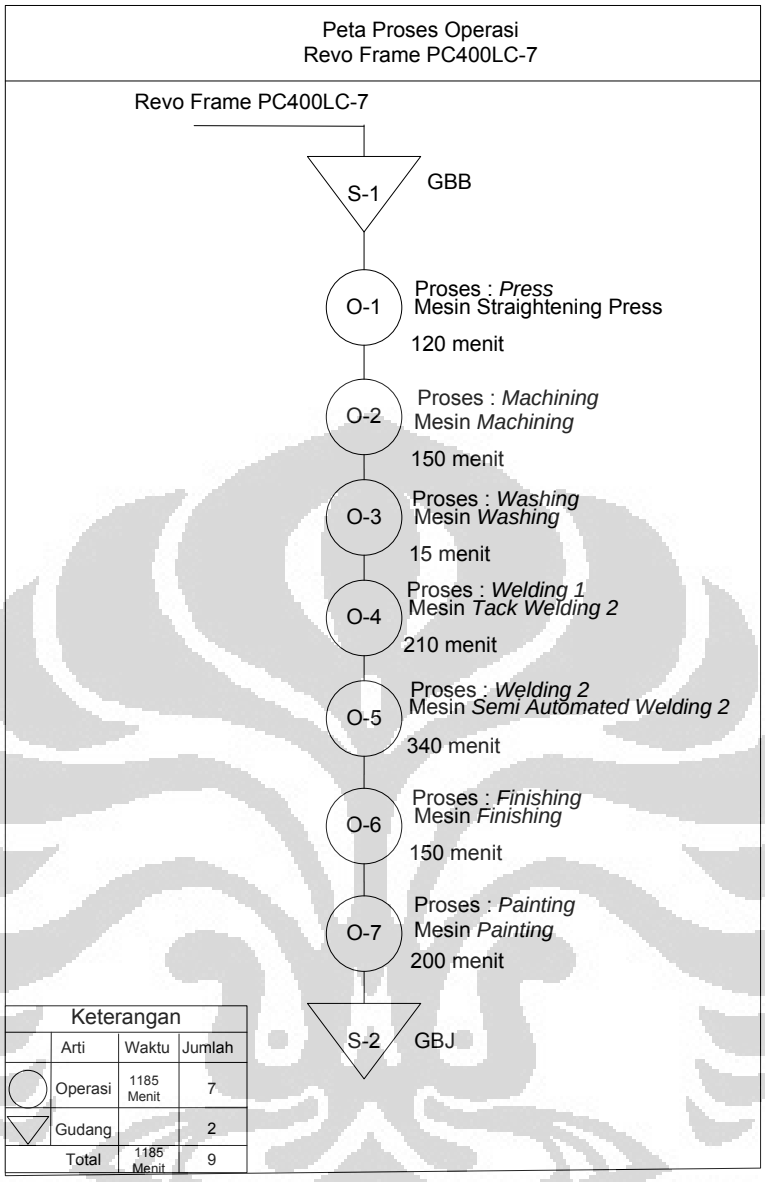
Gambar L5.4 Peta Proses Operasi *Revo Frame* PC 300-7



Gambar L5.5 Peta Proses Operasi *Revo Frame* PC 300-8



Gambar L5.6 Peta Proses Operasi *Revo Frame* PC 300LC-8



Gambar L5.7 Peta Proses Operasi *Revo Frame* PC 400LC-7