PENJADWALAN PRODUKSI KERTAS MENGGUNAKAN ALGORITMA POUR DAN ALGORITMA NEH di PT. KERTAS LECES PROBOLINGGO

SKRIPSI

Oleh

Candra Kuncoro

NIM. 071810101098

JURUSAN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS JEMBER

2013

PENJADWALAN PRODUKSI KERTAS MENGGUNAKAN ALGORITMA POUR DAN ALGORITMA NEH di PT. KERTAS LECES PROBOLINGGO

SKRIPSI

disusun untuk memenuhi salah satu syarat menyelesaikan pendidikan Strata satu (S1) pada jurusan Matematika Fakultas MIPA Universitas Jember.

Oleh

CandraKuncoro

NIM. 071810101098

JURUSAN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS JEMBER

2013

PERSEMBAHAN

Skripsi ini saya persembahkan untuk:

1. Ibunda Buati dan Ayahanda Husin yang tercinta;
2. Istriku Istikanah tercinta serta anak-anakku yaitu Muhammad Ghazi Al-Liwa' dan Arkan 'Ammar Firas;
3. guru-guruku sejak taman kanak-kanak sampai dengan perguruan tinggi;
4. Bapak Kiswara Agung Santoso, S.Si, M.Kom dan Bapak Drs. Rusli Hidayat, M.Sc selaku DPU dan DPA yang telah bersabar dan sungguh-sungguh membimbing dalam penyusunan skripsi ini.
5. Almamater Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember;
6. teman-teman angkatan 2005, 2006 dan 2007 yang telah banyak memberi motivasi dan kerjasama yang baik selama di kampus;
7. syabab-syabab Hizbut Tahrir Link Kampus Jember yang selalu gigih dan sungguh-sungguh berjuang menyampaikan kebenaran Islam sebagai *rahmatan lil 'alamin*.

MOTTO

"Allah telah menjanjikan kepada orang-orang diantara kamu yang beriman dan yang mengerjakan kebajikan, bahwa Dia sungguh akan menjadikan mereka berkuasa di bumi sebagaimana Dia telah menjadikan orang-orang sebelum mereka berkuasa dan sungguh Dia meneguhkan bagi mereka dengan agama yang telah Dia ridhoi. Dan Dia benar-benar mengubah (keadaan) mereka, setelah berada dalam ketakutan menjadi aman sentosa. Mereka (tetap) menyembah-Ku dengan tidak mempersekutukan-Ku dengan sesuatu pun. Tetapi barang siapa (tetap) kafir setelah (janji) itu, maka mereka itulah orang-orang fasik."

(Terjemahan *Al-Qur'an* SuratAn - Nuur ayat 55)*)

---

*)Departemen Agama Republik Indonesia.1998. *Al Quran dan terjemahannya* . Semarang: PT Kumudasmoro Grafindo.

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Candra Kuncoro

NIM : 071810101098

menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul "PENJADWALAN PRODUKSI KERTAS MENGGUNAKAN ALGORITMA POUR DAN ALGORITMA NEH DI PT. KERTAS LECES PROBOLINGGO adalah benar-benar hasil karya sendiri, kecuali kutipan yang sudah saya sebutkan sumbernya, belum pernah diajukan pada institusi mana pun, dan bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa ada tekanan dan paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika ternyata di kemudian hari pernyataan ini tidak benar.

Jember, Juni 2013

Yang menyatakan,

Candra Kuncoro

NIM. 071810101098

SKRIPSI

PENJADWALAN PRODUKSI KERTAS MENGGUNAKAN ALGORITMA POUR DAN ALGORITMA NEH DI PT. KERTAS LECES PROBOLINGGO

Oleh

Candra Kuncoro

NIM. 071810101098

Pembimbing:

Dosen Pembimbing Utama : Kiswara Agung Santoso, S.Si, M.Kom

Dosen Pembimbing Anggota : Drs. Rusli Hidayat, M.Sc.

PENGESAHAN

Skripsi berjudul **"**Penjadwalan Produksi Kertas Menggunakan Algoritma Pour dan Algoritma NEH di PT. Kertas Leces Probolinggo" telah diuji dan disahkan pada:

Hari, tanggal :

tempat : Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember

Tim Penguji :

Ketua,	Sekretaris,
Kiswara Agung Santoso, S.Si, M.Kom. NIP. 197209071998031003	Drs. Rusli Hidayat, M.Sc. NIP. 196610121993031001
Anggota I,	Anggota II,
Kusbudiono, S.Si, M.Si. NIP 197704302005011001	Yuliani Setia Dewi, S.Si., MSi. NIP : 197407162000032001

Mengesahkan

Dekan,

Prof. Drs. Kusno, DEA., Ph.D.
NIP. 196101081986021001

RINGKASAN

"Penjadwalan Produksi Kertas Menggunakan Algoritma Pour dan Algoritma NEH di PT. Kertas Leces Probolinggo" Candra Kuncoro, 071810101098; 2013: 47 halaman; Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Penjadwalan merupakan suatu kegiatan pengalokasian sumber daya yang terbatas untuk mengerjakan sejumlah pekerjaan. Proses penjadwalan timbul jika terdapat keterbatasan sumber daya yang dimiliki, sehingga diperlukan adanya pengaturan sumber-sumber daya yang ada secara efisien. Penjadwalan produksi merupakan kegiatan perencanaan produksi yang terdapat pada perusahaan manufaktur. Adapun tujuan dari penjadwalan produksi umumnya ialah untuk mengoptimalkan dimensi tertentu, yaitu *makespan* (waktu penyelesaian semua tugas atau pekerjaan), keuntungan perusahaan dan waktu tunggu mesin *(machine idletime)*. Penjadwalan *flowshop* adalah salah satu jenis penjadwalan produksi dimana setiap *job* akan melalui setiap mesin dengan urutan yang seragam. PT. Kertas Leces Probolinggo merupakan salah satu industri yang menggunakan pola aliran *flowshop* dalam proses produksinya. Industri tersebut menggunakan 5 buah mesin yakni *stock preparation*, *wire part*, *press part*, *drier part*, dan *finishing* serta menghasilkan 4 jenis produk yaitu kertas HVS 45 *gsm*, HVS 60 *gsm*, BC Super, dan MG *paper*. PT. Kertas Leces Probolinggo sering kali menambah waktu operasional guna memenuhi permintaan konsumen yang mengakibatkan penambahan biaya produksi. Oleh karena itu, dalam skripsi ini dibahas penyelesaian *flowshop* dengan algoritma Pour dan algoritma NEH untuk membangun jadwal dengan *makespan* yang optimal serta perbandingan kedua algoritma berdasarkan kompleksitas waktu yang diperlukan.

Penelitian dilakukan melalui beberapa langkah, yaitu mengolah data yang diperoleh menjadi data urutan mesin dan waktu proses kemudian menjadwalkan dengan kedua algoritma. Selanjutnya menghitung kompleksitas waktu dari tiap algoritma, dan membandingkan hasil *makespan* dan kompleksitas waktu yang

diperoleh. Yang terakhir adalah menentukan kesimpulan berdasarkan perbandingan sebelumnya.

Penjadwalan yang dilakukan melibatkan 5 buah mesin dan menghasilkan 4 jenis kertas. Dimana setiap jenis produk kertas diproses pada 5 buah mesin yang sama dengan urutan yang seragam. Penjadwalan dengan menggunakan algoritma *Pour dan* algoritma NEH menghasilkan nilai *makespan* yang sama yakni 1.620 menit. Begitu pula berdasarkan kompleksitas waktu yang diperlukan dalam perhitungan, penggunaan algoritma Pour dengan $O(mn^2)$ membutuhkan waktu yang sama jika dibandingkan dengan menggunakan algoritma NEH dengan $O(mn^2)$. Artinya, penggunaan algoritma Pour dan algoritma NEH memiliki performa yang baik untuk menjadwalkan produksi kertas. Oleh karena itu, dalam skripsi ini disertakan sebuah program aplikasi penjadwalan *flowshop* yang memanfaatkan bahasa pemrograman Matlab untuk membantu mempercepat dalam proses perhitungan.

PRAKATA

Puji syukur kehadirat Allah SWT. atas segala rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Penjadwalan Produksi Kertas Menggunakan Algoritma Pour dan Algoritma NEH di PT. Kertas Leces Probolinggo". Skripsi ini disusun untuk memenuhi salah satu syarat menyelesaikan pendidikan Strata satu (S1) pada jurusan Matematika Fakultas MIPA Universitas Jember.

Penyusunan skripsi ini tidak lepas dari bantuan berbagai pihak. Oleh karena itu, penulis menyampaikan terima kasih kepada:

1. Ibunda Buati dan Ayahanda husin yang tercinta;
2. Kiswara Agung Santoso, S.Si, M.Kom., selaku Dosen Pembimbing Utama dan Drs. Rusli Hidayat, M.Sc., selaku Dosen Pembimbing Anggota yang telah meluangkan waktu, pikiran, dan perhatian dalam penulisan skripsi ini;
3. Kusbudiono, S.Si, M.Si, dan Yuliani Setia Dewi, S.Si., M.Si., selaku dosen penguji yang telah memberi saran dan masukan dalam skripsi ini;
4. Istikanah, Muhammad Ghazi Al-liwa' dan Arkan 'Ammar Firas yang tersayang;
5. teman-teman angkatan 2007. Terima kasih telah menemani dan memberi semangat untuk terus maju menghadapi hari-hari sulit selama masa perkuliahan;
6. semua pihak yang tidak dapat disebut satu persatu.

Penulis juga menerima segala kritik dan saran dari semua pihak demi kesempurnaan skripsi ini. Akhirnya penulis berharap, semoga skripsi ini bermanfaat.

Jember, Juni 2013 Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR LAMPIRAN

Halaman

BAB I. PENDAHULUAN

1.1 Latar Belakang

Perkembangan dunia usaha sekarang ini mengalami laju pertumbuhan yang sangat pesat, ini ditandai dengan bermunculannya berbagai jenis usaha khususnya dalam bidang manufaktur. Pertumbuhan yang pesat tersebut tentu akan menimbulkan persaingan antara satu perusahaan dengan perusahaan lainnya. Penjadwalan produksi flowshop merupakan kegiatan perencanaan produksi yang terdapat pada perusahaan manufaktur. Penjadwalan produksi melibatkan n job dan m mesin dalam proses produksinya dimana setiap job mengandung informasi tentang jenis produk dan jumlah pesanan. Setiap job memiliki waktu proses yang berbeda dalam setiap mesin. Penjadwalan produksi bertujuan untuk mengurutkan pengerjaan job-job agar mendapatkan suatu kondisi yang optimal. Salah satu tujuan yang penting dalam penjadwalan produksi adalah minimasi *makespan*, yaitu waktu yang dibutuhkan untuk menyelesaikan semua job dalam sistem produksi.

Menurut Baker (1974), penjadwalan flowshop adalah salah satu jenis penjadwalan produksi dimana setiap *n* job akan melalui setiap *m* mesin dengan urutan yang seragam. Penjadwalan flowshop merupakan suatu pergerakan unit-unit yang terus-menerus melalui serangkaian stasiun-stasiun kerja yang disusun berdasarkan produk. Pada penjadwalan flowshop, sumber daya yang dialokasikan akan dilewati oleh setiap job secara berurutan atau dengan kata lain setiap job memiliki rute pengerjaan yang sama. Ukuran performasi penjadwalan tergantung pada kriteria yang digunakan, antara lain total waktu untuk penyelesaian setiap job (*makespan*) yang minimum, rata-rata keterlambatan minimum (*mean tardiness*), rata-rata waktu penyelesaian job yang minimum (*mean flow time*) dan sebagainya. Penentuan penjadwalan yang memenuhi seluruh kriteria yang ada sangat sulit dilakukan. Oleh karena

itu, diambil kriteria yang dapat mewakili seuruh kriteria yang ada yaitu meminimumkan total waktu penyelesaian job (*makespan)*.

PT. Kertas Leces Probolinggo adalah salah satu perusahan yang memproduksi kertas dengan beragam jenis. Saat ini perusahaan tersebut menggunakan aturan *First Come First Serve* (FCFS). Pada penjadwalan ini order yang tiba lebih awal akan dilayani dahulu sehingga aturan ini tidak mempermasalahkan lama atau singkatnya waktu proses.

Terdapat banyak model penjadwalan yang dapat menyelesaikan penjadwalan flowshop seperti CDS, Dannenbring, Algoritma Genetika, Tabu Search. Model yang digunakan untuk menyelesaikan penjadwalan flowshop pada tugas akhir ini yaitu algoritma Pour dan Nawaz, Enscore dan Ham (NEH) dengan tujuan meminimumkan waktu penyelesaian job (*makespan*).

1.2 Rumusan Masalah

Permasalahan yang akan diselesaikan dalam penulisan tugas akhir ini antara lain sebagai berikut.

a. bagaimana penerapan algoritma Pour dan NEH pada penjadwalan mesin di PT. Kertas Leces Probolinggo.
b. bagaimana perbandingan nilai minimum *makespan* untuk optimasi waktu dan performa pada algoritma Pour dan NEH untuk penjadwalan flowshop.

1.3 Tujuan

Tujuan yang ingin dicapai dalam penulisan tugas akhir ini antara lain sebagai berikut:

a. menentukan minimum *makespan* untuk optimasi waktu pada penjadwalan flowshop di PT. Kertas Leces Probolinggo dengan menggunakan algoritma Pour dan NEH.
b. membandingkan performa antara algoritma Pour dan NEH pada penjadwalan flowshop.

c. membuat program untuk menentukan minimum *makespan* dengan algoritma Pour dan NEH pada penjadwalan flowshop menggunakan bahasa pemrograman java.

1.4 Manfaat

Manfaat yang diharapkan dari penulisan skripsi ini antara lain sebagai berikut:

a. memberikan jadwal produksi untuk PT. Kertas Leces Probolinggo dengan menggunakan algoritma Pour dan NEH.
b. mendapatkan hasil perbandingan kedua algoritma tersebut sehingga bisa dipilih algoritma yang terbaik.

BAB II. TINJAUAN PUSTAKA

2.1 Definisi Penjadwalan Produksi

Menurut Ginting (2009), penjadwalan produksi diartikan sebagai pengalokasian sumber daya yang terbatas untuk mengerjakan sejumlah pekerjaan. Sedangkan menurut Baker (1974), penjadwalan (*scheduling*) didefinisikan sebagai proses pengalokasian sumber untuk memilih sekumpulan tugas dalam jangka waktu tertentu. Definisi ini dapat dijabarkan dalam dua arti yang berbeda. Pertama, penjadwalan merupakan sebuah fungsi pengambilan keputusan dalam menentukan jadwal yang paling tepat. Kedua, penjadwalan merupakan teori yang berisi kumpulan prinsip, model, teknik, dan konklusi logis dalam proses pengambilan keputusan.

Keputusan yang dibuat dalam penjadwalan meliputi pengurutan pekerjaan (*sequencing*), waktu mulai dan selesai pekerjaan (*timing*), urutan operasi untuk suatu pekerjaan (*routing*). Masalah penjadwalan selalu berkaitan dengan pengurutan produksi *(sequencing)* yang didefinisikan sebagai penentuan urutan-urutan kedatangan dan bermacam-macam pekerjaan yang harus diselesaikan dalam jangka waktu tertentu. Masalah penjadwalan seringkali muncul jika terdapat sekumpulan tugas secara bersamaan, sedangkan peralatan yang dimiliki terbatas.

Masukan dari suatu penjadwalan mencakup jenis dan banyaknya *part* yang akan dioperasi, urutan ketergantungan antar operasi, waktu proses untuk masing-masing operasi, serta fasilitas yang dibutuhkan oleh setiap operasi. Keluaran penjadwalan meliputi *dispatch list* (daftar urutan-urutan pemrosesan *part* serta waktu mulai dan selesai dari pemrosesan *part*).

Penjadwalan mempunyai beberapa tujuan yaitu :

a. Meningkatkan produktivitas mesin, yaitu dengan mengurangi waktu menganggur.
b. Mengurangi persediaan barang setengah jadi *(work in process inventory)* untuk mengurangi biaya penyimpanan dengan jalan mengurangi jumlah rata-

rata pekerjaan yang menunggu dalam antrian suatu mesin karena mesin terlalu sibuk.

c. Mengurangi waktu keterlambatan karena batas waktu *(due date)* telah dilampaui dengan cara mengurangi maksimum keterlambatan maupun dengan mengurangi jumlah pekerja yang terlambat.
d. Meminimasi ongkos produksi.
e. Pemenuhan *due date* karena dalam kenyataannya apabila terjadi keterlambatan pemenuhan *due date* yang telah ditetapkan dapat dikenakan suatu denda atau *penalty.*

Menurut Baker, jika *makespan* suatu penjadwalan adalah konstan maka urutan kerja yang tepat akan menurunkan *flow time* dan rataan *work in process.*

2.1.1 Permasalahan Dalam Penjadwalan Produksi

Masalah penjadwalan sering kali muncul jika terdapat sekumpulan tugas yang sudah ditetapkan harus dikerjakan terlebih dahulu, bagaimana urutan kerja dan tugas-tugas yang berikutnya, serta pengalokasian tugas pada mesin sehingga diperoleh suatu proses yang terjadwal. Pada umumnya persoalan penjadwalan ini dipecahkan dengan sendirinya menurut kebiasaan tanpa memberikan perhatian yang lebih besar sehingga pemecahan persoalan dengan suatu teknik baru akan lebih mudah dan lebih menguntungkan. Cara yang umum dilakukan adalah cara yang didasarkan pada FCFS (*First Come First Serve),* sehingga tugas yang datang lebih dahulu akan dilayani lebih awal daripada tugas yang datang kemudian.

Secara umum, persoalan penjadwalan dapat dinyatakan sebagai berikut :

a. Misalkan α adalah resiko yang ditanggung karena mengerjakan tugas lebih dahulu daripada tugas B.
b. Misalkan β adalah resiko yang ditanggung karena mengerjakan tugas lebih dahulu daripada tugas A.
c. Jika α lebih baik daripada β maka tugas B dikerjakan lebih awal kemudian dikuti oleh tugas A.

Pemilihan α dan β ini dapat dikaitkan dengan pemilihan kriteria optimalitas yang diterapkan oleh pengambil keputusan.

2.1.2 Klasifikasi Penjadwalan Produksi

Penjadwalan produksi dapat berbeda-beda dilihat dari kondisi yang mendasarinya. Beberapa model penjadwalan sering terjadi di dalam proses produksi berdasarkan beberapa keadaan antara lain :

a. Berdasarkan mesin yang dipergunakan dalam proses
   1) Penjadwalan pada mesin tunggal *(single machine shop)*
   2) Penjadwalan pada mesin jamak *(m machine)*

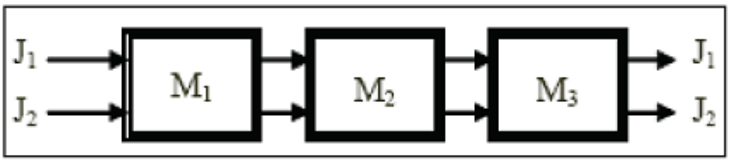
b. Berdasarkan pola aliran proses
   1) Penjadwalan Flowshop

      Proses produksi dengan aliran flowshop berarti proses produksi dengan pola aliran identik dari satu mesin ke mesin lain atau dengan kata lain *job* akan diproses seluruhnya mengalir pada arah jalur produk yang sama.

      Sebagai contoh, misalkan terdapat 2 job dan 3 mesin. Data waktu proses untuk permasalahan tersebut disajikan dalam bentuk tabel 2.1 dibawah ini. Sedangkan aliran jobnya ditunjukkan pada gambar 2.1.

Tabel 2.1 Waktu proses tiap job dalam mesin

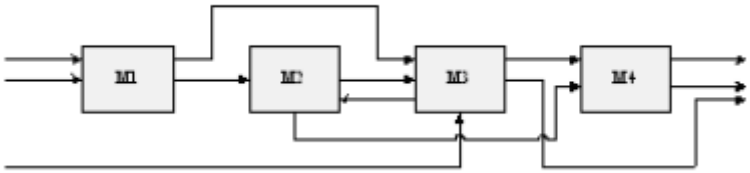
Job	Mesin		
	M1	M2	M3
J1	2	6	5
J2	4	3	1



Gambar 2.1 Jalur Proses Flowshop

2) Penjadwalan Jobshop

Proses produksi dengan aliran jobshop berarti proses produksi dengan pola aliran atau rute proses pada tiap mesin yang spesifik untuk setiap pekerjaan, dan mungkin berbeda untuk tiap job. Akibat aliran proses yang tidak searah ini, maka setiap job yang akan diproses pada satu mesin dapat merupakan job yang baru atau job dalam proses, dan job yang keluar dari suatu mesin dapat merupakan job tadi atau job dalam proses.



Gambar 2.2 Jalur Proses Jobshop

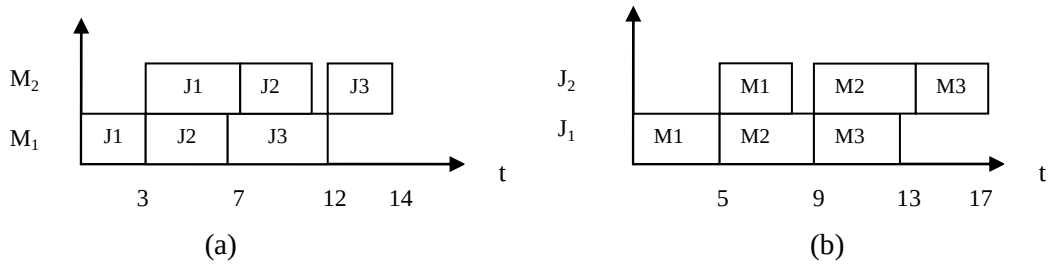
c. Berdasarkan pola kedatangan job
   1) Penjadwalan statis yaitu job yang datang bersamaan dan siap dikerjakan pada mesin yang tidak bekerja.
   2) Penjadwalan dinamis yaitu kedatangan job tidak menentu.
d. Berdasarkan sifat informasi yang diterima
   1) Penjadwalan deterministik yaitu informasi yang diperoleh pasti, misalnya informasi tentang pekerjaan dan mesin seperti waktu kedatangan pekerjaan dan waktu proses.
   2) Penjadwalan stokastik yaitu informasi yang diperoleh tidak pasti tetapi memiliki kecenderungan yang jelas atau menyangkut adanya distribusi probabilitas tertentu.

2.1.3 Diagram *Gantt*

Masalah penjadwalan sebenarnya masalah murni pengalokasian dan dengan bantuan model matematis akan dapat ditentukan solusi optimal. Model-model penjadwalan akan memberikan rumusan masalah yang sistematik berikut dengan solusi yang diharapkan. Sebagai alat bantu yang digunakan dalam

menyelesaiakan masalah penjadwalan dikenal satu model yang sederhana dan umum digunakan secara luas yakni diagram *Gantt (Gantt chart)*. Diagram *Gantt* merupakan grafik hubungan antara alokasi sumber daya dengan waktu.

Gantt chart terdiri dari 2 jenis yaitu *Machine Oriented Gantt Chart* dan *Job Oriented Gantt Chart* seperti ditunjukkan pada Gambar 2.3 (a) dan Gambar 2.1 (b).



Gambar 2.3 Diagram *Gantt*

Dari diagram *Gantt* dapat diketahui urutan dari *job* yang memberikan kriteria penjadwalan terbaik, misalnya waktu pemrosesan tersingkat, penggunaan mesin yang memiliki waktu proses tertinggi, waktu tunggu minimum dan lain-lain.

Keuntungan menggunakan diagram *gantt* adalah sebagai berikut.

a. dalam situasi keterbatasan sumber, penggunaan diagram *gantt* memungkinkan evaluasi lebih awal mengenai penggunaan sumber daya seperti yang telah direncanakan.
b. kemajuan pekerjaan mudah diperiksa pada setiap waktu karena sudah tergambar dengan jelas.
c. semua pekerjaan diperlihatkan secara grafis dalam suatu diagram yang mudah dipahami (Ginting, 2009).

2.1.4 Kriteria Optimalitas

Beberapa kriteria optimalitas dalam proses penjadwalan adalah sebagai berikut :

a. Berhubungan dengan Waktu

Beberapa kriteria optimalitas yang dapat digunakan adalah :

1. Minimasi *Mean Flow Time*

Kriteria ini menunjukkan rata-rata waktu yang digunakan setiap komponen dilantai produksi.

2. Minimasi *Makespan*

Makespan adalah sejumlah waktu yang dibutuhkan untuk menyelesaikan seluruh proses pada semua komponen yang dijadwalkan mulai dari saat pemrosesan komponen pertama sampai komponen terakhir selesai diproses.

3. Pemenuhan *Due Date*

Due date adalah batas waktu yang ditetapkan konsumen agar seluruh produk yang dipesannya sudah siap. Pihak produsen selalu berusaha untuk memenuhi *due date* tersebut terutama untuk produk-produk yang kritis.

b. Berhubungan dengan Ongkos

Kriteria ini lebih mengarah ke biaya produksi seperti biaya persediaan, biaya pinalti dan sebagainya dan tidak memperhatikan kriteria waktu yang ada sehingga dengan suatu penjadwalan produksi tertentu diharapkan mendapatkan ongkos yang minimal.

c. Kriteria Gabungan

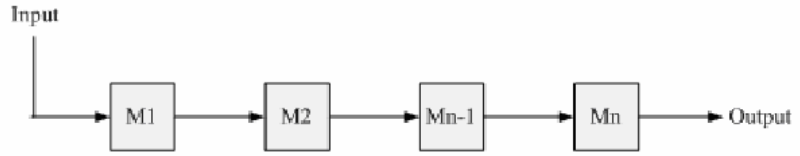
Beberapa kriteria optimalitas tersebut dapat digabungkan dan dikombinasikan menjadi beberapa kriteria yang benar-benar diperlukan.

2.2 Penjadwalan Flowshop

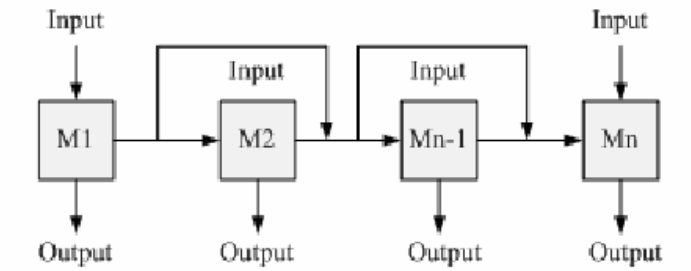
Sistem penjadwalan dalam flowshop adalah penjadwalan dari seluruh job dengan urutan proses sama dan masing-masing job menuju ke masing-masing mesin dalam waktu tetentu. Sistem ini dapat digambarkan seperti urutan linear pada mesin-mesin seperti pada lini perakitan. Setiap job diproses sesuai dengan urutan prosesnya dan dari suatu mesin ke mesin lainnya. Penjadwalan yang memiliki urutan yang sama atas penggunaan masing-masing mesin disebut dengan *permutation schedule*. Dalam kriteria pengukuran diperlukan penjadwalan yang terus berjalan tanpa adanya waktu menganggur. Perhitungan penjadwalan harus dipertimbangkan ketika diperoleh solusi yang optimal dengan meningkatkan jumlah job atau mesin.

Pada umumnya pada setiap operasi berikutnya berasal dari satu operasi yang mendahuluinya dan operasi kedua dari terakhir mempunyai satu operasi yang mengikutinya. Oleh karena itu, setiap job memiliki urutan operasi yang spesifik untuk menyelesaikan job tersebut. Tipe struktur ini sering disebut sebagai *linier precedence diagram*. Lantai produksi terdiri dari *m* mesin berbeda, setiap job terdiri dari *m* operasi yang memerlukan mesin yang berbeda. Karakteristik flowshop dinyatakan dengan aliran pekerjaan yang terarah. Pada pekerjaan flowshop penomoran mesin dimungkinkan, sehingga jika operasi ke-*j* dari suatu job mendahului operasi ke-*k*, maka mesin yang diperlukan dari operasi ke-*j* mempunyai nomor yang lebih kecil dibandingkan dengan mesin yang dibutuhkan oleh operasi ke-*k*. Mesin-mesin dalam flow shop diberi nomor 1, 2, 3, …, *m* dan operasi job ke-*i* ditandai dengan (i, 1), (i, 2), …, (i, *m*).

Setiap job dapat diperlakukan seolah-olah job tersebut memiliki *m* operasi yang tetap. Aliran pekerjaan flowshop terbagi menjadi 2 yaitu *pure flowshop* dan *general flow shop*. Pada aliran pekerjaan *pure flowshop* setiap job memiliki satu operasi pada setiap mesin. Sedangkan pada *general flow shop* suatu pekerjaan dimungkinkan terdiri kurang dari *m* operasi dengan operasi-operasi pada mesin-mesin yang tidak berdekatan (bersebelahan) dan operasi terakhir tidak selalu dimulai pada mesin 1 dan diakhiri pada mesin *m*.



Gambar 2.4 Aliran *Pure Flowshop*



Gambar 2.5 Aliran General *Flowshop*

Karakteristik dasar penjadwalan flowshop adalah sebagai berikut :

a. Terdapat *n* job yang tersedia dan siap diproses pada waktu t = 0

b. Waktu *set up independent* terhadap urutan pengerjaan.

c. Terdapat *m* mesin berbeda yang tersedia secara *continue*.

d. Operasi-operasi individual tidak dapat dipecah-pecah.

2.3 Algoritma Pour

Hamid Davoud Pour (2001) mengembangkan algoritma heuristik baru di dalam menyelesaikan penjadwalan flowshop dengan tujuan meminalkan *makespan* yaitu berdasarkan pendekatan kombinasi. Hal ini dilakukan dengan cara mengganti setiap job dengan job yang lainnya dalam urutan sampai ditemukan kombinasi urutan yang dapat memenuhi kriteria tujuan. Dalam metode ini diasumsikan bahwa semua job diproses secara terpisah dan *independent* untuk setiap mesinnya. Berikut adalah notasi yang digunakan:

1. P_{ij} = waktu proses dari job *i* pada mesin *j*.
2. C_{ij} = rentang waktu antara saat job *i* pada mesin *j* dimulai (*t=0*) sampai job itu selesai.
3. C_i = *sum of completion time* untuk job i pada semua mesin.
4. *Fmax* = rentang waktu antara saat pekerjaan tersedia atau dapat dimulai sampai pekerjaan itu selesai (*makespan*).

Langkah-langkah pengerjaan Algoritma Pour:

a. Memilih job secara acak sebagai urutan pertama sementara dalam urutan pengerjaan.
b. Menempatkan job-job lain (selain job yang sudah dipilih sebagai urutan pertama) pada urutan berikutnya.
c. Memilih waktu proses terkecil untuk masing-masing mesin.
d. Melakukan penambahan waktu proses secara increasing time pada *Pij* yang lain, selain P_{ij} paling minimal yang terpilih sebelumnya.
e. Menghitung *sum of completion time* (C_i) atau jumlah waktu dimana setiap pekerjaan dianggap selesai.

f. Mengurutkan C_i dengan aturan *increasing order* untuk diletakkan pada urutan setelah job yang sudah dipilih untuk urutan pertama sementara.
g. Setelah diperoleh urutan sementara, maka hitunglah *Fmax*-nya.
h. Lakukan ulang langkah pada poin a sampai g untuk setiap *job* yang ada sampai diperoleh *Fmax* paling minimal, yang akan ditempatkan sebagai urutan pertama dari urutan *job.*
i. Lakukan ulang langkah pada poin a sampai h semua job berada pada urutan pengerjaan.

2.4 Algoritma NEH

Metode ini dikembangkan oleh Nawaz, Enscore dan Ham pada tahun 1983. Metode ini juga disebut metode *Incremental Construction Algorithms*. Algoritma NEH dapat ditunjukkan melalui langkah-langkah berikut ini :

Langkah 1:

a. Jumlahkan waktu proses setiap job.
b. Urutkan job-job menurut waktu prosesnya dimulai dari yang terbesar hingga terkecil.
c. Hasil urutan ini disebut sebagai daftar pengurutan job.

Langkah 2:

a. Ambil job yang menempati urutan pertama dan kedua pada daftar pengurutan job-job.
b. Buat dua alternatif calon urutan parsial baru.
c. Hitung *makespan* parsial dan *mean flow time* parsial dari calon urutan parsial baru.
d. Pilih calon urutan parsial baru yang memiliki *makespan* parsial yang terkecil. Jika ada urutan calon parsial baru yang memiliki makespan parsial terkecil yang sama, pilihlah calon urutan parsial yang baru tadi yang memiliki *mean flow time* parsial yang lebih kecil. Jika masih sama, maka pilihlah calon urutan parsial baru tadi secara acak.
e. Calon urutan parsial yang baru yang terpilih menjadi urutan parsial baru.
f. Coret job-job yang diambil dari daftar pengurutan job-job.

g. Periksa apakah k = n (n adalah jumlah job yang ada). Jika ya, lanjutkan ke langkah 4. Jika tidak lanjutkan ke langkah 3.

Langkah 3:

a. Ambil job yang menempati urutan pertama dari daftar pengurutan job-job.
b. Hasilkan sebanyak *k* calon urutan parsial baru dengan memasukkan job yang diambil ke dalam slot urutan parsial sebelumnya.
c. Hitung *makespan* parsial dan *mean flow time* parsial dari calon urutan parsial baru.
d. Pilih calon urutan parsial baru yang memiliki *makespan* parsial yang terkecil. Jika ada urutan calon parsial baru yang memiliki makespan parsial terkecil yang sama, pilihlah calon urutan parsial yang baru tadi yang memiliki *mean flow time* parsial yang lebih kecil. Jika masih sama, maka pilihlah calon urutan parsial baru tadi secara acak.
e. Calon urutan parsial yang baru yang terpilih menjadi urutan parsial baru.
f. Coret job-job yang diambil dari daftar pengurutan job-job.
g. Periksa apakah k = n (n adalah jumlah job yang ada). Jika ya, lanjutkan ke langkah 4. Jika tidak lanjutkan ke langkah 3.

Langkah 4:

a. Urutan parsial baru menjadi urutan final.
b. Proses perhitungan berhenti.

2.4 Kompleksitas Algoritma

Algoritma memegang peranan penting dalam bidang pemrograman. Begitu pentingnya suatu algoritma, sehingga perlu dipahami konsep dasar algoritma. Apalagi untuk seorang programer, tentu diperlukan suatu algoritma sehingga dapat membuat program yang lebih efektif dan efisien. Bagi kebanyakan orang, algoritma sangat membantu dalam memahami konsep logika pemrograman. Algoritma adalah kumpulan instruksi yang dibuat secara jelas untuk menunjukan langkah-langkah penyelesaian suatu masalah. Sebuah algoritma tidak saja harus

benar tetapi juga harus mangkus (efisien). Algoritma yang mangkus ialah algoritma yang dapat meminimumkan kebutuhan waktu dan ruang.

Dalam bidang komputer, misalnya EDP (*Elektronik Data Processing*) atau MIS (*Management Information System*), algoritma sering dimanfaatkan untuk menyelesaikan suatu masalah atau untuk proses pengambilan keputusan. Seorang sistem analisis (*analisist system*) tentunya menggunakan algoritma untuk merancang suatu sistem. Bagi seorang programer, algoritma digunakan untuk membuat modul-modul program. Menurut Wibowo (2011), kemangkusan (efisiensi) sebuah algoritma dapat digunakan untuk menilai algoritma terbaik. Hal tersebut dapat ditentukan dengan menggunakan:

a. Kompleksitas waktu $T(n)$, merupakan banyaknya operasi yang dilakukan untuk menjalankan sebuah algoritma sebagai fungsi dari ukuran masukan n.
b. Kompleksitas ruang $S(n)$, merupakan besarnya ruang memori yang dibutuhkan algoritma sebagai fungsi dari ukuran masukan n.

Kedua besaran tersebut besifat independen terhadap spesifikasi komputer dan *compiler*. Operasi yang dihitung adalah operasi dasar, yaitu operasi yang mendasari algoritma misal operasi perbandingan elemen pada algoritma, pengurutan/pencarian, penjumlahan dan perkalian. Dengan asumsi bahwa setiap operasi dasar membutuhkan waktu konstan. Sehingga dengan menggunakan besaran kompleksitas waktu atau ruang algoritma, peneliti dapat menentukan laju peningkatan waktu atau ruang yang diperlukan algoritma dengan meningkatnya ukuran masukan n. Dalam penelitian ini, analisis algoritma yang akan digunakan untuk menentukan kemangkusan algoritma adalah kompleksitas waktu $T(n)$ dengan asumsi bahwa setiap operasi dasar yang dilakukan membutuhkan waktu yang konstan. Perhitungan kompleksitas waktu dapat dilakukan dengan menggunakan perhitungan kompleksitas waktu asimptotik.

Kompleksitas waktu asimptotik adalah perkiraan kebutuhan waktu algoritma sejalan dengan meningkatnya nilai n. Pada umumnya, algoritma menghasilkan laju waktu yang semakin lama bila ukuran input n semakin besar. Salah satu cara untuk menghitung kompleksitas waktu dari suatu algoritma adalah

dengan menghitung kompleksitas waktu asimptotik dengan menggunakan notasi *Big-O (O)*.

Dalam analisis sebuah algoritma biasanya yang dijadikan ukuran adalah operasi aljabar seperti penjumlahan, pengurangan, perkalian dan pembagian, proses pengulangan (looping/Iterasi), proses pengurutan (sorting) dan proses pencarian (searching).

Notasi *Big* O sangat berguna di dalam menganalisa efisiensi dari suatu algoritma. Tinjau perbandingan $T(n) = 2n^2 + 2n + 2$ dengan n pada Tabel 2.2 berikut.

Tabel 2.2 Perbandingan pertumbuhan $T(n)$ dengan n^2

N	$T(n) = 2n^2 + 2n + 2$	n^2
10	222	100
100	20.202	10.000
1.000	2.002.002	1.000.000
10.000	200.020.002	100.000.000

Untuk n yang besar, pertumbuhan $T(n)$ sebanding dengan n^2. Pada kasus diatas laju pertumbuhan $T(n)$ adalah sama seperti laju pertumbuhan n^2. $T(n)$ bertambah seperti n^2 pada saat n bertambah. Maka dapat dikatakan bahwa $T(n)$ berorde n^2 atau $T(n) = O(n^2)$.

Notasi *Big-O* digunakan untuk menentukan kompleksitas suatu algoritma dengan melihat waktu tempuh algoritma. Selain itu, notasi *Big-O* juga berguna untuk membandingkan beberapa algoritma untuk masalah yang sama sehingga dapat menentukan algoritma terbaik.

Contoh:

Untuk masalah pengurutan memiliki banyak algoritma penyelesaian,

Selection sort, insertion sort → $T(n) = O(n^2)$

Quicksort → $T(n) = O(n \log n)$

Karena $n \log n < n^2$ untuk n yang besar, maka algoritma *Quicksort* dapat dikatakan lebih cepat dibandingkan dengan algoritma *Selection sort* dan *Insertion sort.*

Dari contoh tersebut dapat terlihat bahwa algoritma dengan orde *big-O* yang lebih kecil merupakan algoritma yang lebih efisien. Artinya, semakin kecil orde dari suatu algoritma maka algoritma tersebut akan semakin efisien (Wibowo, 2001).

Menurut definisi:

$T(n) = O(f(n))$ jika terdapat konstanta C dan n_0 sedemikian sehingga

$T(n) \leq C(f(n))$.

Artinya, $T(n)$ adalah orde paling besar dari $f(n)$. Misalkan sebuah algoritma memiliki kompleksitas waktu $O(f(n))$, ini berarti untuk n yang besar maka algoritma akan berhenti setelah melakukan operasi dasar paling banyak sebesar konstanta dikalikan $f(n)$. Jadi algoritma membutuhkan konstanta kali $f(n)$ unit waktu (Wibowo, 2001).

Contoh:

$T(n) = 2n^2 + 2n + 2 = O(n^2)$

Penyelesaiannya:

$2n^2 + 2n + 2 = O(n^2)$

karena

$2n^2 + 2n + 2 \leq 2n^2 + 2n^2 + 2n^2 = 6n^2$ untuk semua $n \geq 1$ ($C = 6$).

BAB III. METODOLOGI PENELITIAN

3.1 Data Penelitian

Data yang digunakan dalam penelitian ini adalah data dari PT. Kertas Leces Probolinggo pada Juni tahun 2005 yang berkaitan dengan permasalahan yang akan dipecahkan yaitu berupa jumlah mesin yang digunakan dalam operasi, jumlah job yang dikerjakan dan waktu proses setiap job pada setiap mesin yang ada. Data tersebut disajikan dalam tabel 3.1 berikut.

Tabel 3.1 waktu proses setiap job pada setiap mesin (dalam satuan menit)

	M1	M2	M3	M4	M5
Job 1	360	70	45	40	85
Job 2	300	60	35	30	55
Job 3	360	80	60	70	90
Job 4	420	100	70	90	100

Keterangan :

Job 1 : Kertas HVS 45 *gsm* dengan jumlah produksi sebanyak 120 ton/hari

Job 2 : Kertas MG *paper* dengan jumlah produksi sebanyak 50 ton/hari

Job 3 : Kertas HVS 60 *gsm* dengan jumlah produksi sebanyak 140 ton/hari

Job 4 : Kertas BC Super dengan jumlah produksi sebanyak 70 ton/hari

3.2 Langkah-Langkah Penyelesaian

Secara umum dalam membangun aplikasi penjadwalan produksi flowshop ini dilakukan melalui tahap-tahap sebagai berikut:

1. mengambil data sekunder yang diperoleh menjadi data urutan mesin dan waktu pemrosesan. Data tersebut terdiri dari 4 jenis kertas yang berbeda dan melalui 5 tahapan mesin produksi. Lima jenis kertas tersebut adalah

HVS 45 *gsm*, MG *paper*, HVS 60 *gsm* dan BC Super. Sedangkan proses pembuatan kertas tersebut melalui 5 tahapan mesin, yaitu *stock preparation*, *wire part*, *press part*, *drier part*, dan *finishing*.

2. membuat penjadwalan produksi mesin dengan algoritma Pour dan algoritma NEH.
3. membandingkan *makespan* dan jumlah langkah (performa) dari algoritma Pour dan NEH.
4. membuat program penjadwalan produksi mesin menggunakan algoritma Pour dan NEH dengan bantuan bahasa pemrograman matlab.

BAB 4. HASIL DAN PEMBAHASAN

Pada bab ini akan dibahas masalah penjadwalan *flowshop* pada industri PT. Kertas Leces Probolinggo dengan algoritma Pour dan NEH. Hal ini dilakukan karena selama ini PT. Kertas Leces Probolinggomemproduksi beberapa jenis kertas dengan menggunakan aturan *First Come First Serve* (FCFS) tanpa memperhatikan lama atau singkatnya waktu proses produksi. Produk yang diteliti terdiri dari 4 jenis kertas yang berbeda dan melalui 5 tahapan mesin produksi. Lima jenis kertas tersebut adalah HVS 45 *gsm*, MG *paper*, HVS 60 *gsm* dan BC Super. Sedangkan proses pembuatan kertas tersebut melalui 5 tahapan mesin, yiatu *stock preparation*, *wirepart*, *presspart*, *drierpart*, dan *finishing*. Penjadwalan kali ini dilakukan dengan bantuan program Matlab.

4.1 Hasil

Penjadwalan jenis *flowshop* yang ada di perusahan tersebut menggunakan aturan *First Come First Serve* (FCFS). Berdasarkan data pada Tabel 3.1, maka jadwal yang digunakan sebagai solusi dengan metode FCFSuntuk memproduksi kertas memiliki urutan jobyakni Job 1 – Job 2 – Job 3 – Job 4, dengan perhitungan *makespan* pada Tabel 4.1.

Tabel 4.1 Perhitungan *makespan* dengan metode FCFS

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 1	0	360	360	430	430	475	475	515	515	600
Job 2	360	660	660	720	720	755	755	785	785	830
Job 3	660	1020	1020	1100	1100	1160	1160	1230	1230	1320
Job 4	1020	1440	1440	1540	1540	1610	1610	1700	1700	1800

Keterangan:

S : start atau mulai

E : end atau berhenti

Dari Tabel 4.1 dapat diketahui bahwa nilai *makespan* dengan metode *FCFS* adalah 1.800 menit. Artinya dengan jadwal tersebut, industri PT. Kertas Leces membutuhkan waktu 1.800 menit dalam proses produksinya.

4.1.1 Penjadwalan Produksi dengan Algoritma Pour

Berdasarkan tahapan perhitungan algoritma Pour yang telah dijelaskan pada subbab 2.3, maka penjadwalan produksi kertas adalah sebagai berikut:

a. Menentukan Urutan Job Pertama

Dalam algoritma Pour semua job pertama kali ditempatkan sebagai urutan pertama. Setelah itu dilakukan perhitungan hingga *makespan* diketahui sesuai tahapan algoritma tersebut. Hasil *makespan* yang diperoleh dari masing-masing job yang telah menempati urutan pertama dibandingkan. Nilai *makespan* yang terkecil diambil sebagai job yang menempati urutan pertama. Dalam hal ini diperoleh Job 4 sebagai urutan pertama karena memiliki *makespan* terkecil yakni 1.740 menit dibandingkan dengan Job 1, Job 2 dan Job 3 yakni 1.800 menit pada saat menempati urutan pertama. Hasil perhitungan *makespan* pada saat Job 4 menempati urutan pertama dapat ditunjukkan oleh Tabel 4.2.

Tabel 4.2 Perhitungan *makespan* untuk Job 4 menempati urutan pertama

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 4	0	420	420	520	520	590	590	680	680	780
Job 2	420	720	720	780	780	815	815	845	845	900
Job 1	720	1080	1080	1150	1150	1195	1195	1235	1235	1320
Job 3	1080	1440	1440	1520	1520	1580	1580	1650	1650	1740

b. Menentukan Urutan Job Kedua

Pada langkah ini Job 4 tidak diperlukan untuk menentukan urutan job yang menempati urutan kedua karena Job 4 sudah terpilih untuk menempati urutan

pertama. Setelah itu dilakukan perhitungan hingga *makespan* diketahui sesuai tahapan algoritma tersebut. Hasil *makespan* yang diperoleh dari masing-masing job yang telah menempati urutan pertama pada langkah ini dibandingkan. Nilai *makespan* yang terkecil diambil sebagai job yang menempati urutan kedua. Dalam hal ini diperoleh Job 3 sebagai urutan pertama karena memiliki *makespan* terkecil yakni 1.200 menit dibandingkan dengan Job 1 yang memiliki *makespan* 1.420 menit dan Job 2 yakni 1.320 menit pada saat menempati urutan pertama. Hasil perhitungan *makespan* pada saat Job 3 menempati urutan pertama dalam tahap ini dapat ditunjukkan oleh Tabel 4.3.

Tabel 4.3 Perhitungan *makespan* untuk Job 3 menempati urutan kedua

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 3	0	360	360	440	440	500	500	570	570	630
Job 2	360	600	600	660	660	695	695	725	725	780
Job 1	600	960	960	1030	1030	1075	1075	1115	1115	1200

c. Menentukan Urutan Job Ketiga dan Keempat

Pada langkah ini Job 4 dan Job 3 tidak diperlukan untuk menentukan urutan job yang menempati urutan ketiga dan keempat karena Job 4 dan Job 3 sudah terpilih untuk menempati urutan pertama dan kedua. Setelah itu, dilakukan perhitungan hingga *makespan* diketahui sesuai tahapan algoritma tersebut. Hasil *makespan* yang diperoleh dari masing-masing jobyang telah menempati urutan pertama pada langkah ini dibandingkan. Nilai *makespan* yang terkecil diambil sebagai job yang menempati urutan ketiga dan keempat. Dalam hal ini diperoleh Job 1 sebagai urutan ketiga karena memiliki *makespan* terkecil yakni 830 menit dibandingkan dengan Job 2 yang memiliki makespan 860 menit pada saat menempati urutan pertama. Hasil perhitungan *makespan* pada saat Job 3 menempati urutan pertama dalam tahap ini dapat ditunjukkan oleh Tabel 4.4.

Tabel 4.4 Perhitungan *makespan* untuk Job 1 menempati urutan ketiga

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 1	0	360	360	430	430	475	475	515	515	600
Job 2	360	660	660	720	720	755	755	785	785	830

Berdasarkan perhitungan algoritma Pour di atas diperoleh penjadwalan produksi kertas dengan urutan Job 4 – Job 3 – Job 1 – Job 2 dengan *makespan* sebesar 1.620 menit. Hasil perhitungan *makespan* berdasarkan urutan job yang diperoleh ditunjukkan pada Tabel 4.5.

Tabel 4.5 Perhitungan *makespan* untuk urutan Job 4 – Job 3 – Job 1 – Job 2

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 4	0	420	420	520	520	590	590	680	680	780
Job 3	420	780	780	960	960	1020	1020	1090	1090	1170
Job 1	780	1140	1140	1210	1210	1250	1250	1295	1295	1380
Job 2	1140	1440	1440	1500	1500	1535	1535	1565	1565	1620

4.1.2 Penjadwalan Produksi dengan Algoritma NEH

Berdasarkan tahapan perhitungan algoritma NEH yang telah dijelaskan pada subbab 2.4, maka penjadwalan produksi kertas adalah sebagai berikut:

a. Menentukan Alternatif Urutan Parsial dari 2 Job

Berdasarkan data pada tabel 3.1, langkah pertama yang harus dilakukan dalam perhitungan algoritma NEH adalah menentukan total waktu proses yang dibutuhkan tiap job dalam 5 mesin. Selanjutnya, total waktu

proses yang diperoleh menjadi acuan untuk membentuk urutan Job dimulai dari total waktu proses terbesar hingga terkecil. Dalam hal ini diperoleh urutan Job 4 – Job 3 – Job 1 – Job 2 karena Job 4 memiliki total waktu prosesterbesar dibandingkan dengan job yang lain. Hal ini bisa dilihat pada Tabel 4.6.

Tabel 4.6 Urutan job mulai dari total waktu proses terbesar

	M1	M2	M3	M4	M5	Total
Job 4	420	100	70	90	100	780
Job 3	360	80	60	70	90	660
Job 1	360	70	45	40	85	600
Job 2	300	60	35	30	55	480

Selanjutnya, dari hasil perhitungan tersebut akan dihitung *makespan* untuk urutan parsial dari 2 job yang memiliki total waktu proses terbesar. Dalam hal ini, job yang terpilih adalah Job 4 dan Job 3 karena memiliki total waktu proses terbesar dibandingkan dengan Job 1 dan Job 2. Sehingga diperoleh urutan alternatif dari 2 job terpilih yakni Job 4 – Job 3 atau Job 3 – Job 4. Melalui perhitungan algoritma NEH diperoleh *makespan* terkecil yakni 1.080 menit dengan urutan parsial Job 4 – Job 3 dibandingkan dengan Job 3 – Job 4 yang memiliki *makespan* 1.140 menit. Perhitungan *makespan* dari urutan parsial terpilih dapat ditunjukkan pada Tabel 4.7.

Tabel 4.7 Perhitungan *makespan* untuk urutan parsial Job 4 – Job 3

M	Job	Durasi	Mulai	Siap
1	Job 4	420	0	420
1	Job 3	360	420	780
2	Job 4	100	420	520
2	Job 3	80	780	860
3	Job 4	70	520	590
3	Job 3	60	860	920
4	Job 4	90	590	680

4	Job 3	70	920	990
5	Job 4	100	680	780
5	Job 3	90	990	1080

b. Menentukan Alternatif Urutan Parsial dari 3 Job

Setelah alternatif urutan Job 4 – Job 3 terpilih, langkah selanjutnya adalah menentukan alternatif urutan dari 3 job dengan mengikutsertakan urutan parsial yang terpilih sebelumnya. Dalam hal ini diperoleh beberapa kemungkinan alternatif urutan job jika urutan parsial Job 4 – Job 3 dihubungkan dengan Job 1 yakni Job 4 – Job 3 – Job 1 atau Job 1 – Job 3 – Job 4 atau Job 4 – Job 1 – Job 3. Apabila alternatif urutan parsial Job 4 – Job 3 dihubungkan dengan Job 2, maka urutan alternatif yang diperoleh adalah Job 4 – Job 3 – Job 2 atau Job 2 – Job 4 – Job 3 atau Job 4 – Job 2 – Job 3.

Melalui perhitungan algoritma NEH diperoleh *makespan* terkecil yakni 1.260 menit dengan urutan parsial Job 4 – Job 3 – Job 2 dibandingkan dengan urutan alternatif yang lain. Perhitungan *makespan* dari urutan parsial terpilih dapat ditunjukkan pada Tabel 4.8.

Tabel 4.8 Penjadwalan parsial Job 4 – Job 3 – Job 2

M	Job	Durasi	Mulai	Siap
1	Job 4	420	0	420
1	Job 3	360	420	780
1	Job 2	300	780	1080
2	Job 4	100	420	520
2	Job 3	80	780	860
2	Job 2	60	1080	1140
3	Job 4	70	520	590
3	Job 3	60	860	920
3	Job 2	35	1140	1175
4	Job 4	90	590	680

4	Job 3	70	920	990
4	Job 2	30	1175	1205
5	Job 4	100	680	780
5	Job 3	90	990	1080
5	Job 2	55	1205	1260

c. Menentukan Alternatif Urutan dari 4 Job

Setelah alternatif urutan Job 4 – Job 3 – Job 2 terpilih, langkah selanjutnya adalah menentukan alternatif urutan dari 4 job dengan mengikutsertakan urutan parsial yang terpilih sebelumnya. Dalam hal ini diperoleh beberapa kemungkinan alternatif urutan job jika urutan parsial Job 4 – Job 3 – Job 2 dihubungkan dengan Job 1 yakni Job 4 – Job 3 – Job 2 – Job 1 atau Job 4 – Job 3 – Job 1 – Job 2 atau Job 4 – Job 1 – Job 3 – Job 2 atau Job 1 – Job 4 – Job 3 – Job 2.

Melalui perhitungan algoritma NEH diperoleh *makespan* terkecil yang sama yakni 1.620 menit dengan beberapa urutan parsial yakni Job 1 – Job 4 – Job 3 – Job 2 atau Job 4 – Job 1 – Job 3 – Job 2 atau Job 4 – Job 3 – Job 1 – Job 2 dibandingkan dengan urutan alternatif Job 4 – Job 3 – Job 2 – Job 1 dengan *makespan* 1.680 menit. Perhitungan *makespan* dari salah satu urutan parsial terpilih dapat ditunjukkan pada Tabel 4.9.

Tabel 4.9 Penjadwalan parsial Job 1 – Job 4 – Job 3 – Job 2

M	Job	Durasi	Mulai	Siap
1	Job 1	360	0	360
1	Job 4	420	360	780
1	Job 3	360	780	1140
1	Job 2	300	1140	1440
2	Job 1	70	360	430
2	Job 4	100	780	880
2	Job 3	80	1140	1220
2	Job 2	60	1440	1500
3	Job 1	45	430	475

3	Job 4	70	880	950
3	Job 3	60	1220	1280
3	Job 2	35	1500	1535
4	Job 1	40	475	515
4	Job 4	90	950	1040
4	Job 3	70	1280	1350
4	Job 2	30	1535	1565
5	Job 1	85	515	600
5	Job 4	100	1040	1140
5	Job 3	90	1350	1440
5	Job 2	55	1565	1620

4.1.3 Penjadwalan *Flowshop* dengan Program Matlab

Pada skripsi ini, disertakan sebuah aplikasi penjadwalan *flowshop* dengan bantuan program Matlab. Aplikasi tersebut dibuat untuk mempercepat dan mempermudah dalam melakukan perhitungan menggunakan kedua algoritma. Gambar 4.1 merupakan tampilan awal dari program yang telah dibuat. Pada Gambar 4.1 terdapat beberapa kolom yang ditampilkan, yakni:

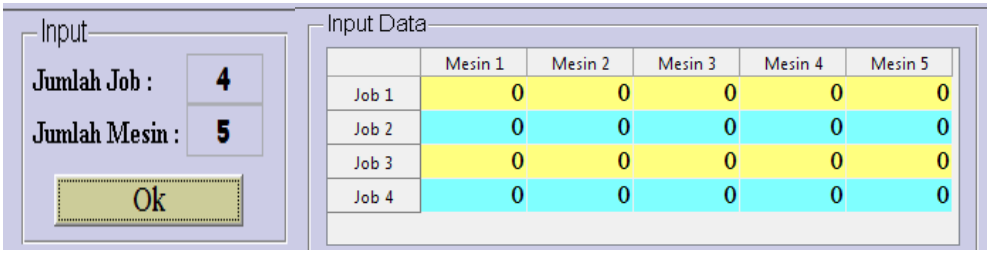
a. *Input*, yaitu tampilan awal program untuk memulai meng-*input* ukuran masalah yang akan diselesaikan oleh program.
b. Jenis Metode, yaitu tampilan metode yang akan melakukan perhitungan untuk menyelesaikan ukuran masalah pada kolom *input*. Dalam hal ini, ada dua metode pilihan yang disajikan yakni algoritma Pour dan algoritma NEH.
c. *Output*, yaitu tampilan penyelesaian masalah dalam satuan waktu dan urutan job setelah dilakukan perhitungan dengan algoritma Pour atau algoritma NEH.

Gambar 4.1 Tampilan Awal Aplikasi Penjadwalan *Flowshop*

Terdapat beberapa langkah yang harus dilakukan untuk menjalankan program aplikasi penjadwalan *flowshop*. Berikut ini penjelasan dari tiap langkah dalam menjalankan program.

1. *Input* Ukuran Masalah

Ukuran masalah penjadwalan *flowshop* dalam skripsi ini adalah 4 x 5, artinya terdiri dari empat job yang akan diproses pada lima mesin, sehingga jadwal yang terbentuk sebanyak 20 digit jadwal dengan satuan waktu yaitu menit. Pengguna harus meng-*input*terlebih dahulu jumlah job dan jumlah mesin yang akan diproses pada kolom "*input*" dan dilanjutkan dengan klik tombol "Ok" sehingga akan muncul tabel "Input Data" yang akan diisi oleh data tertentu untuk diselesaikan dengan program seperti pada Gambar 4.2.

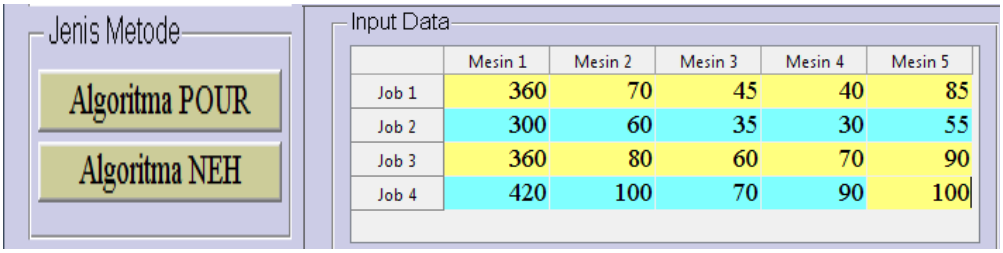


Gambar 4.2 Tampilan Tabel "Input Data"

Program aplikasi penjadwalan *flowshop* ini dibuat sesuai dengan langkah-langkah algoritma yang digunakan. Oleh karena itu, penulis membuat penyelesaian secara umum sehingga dapat menyelesaikan ukuran masalah *n* x *m*, artinya sejumlah *n* job dan *m* mesin.

2. Jenis Metode

Setelah muncul tabel "Input Data", pengguna dapat meng-*input* data tertentu yang akan diselesaikan dengan program sesuai dengan ukuran maslah yang telah ditentukan sebelumnya.

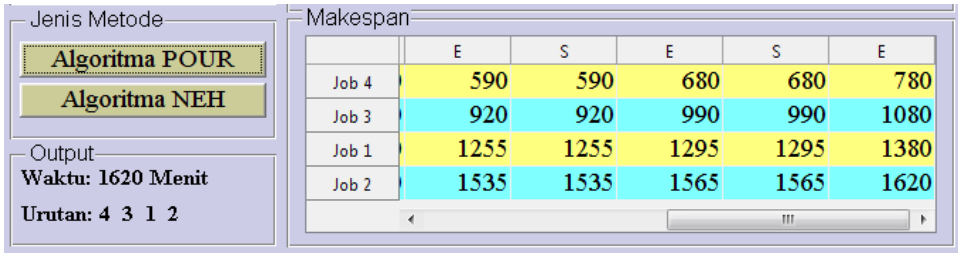


Gambar 4.3 Tampilan "Jenis Metode" dan Tabel "Input Data" yang Telah Diisi Data

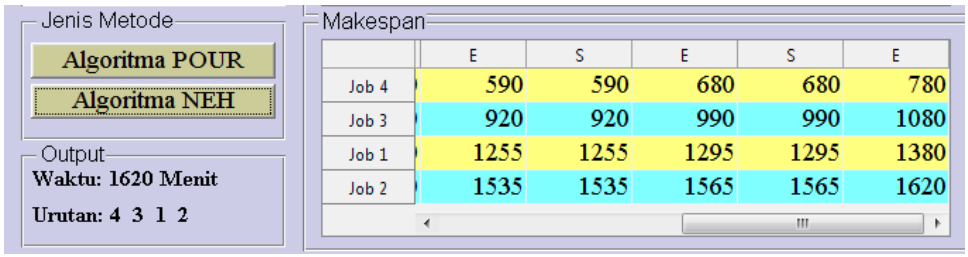
Pada Gambar 4.3 terdapat tombol pilihan jenis metode dari kedua algoritma yang akan digunakan untuk menyelesaikan perhitungan *makespan* dari data tersebut. Pada saat pengguna meng-*klik* tombol "Algoritma POUR", maka data yang telah di-*input* akan diproses sesuai dengan langkah-langkah algoritma Pour. Begitu juga sebaliknya, pada saat pengguna meng-*klik* tombol "Algoritma NEH", maka data yang telah di-*input* akan diproses sesuai dengan langkah-langkah algoritma NEH.

3. *Output* Program

Hasil akhir yang akan ditampilkan dalam perhitungan program ini ialah sebuah jadwal urutan job dengan *makespan* terkecil dari masing-masing algoritma dan akan muncul gambar *gannt chart*. Gambar 4.4 (a) dan Gambar 4.4 (b) merupakan hasil perhitungan *makespan* dengan menggunkan algoritma Pour dan algoritma NEH. Hal ini berarti, dari data yang telah di-*input* diperoleh jadwal untuk meminimumkan *makespan* menurut algoritma Pour dan algoritma NEH ialah Job 4 – Job 3 – Job 1 – Job 2 dengan nilai *makespan* sebesar 1.620 menit.



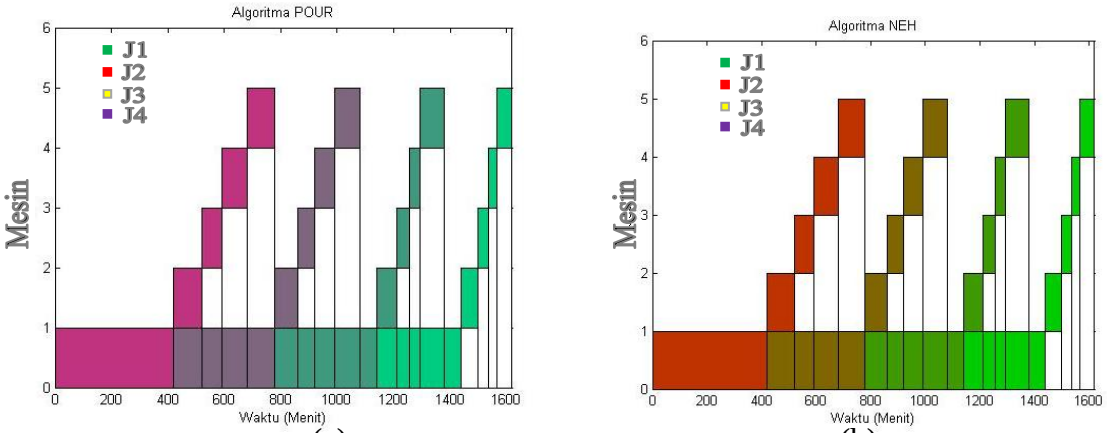
(a)



(b)

Gambar 4.4 Tampilan Hasil *Output makespan* dan Urutan Job

Sedangkan pada Gambar 4.5 (a) dan Gambar 4.5 (b) merupakan *Gantt Chart* dari hasil akhir perhitungan untuk algoritma Pour dan algoritma NEH. Urutan job dari masing-masing algoritma digambarkan dengan sumbu vertikal dan sumbu horisontal menggambarkan pergerakan waktu. Perbedaan warna yang digunakan pada *gantt chart* bertujuan untuk memudahkan pengguna untuk membedakan urutan mesin.

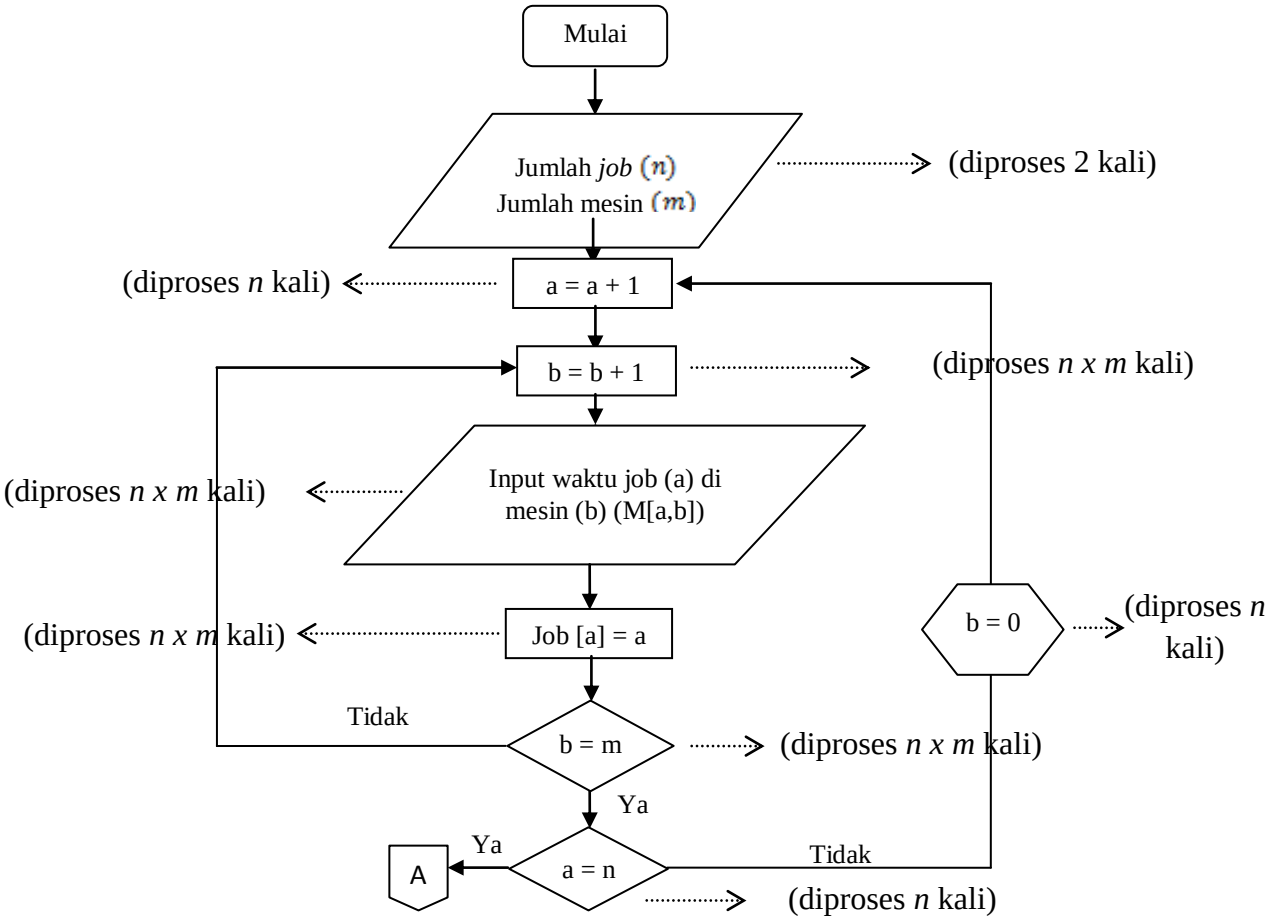


(a) (b)

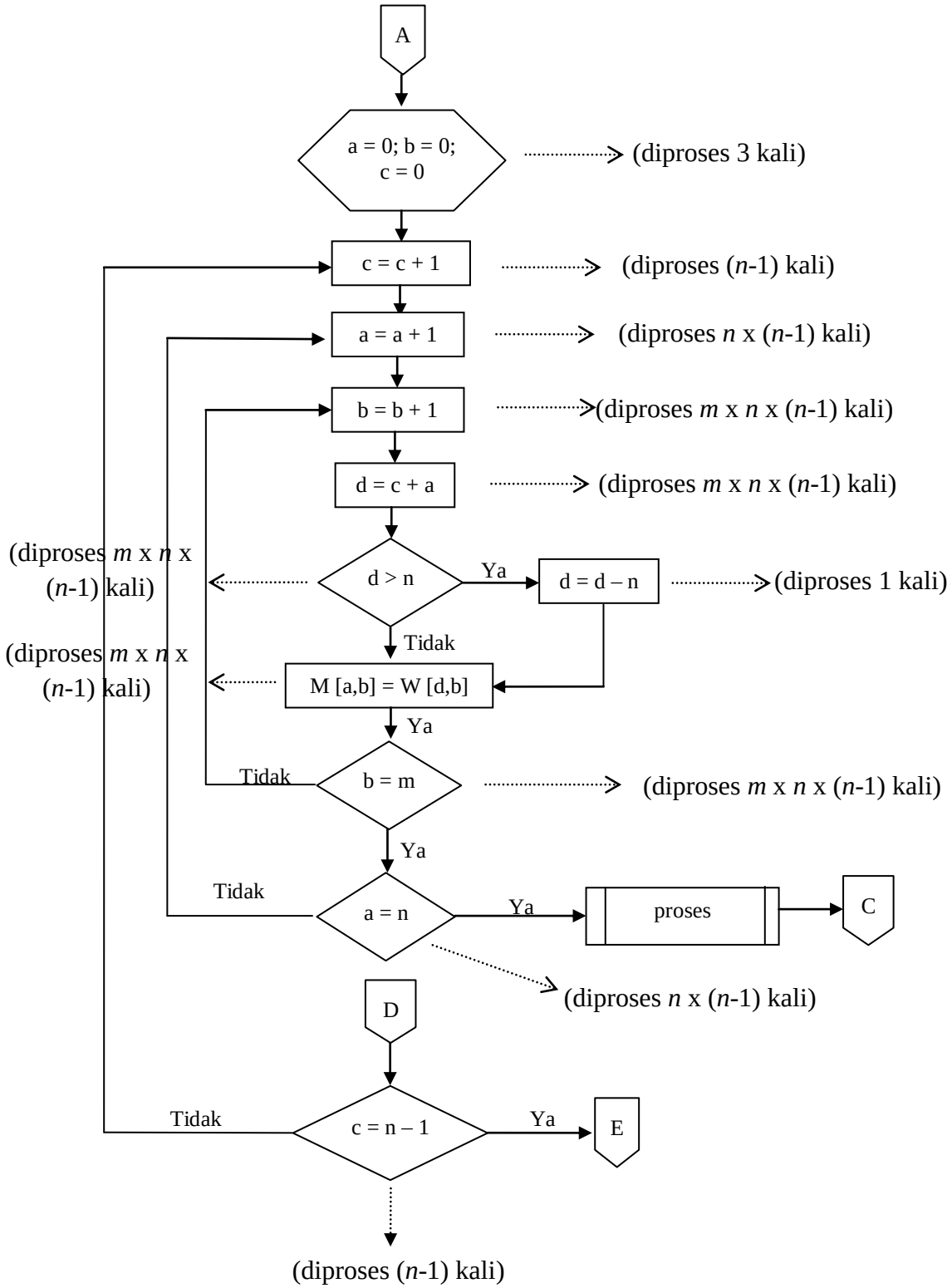
Gambar 4.5 Tampilan *Gantt Chart* Hasil Akhir Perhitungan

4.1.4 *Flowchart* Algoritma

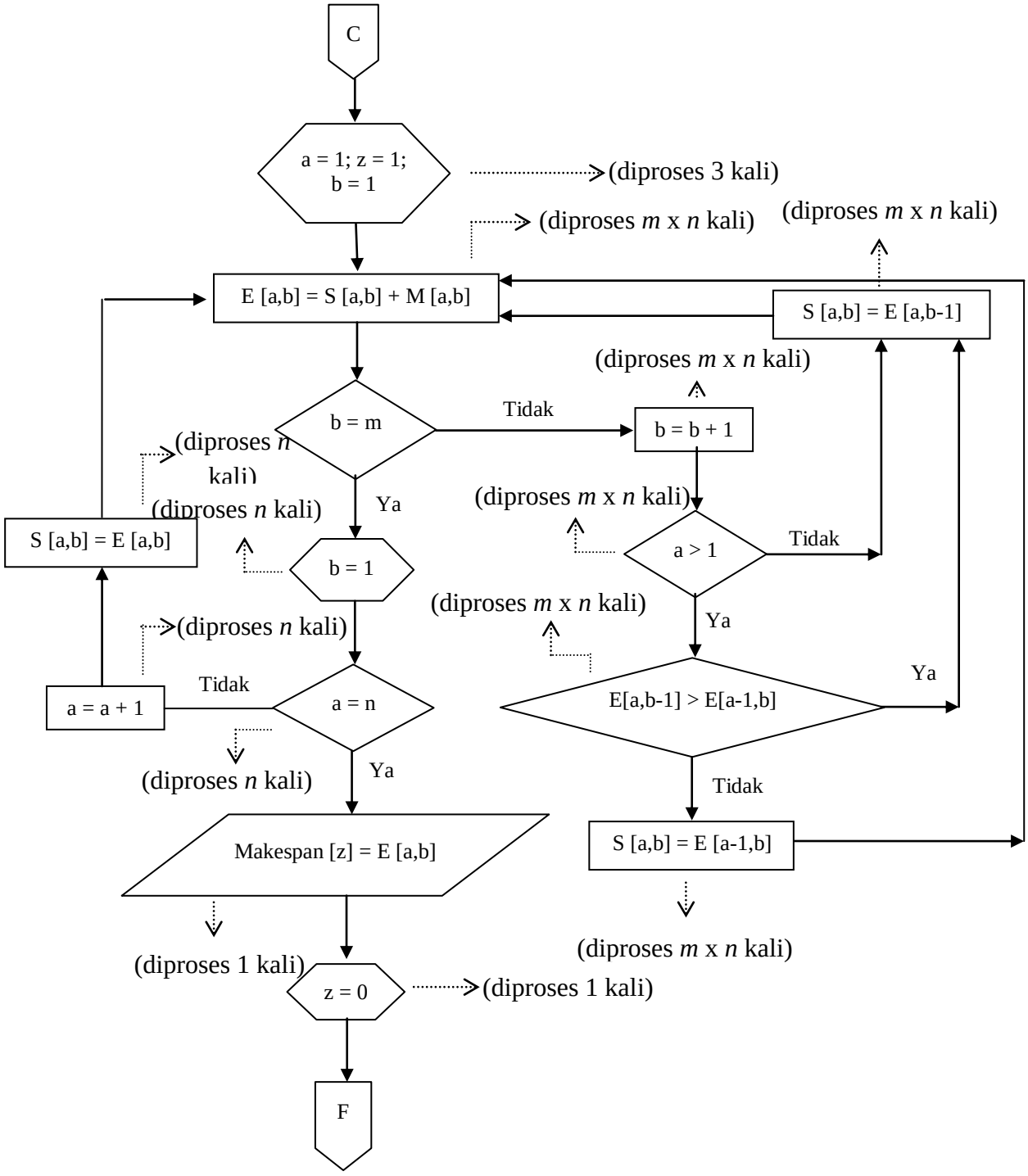
1. *Flowchart* Algoritma Pour



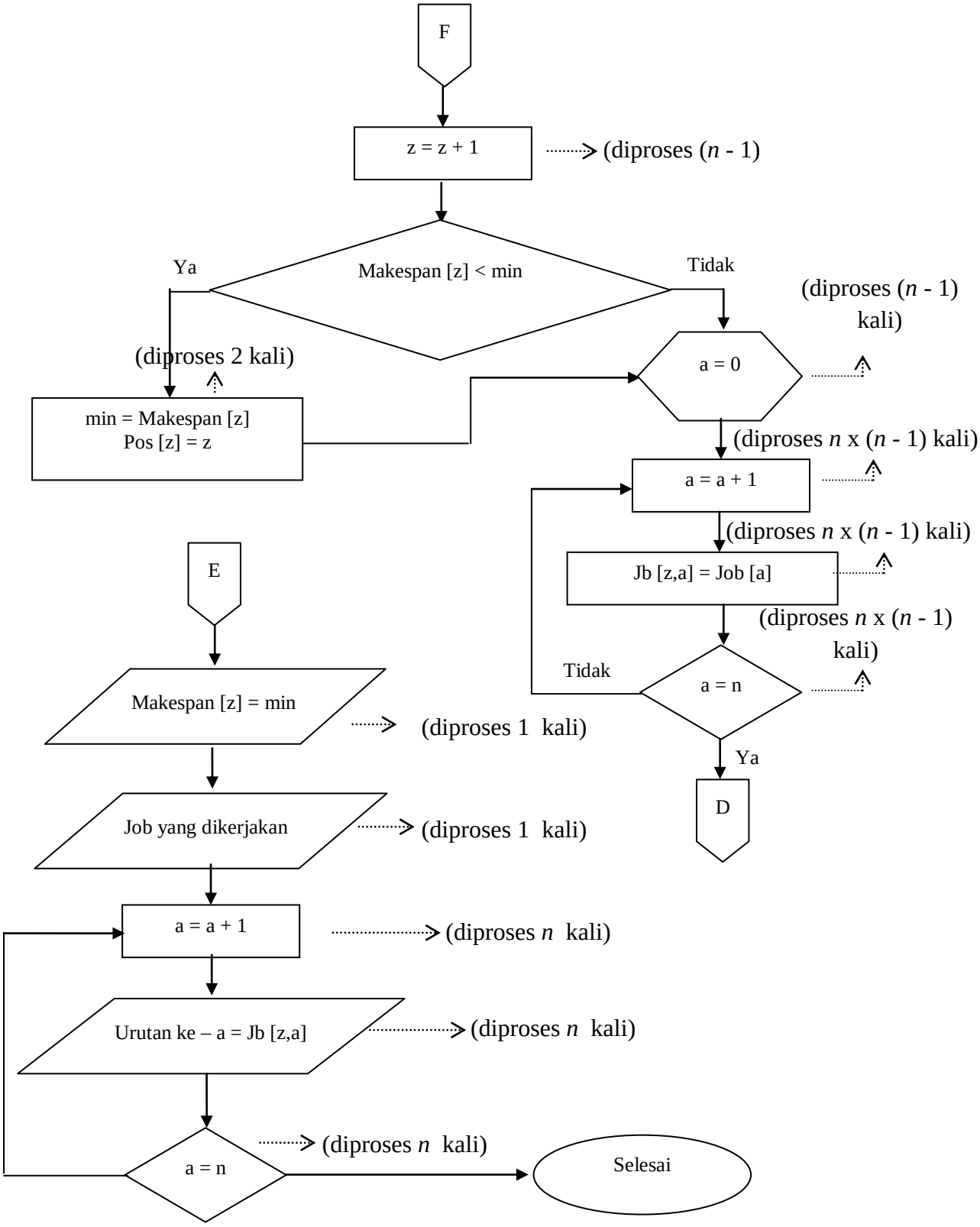
Jumlah perhitungannya adalah $T_1 = 2 + 4mn + 3n$



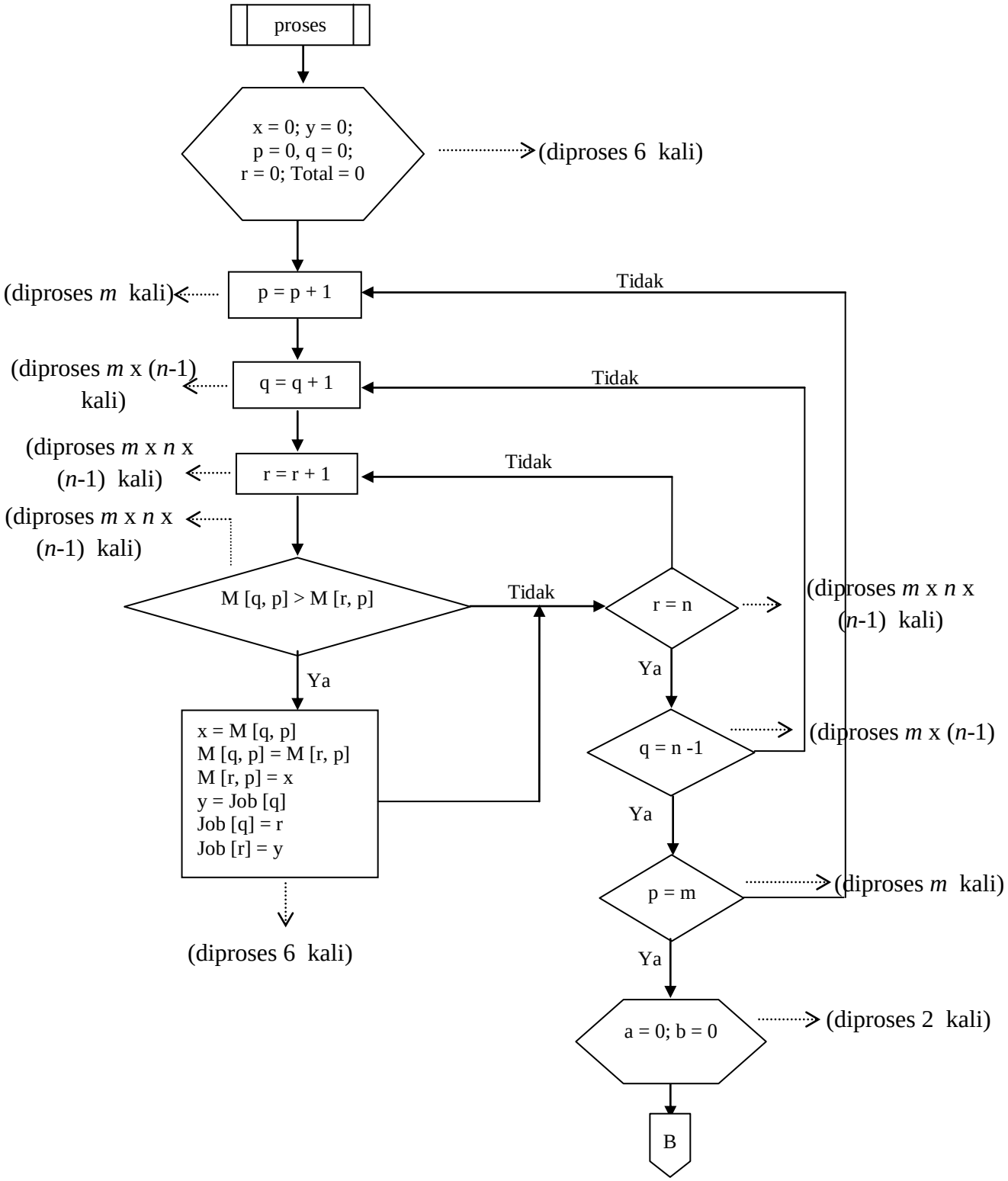
Jumlah perhitungannya adalah $T_2 = 4 + (n\text{-}1) + n\,(n\text{-}1) + 5\,mn\,(n\text{-}1)$



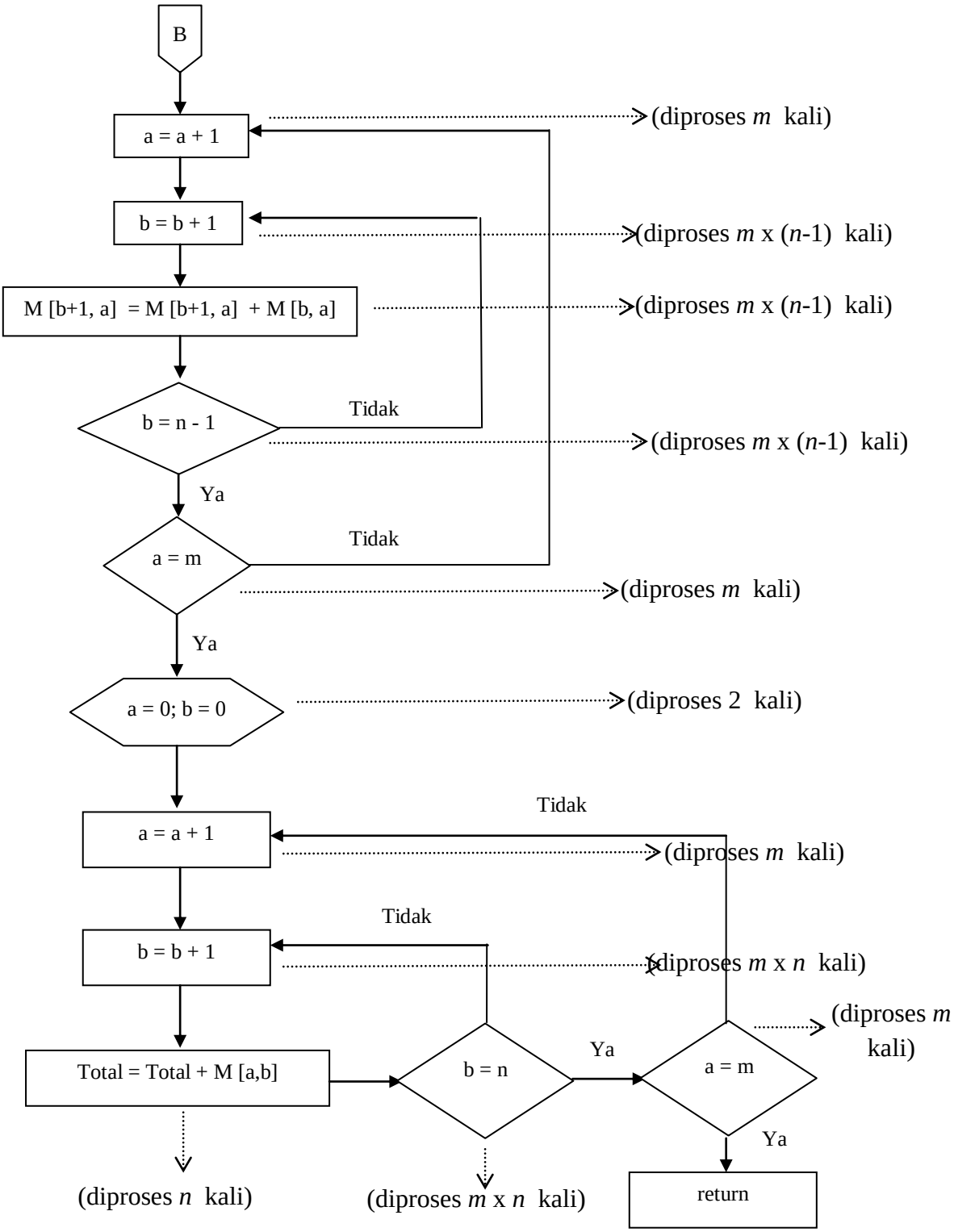
Jumlah perhitungannya adalah $T_3 = 5 + 4n + 7mn$



Jumlah perhitungannya adalah $T_4 = 4 + 3n + 3(n\text{-}1) + 4n(n\text{-}1)$



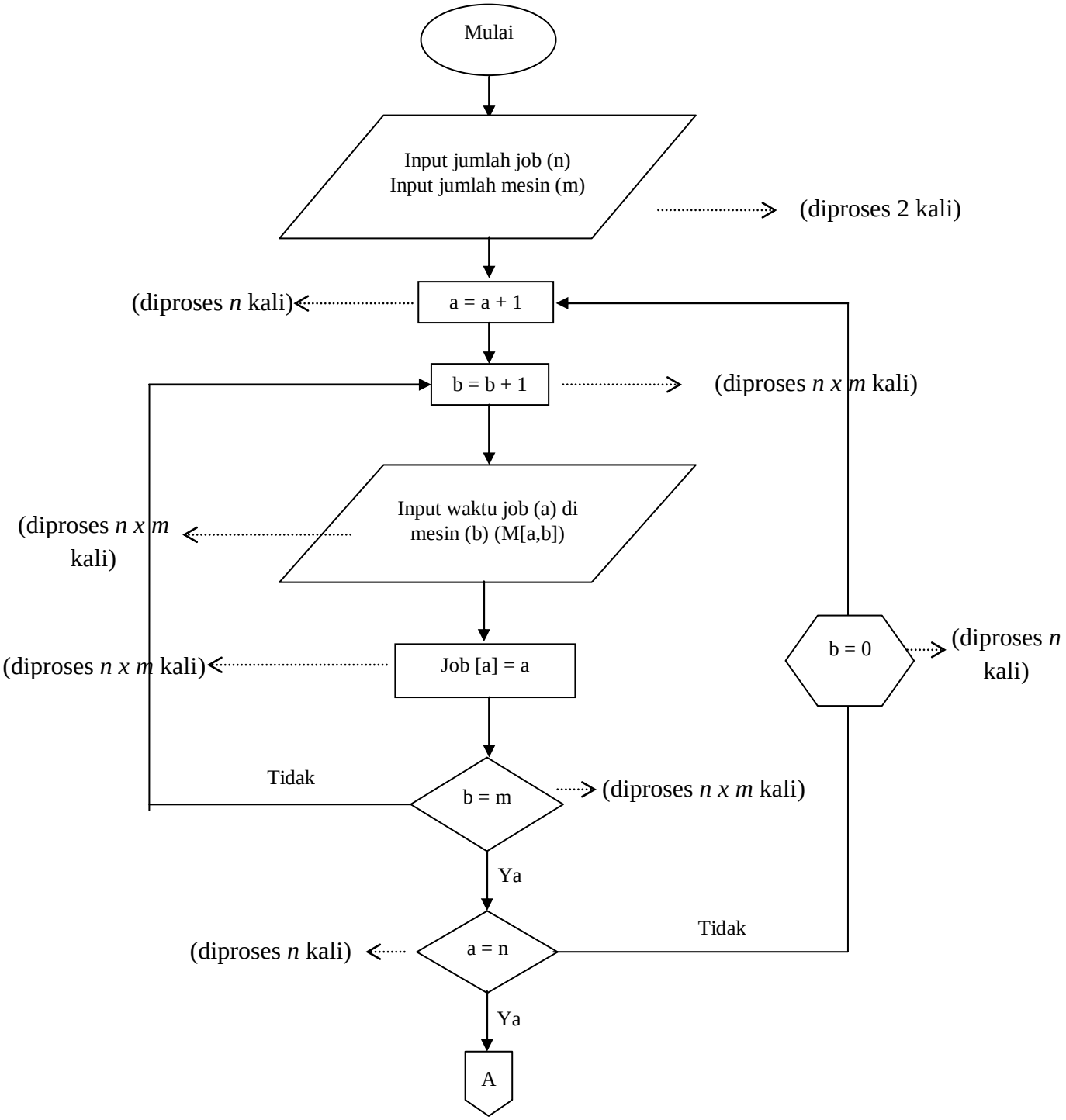
Jumlah perhitungannya adalah $T_5 = 14 + 2m + 2m(n\text{-}1) + 3mn(n\text{-}1)$



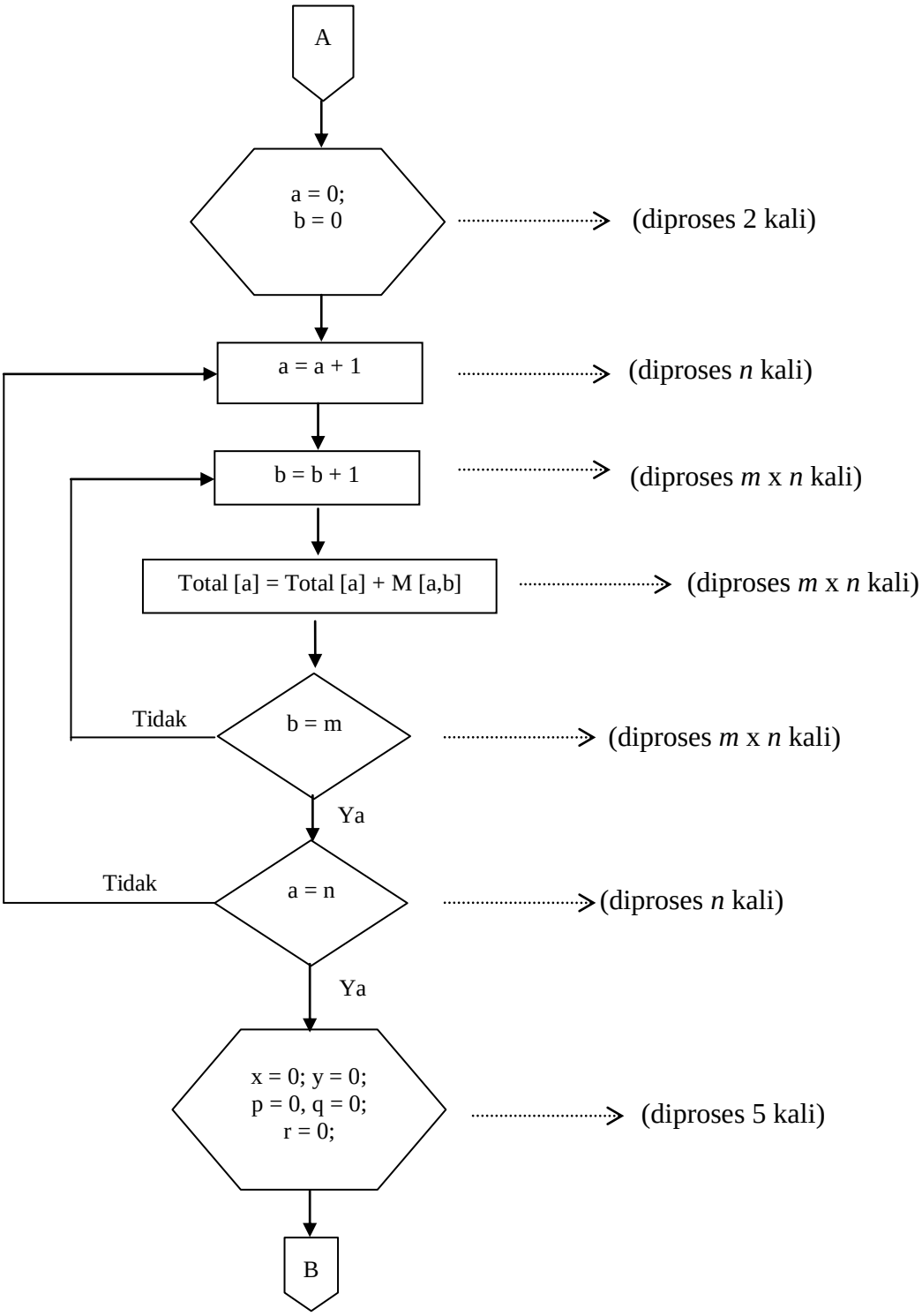
Gambar 4.6 *Flowchart* algoritma Pour dan jumlah

Jumlah perhitungannya adalah $T_6 = 2 + 4m + 3m(n\text{-}1) + 3mn$

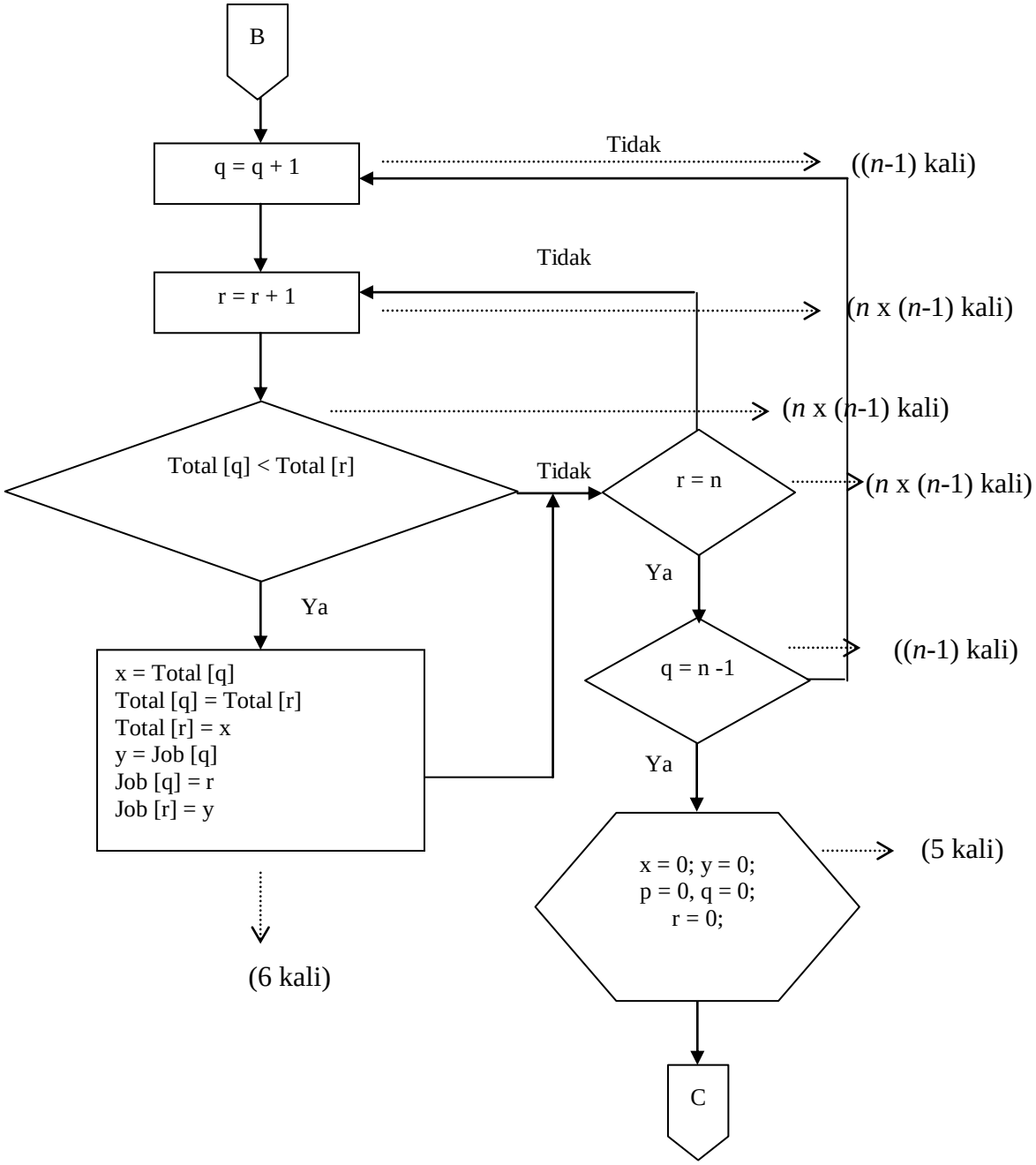
2. *Flowchart* Algoritma NEH



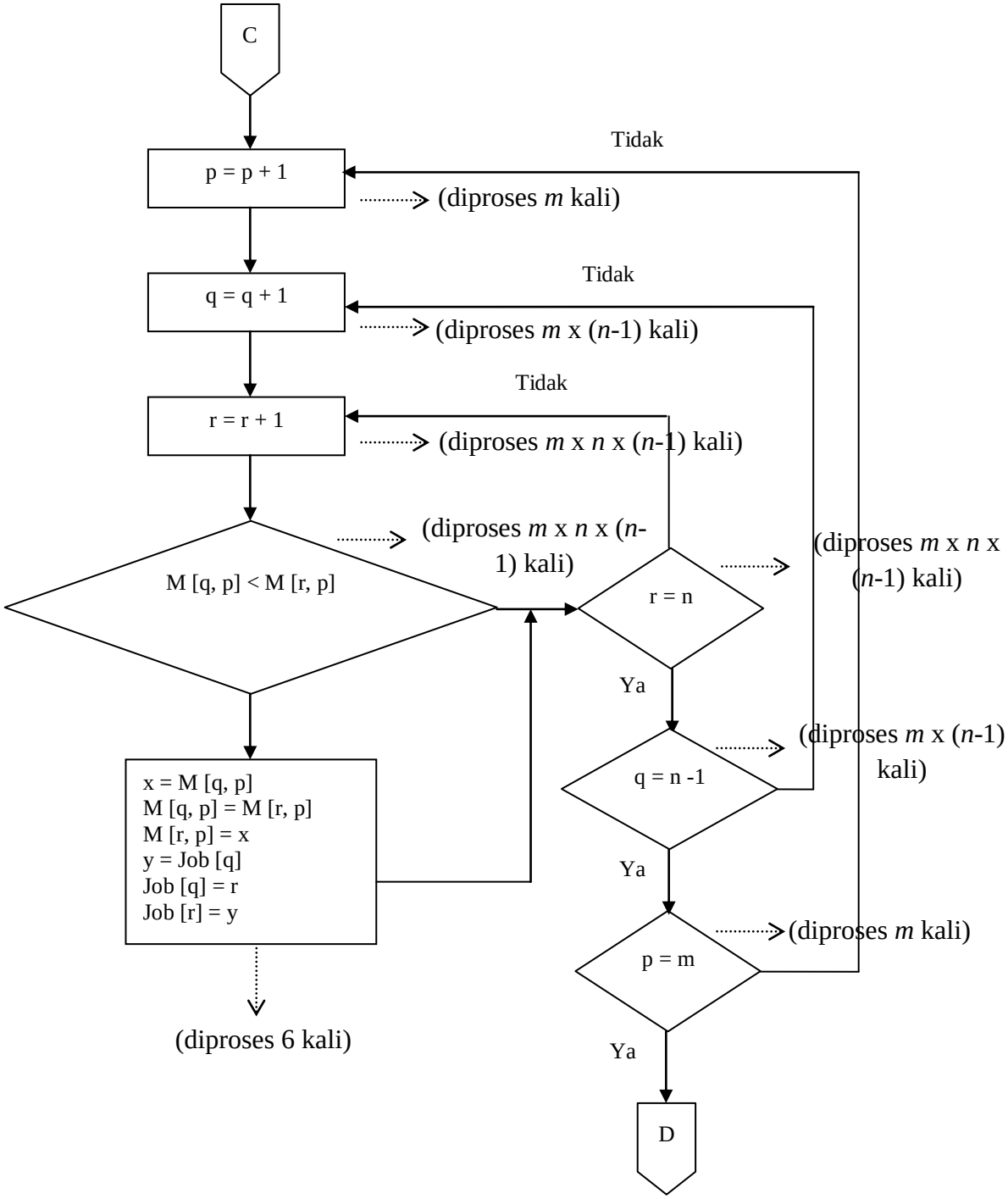
Jumlah perhitungannya adalah $T_1 = 2 + 3n + 4mn$



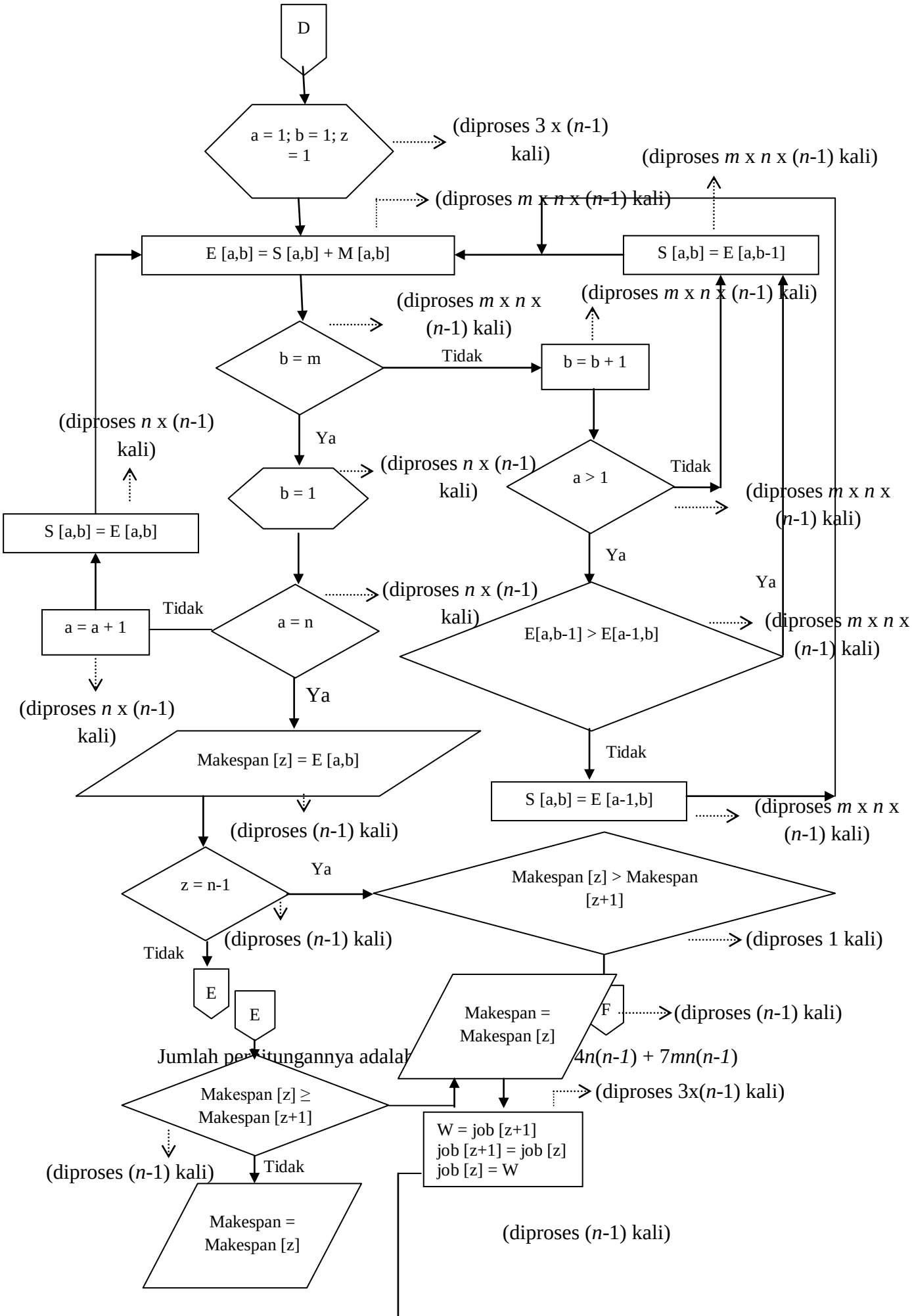
Jumlah perhitungannya adalah $T_2 = 7 + 2n + 3mn$

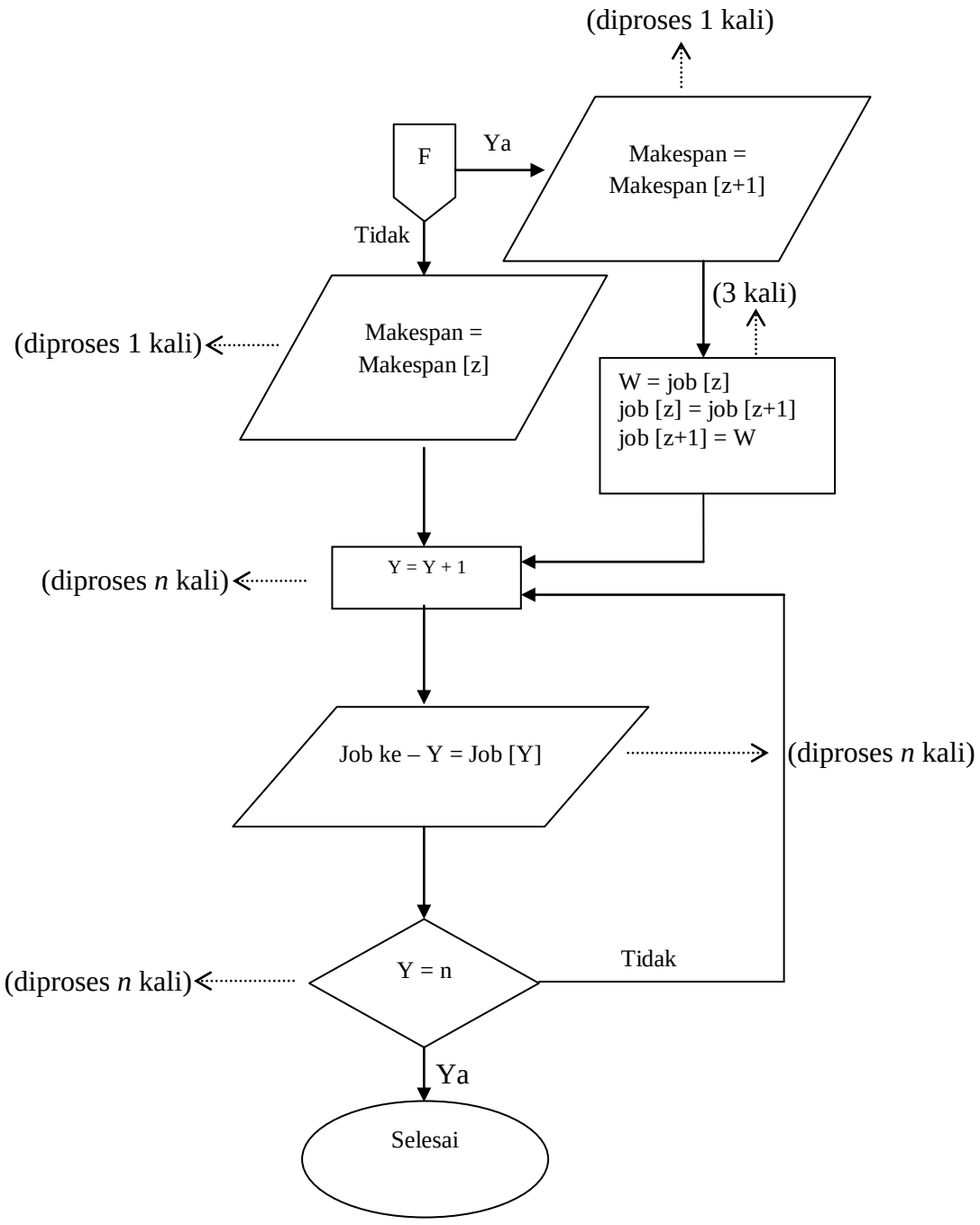


Jumlah perhitungannya adalah $T_3 = 11 + 2(n\text{-}1) + 4n(n\text{-}1)$



Jumlah perhitungannya adalah $T_4 = 6 + 2m + 2m(n\text{-}1) + 3mn(n\text{-}1)$


D
a = 1; b = 1; z = 1
(diproses 3 x (n-1) kali)
(diproses m x n x (n-1) kali)
(diproses m x n x (n-1) kali)
E [a,b] = S [a,b] + M [a,b]
S [a,b] = E [a,b-1]
(diproses m x n x (n-1) kali)
(diproses m x n x (n-1) kali)
b = m
Tidak
b = b + 1
(diproses n x (n-1) kali)
Ya
(diproses n x (n-1) kali)
b = 1
a > 1
Tidak
(diproses m x n x (n-1) kali)
S [a,b] = E [a,b]
Ya
Ya
a = a + 1
Tidak
a = n
(diproses n x (n-1) kali)
E[a,b-1] > E[a-1,b]
(diproses m x n x (n-1) kali)
(diproses n x (n-1) kali)
Ya
Makespan [z] = E [a,b]
Tidak
S [a,b] = E [a-1,b]
(diproses m x n x (n-1) kali)
(diproses (n-1) kali)
z = n-1
Ya
Makespan [z] > Makespan [z+1]
(diproses (n-1) kali)
(diproses 1 kali)
Tidak
E
E
Makespan = Makespan [z]
F
(diproses (n-1) kali)
Jumlah perhitungannya adalah
4n(n-1) + 7mn(n-1)
Makespan [z] ≥ Makespan [z+1]
(diproses 3x(n-1) kali)
W = job [z+1]
job [z+1] = job [z]
job [z] = W
(diproses (n-1) kali)
Tidak
Makespan = Makespan [z]
(diproses (n-1) kali)



Gambar 4.7 *Flowchart* algoritma NEH dan jumlah

Jumlah perhitungannya adalah $T_7 = 5 + 3n$

4.1.5 Perhitungan Kompleksitas Waktu

Untuk menentukan efisiensi algoritma, penulis menyertakan perhitungan kompleksitas waktu algoritma dengan memperhatikan tahapan komputasi masing-masing algoritma pada *flowchart*.

1. Kompleksitas waktu untuk algoritma Pour

Berikut perhitungan kompleksitas total waktu dari tahapan perhitungan untuk algoritma Pour:

$T(m,n) = T_1 + T_2 + T_3 + T_4 + T_5 + T_6$

$= (2 + 4mn + 3n) + (4 + (n\text{-}1) + n\,(n\text{-}1) + 5\,mn\,(n\text{-}1)) + (5 + 4n + 7mn) +$
$(4 + 3n + 3(n\text{-}1) + 4n(n\text{-}1)) + (14 + 2m + 2m(n\text{-}1) + 3mn(n\text{-}1)) +$
$(2 + 4m + 3m(n\text{-}1) + 3mn)$

$= (2 + 4mn + 3n) + (4 + n - 1 + n^2 - n + 5mn^2 - 5mn) + (5 + 4n + 7mn) +$
$(4 + 3n + 3n - 3 + 4n^2 - 4) + (14 + 2m + 2mn - 2m + 3mn^2 - 3mn) +$
$(2 + 4m + 3mn - 3m + 3mn)$

$= 8mn^2 + 5n^2 + 11mn + m + 9n + 23$

Untuk m dan n adalah bilangan asli, maka

$8mn^2 + 5n^2 + 11mn + m + 9n + 23 \leq 8mn^2 + 5mn^2 + 11mn^2 + mn^2 + 9mn^2 + 23mn^2$

$= 57mn^2$

sehingga diperoleh $O(mn^2)$.

2. Kompleksitas waktu untuk algoritma NEH:

Berikut perhitungan kompleksitas total waktu dari tahapan perhitungan untuk algoritma NEH:

$T(m,n) = T_1 + T_2 + T_3 + T_4 + T_5 + T_6 + T_7$

$= (2 + 4mn + 3n) + (7 + 2n + 3mn) + (11 + 2(n\text{-}1) + 4n(n\text{-}1)) + (6 + 2m +$
$2m(n\text{-}1) + 3mn(n\text{-}1)) + (1 + 4(n\text{-}1) + 4n(n\text{-}1) + 7mn(n\text{-}1)) + (10(n\text{-}1) +$
$5m(n\text{-}1)\,) + (5 + 3n)$

$= (2 + 4mn + 3n) + (7 + 2n + 3mn) + (11 + 2n - 2 + 4n^2 - 4n) + (6 + 2m +$

$$2mn - 2m + 3mn^2 - 3mn) + (1 + 4n - 4 + 4n^2 - 4n + 7mn^2 - 7mn) + (10n -$$

$$10 + 5mn - 5m) + (5 + 3n)$$

$$= 10mn^2 + 8n^2 + 4mn - 5m + 16n + 18$$

Untuk m dan n adalah bilangan asli, maka

$$10mn^2 + 8n^2 + 4mn - 5m + 16n + 18 \leq 10mn^2 + 8mn^2 + 4mn^2 + 5mn^2 + 16mn^2 + 18mn^2$$

$$= 51mn^2$$

sehingga diperoleh $O(mn^2)$.

4.2 Pembahasan

Dalam skripsi ini, algoritma Pour dan algoritma NEH digunakan untuk mengoptimalkan waktu produksi kertas di PT. Kertas Leces Probolinggo dengan bantuan aplikasi yang memanfaatkan bahasa pemrograman Matlab. Langkah-langkah dalam algoritma Pour untuk mendapatkan solusi ialah dengan menggunakan pendekatan perhitungan kombinasi, sedangkan langkah-langkah dalam algoritma NEH menggunakan pendekatan perhitungan permutasi.

Pada subbab 4.1 telah dijelaskan dengan perhitungan manual dari masing-masing algoritma. Pada perhitungan algoritma Pour, job yang menempati urutan pertama adalah Job 4 karena memiliki *makespan* lebih kecil, yaitu 1.740 menit dari pada Job 1, Job 2 dan Job 3 dengan *makespan* yang sama, yaitu 1.800 menit. Kemudian urutan kedua ditempati oleh Job 3 yang memiliki *makespan* lebih kecil, yaitu 1.200 menit dari pada Job 1 dan Job 2 dengan masing-masing *makespan* 1.420 menit dan 1.320 menit pada saat menempati urutan kedua. Sedangkan urutan ketiga ditempati oleh Job 1 yang memiliki *makespan* lebih kecil, yaitu 830 menit dibandingkan dengan Job 2 dengan *makespan* 860 menit pada saat menempati urutan ketiga. Kemudian, urutan terakhir ditempati oleh Job 2.

Melalui perhitungan algoritma NEH diperoleh *makespan* terkecil yang sama yakni 1.620 menit dengan beberapa urutan parsial yakni Job 1 – Job 4 – Job 3 – Job 2 atau Job 4 – Job 1 – Job 3 – Job 2 atau Job 4 – Job 3 – Job 1 – Job 2

dibandingkan dengan urutan alternatif Job 4 – Job 3 – Job 2 – Job 1 dengan *makespan* 1.680 menit. Dari perhitungan manual diperoleh jadwal yang meminimumkan *makespan* ialah sebagai berikut:

Tabel 4.10 Solusi Optimal dari Perhitungan Manual

No.	Algoritma	Jadwal	*Makespan*
1	Pour	Job 4 – Job 3 – Job 1 – Job 2	1.620
2	NEH	Job 4 – Job 3 – Job 1 – Job 2	1.620
		Job 4 – Job 1 – Job 3 – Job 2	1.620
		Job 1 – Job 4 – Job 3 – Job 2	1.620

Menurut Tabel 4.10 dapat diketahui bahwa menurut hasil perhitungan manual atau penjadwalan dengan menggunakan algoritma Pour menghasilkan nilai *makespan* yang sama jika dibandingkan dengan nilai *makespan* yang dihasilkan oleh algoritma NEH. Artinya, penggunaan algoritma Pour dan algoritma NEH sama-sama efektif jika diterapkan pada penjadwalan produksi kertas di PT. Kertas Leces Probolinggo dari pada produksi kertas menggunakan metode FCFS, karena dapat mengurangi waktu operasional mesin dalam proses produksi sehingga dapat pula mengurangi biaya produksi.

Apabila ditinjau dari perhitungan kompleksitas waktu yang dihasilkan. Algoritma Pour memiliki kompleksitas waktu yakni $O(mn^2)$. Sedangkan algoritma NEH memiliki kompleksitas waktu dengan $O(mn^2)$. Artinya, proses perhitungan menggunakan algoritma Pour membutuhkan waktu yang sama jika dibandingkan dengan proses perhitungan menggunakan algoritma NEH walaupun jumlah koefisiennya sedikit berbeda. Dengan kata lain menurut kompleksitas waktu yang diperoleh dapat dikatakan algoritma Pour dan algoritma NEH mempunyai tingkat efisiensi yang sama untuk diterapkan pada penjadwalan produksi kertas di PT. Kertas Leces Probolinggo.

Berdasarkan analisa diatas dapat disimpulkan, bahwa dalam penjadwalan produksi kertas di PT. Kertas Leces Probolinggo, algoritma Pour dan algoritma

NEH merupakan algoritma yang memiliki performa baik karena membutuhkan waktu yang sama dalam perhitungannya.

BAB 5. PENUTUP

5.1 Kesimpulan

Berdasarkan hasil dan pembahasan dapat disimpulkan bahwa:

a. Pada penjadwalan produksi kertas di PT. Kertas Leces Probolinggo, nilai *makespan* minimum yang dihasilkan oleh algoritma Pour dan NEH ialah 1.620 menit (27 jam), diperoleh dengan urutan pekerjaan *(job)* salah satu diantaranya yaitu BC Super → HVS 60 *gsm* → HVS 45 *gsm* → MG *paper*.
b. Algoritma Pour dan NEH merupakan algoritma yang memiliki performa baik karena:
   1) nilai *makespan* minimum yang dihasilkan adalah sama yakni 1.620 menit.
   2) memiliki kompleksitas waktu yang sama yakni $O(mn^2)$ sehingga dapat diterapkan pada penjadwalan produksi kertas di PT. Kertas Leces Probolinggo.

5.2 Saran

Algoritma Pour dan NEH merupakan algoritma yang dapat digunakan untuk menyelesaikan permasalahan optimasi dan masih terbuka bagi peneliti lain untuk mengaplikasikan kedua algoritma tersebut ke dalam permasalahan optimasi lainnya selain *flowshop*, seperti *jobshop* dan *travelling salesman problem* (TSP). Penggunaan bahasa pemrograman selain Matlab sebagai aplikasi juga masih terbuka bagi peneliti lainnya, seperti *Java Script* dan *PHP*.

DAFTAR PUSTAKA

Baker, K. R. 1974. *Introduction to Squencing and Scheduling*. New York: John Wiley dan Sons.

Berlianty, I & Arifin, M.2010. *Teknik-Teknik Optimasi Heuristik*. Grahailmu: Jogjakarta.

Ginting, R. 2009. *Penjadwalan Mesin*. Grahailmu: Jogjakarta.

Huda, M. 2010. *Membuat Aplikasi Database dengan Java, MySQL dan NetBeans*. Elex Media Komputindo: Jakarta.

Imam, H, Budi, R & Arif, H. 2007. *Mudah Belajar Java*. Informatika: Jakarta.

Laha, Dipak. 2008. *Heuristics and Metaheuristics for Solving Scheduling Problems*. India : Jadavpur University.

Pour, H. D. 2001. "*A New Heuristic for n-Job m-Machine Flowshop Problem"*. Production Planning Control, vol. 12, no. 7.

Wibowo, Ferry Wahyu. 2011. *Matematika Diskrit: Kompleksitas Algoritma.* STMIK Amikom Jogjakarta.

LAMPIRAN

A. Hasil Perhitungan Algoritma Pour

Langkah-langkah penyelesaian dengan menggunakan algoritma Pour adalah sebagai berikut:

d. Menentukan urutan pertama
   1. Job 1 ditempatkan sebagai urutan pertama
      a. Job 1 dipilih untuk menduduki urutan pertama sehingga waktu proses Job 1 pada semua mesin dianggap 0. Tempatkan Job selain Job 1 sebagai urutan pertama pada urutan berikutnya.

Tabel 2.3.1 langkah 1 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 1	-	-	-	-	-
Job 2	300	60	35	30	55
Job 3	360	80	60	70	90
Job 4	420	100	70	90	100

      b. Memilih waktu proses terkecil untuk masing-masing mesin yaitu:
         M1 = 300 M2 = 60 M3 = 35 M4 = 30 M5 = 55
      c. Lakukan penambahan waktu proses pada setiap *Pij* mulai dari yang terkecil menuju yang terbesar.

Tabel 2.3.2 langkah 3 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 1	-	-	-	-	-
Job 2	300	60	35	30	55
Job 3	660	140	95	100	145
Job 4	1080	240	165	190	245

d. Hitung $\sum C_i$ untuk setiap job yang ada.

Tabel 2.3.3 langkah 4 pada algoritma Pour

	M1	M2	M3	M4	M5	$\sum C_i$
Job 1	-	-	-	-	-	-
Job 2	300	60	35	30	55	480
Job 3	660	140	95	100	145	1140
Job 4	1080	240	165	190	245	1920

e. Mengurutkan $\sum C_i$ dimulai dari yang terkecil hingga yang terbesar dimana Job 1 sebagai urutan pertama dan diperoleh urutan sebagai berikut: Job 1 – Job 2 – Job 3 – Job 4.

f. Hitung *makespan*

Tabel 2.3.4 langkah 6 pada algoritma Pour

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 1	0	360	360	430	430	475	475	515	515	600
Job 2	360	660	660	720	720	755	755	785	785	830
Job 3	660	1020	1020	1100	1100	1160	1160	1230	1230	1320
Job 4	1020	1440	1440	1540	1540	1610	1610	1700	1700	1800

g. Dari data di atas diperoleh *makespan* 1.800 menit.

2. Job 2 ditempatkan sebagai urutan pertama

a. Job 2 dipilih untuk menduduki urutan pertama sehingga waktu proses Job 2 pada semua mesin dianggap 0. Tempatkan Job selain Job 2 sebagai urutan pertama pada urutan berikutnya.

Tabel 2.3.5 langkah 1 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 2	-	-	-	-	-
Job 3	360	80	60	70	90
Job 4	420	100	70	90	100
Job 1	360	70	45	40	85

b. Memilih waktu proses terkecil untuk masing-masing mesin yaitu:

M1 = 360 M2 = 70 M3 = 45 M4 = 40 M5 = 85

c. Lakukan penambahan waktu proses pada setiap *Pij* mulai dari yang terkecil menuju yang terbesar.

Tabel 2.3.6 langkah 3 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 2	-	-	-	-	-
Job 3	720	150	105	110	175
Job 4	1140	250	175	200	275
Job 1	360	70	45	40	85

d. Hitung $\sum C_i$ untuk setiap job yang ada.

Tabel 2.3.7 langkah 4 pada algoritma Pour

	M1	M2	M3	M4	M5	$\sum C_i$
Job 2	-	-	-	-	-	-
Job 3	720	150	105	110	175	1260
Job 4	1140	250	175	200	275	2040
Job 1	360	70	45	40	85	600

e. Mengurutkan $\sum C_i$ dimulai dari yang terkecil hingga yang terbesar dimana Job 2 sebagai urutan pertama dan diperoleh urutan sebagai berikut: Job 2 – Job 1 – Job 3 – Job 4.

f. Hitung *makespan*

Tabel 2.3.8 langkah 6 pada algoritma Pour

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 2	0	300	300	360	360	395	395	425	425	480
Job 1	300	660	660	730	730	775	775	815	815	900
Job 3	660	1020	1020	1100	1100	1160	1160	1230	1230	1320
Job 4	1020	1440	1440	1540	1540	1610	1610	1700	1700	1800

g. Dari data di atas diperoleh *makespan* 1.800 menit.

3. Job 3 ditempatkan sebagai urutan pertama

a. Job 3 dipilih untuk menduduki urutan pertama sehingga waktu proses Job 3 pada semua mesin dianggap 0. Tempatkan Job selain Job 3 sebagai urutan pertama pada urutan berikutnya.

Tabel 2.3.9 langkah 1 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 3	-	-	-	-	-
Job 4	420	100	70	90	100
Job 1	360	70	45	40	85
Job 2	300	60	35	30	55

b. Memilih waktu proses terkecil untuk masing-masing mesin yaitu:
M1 = 300 M2 = 60 M3 = 35 M4 = 30 M5 = 55

c. Lakukan penambahan waktu proses pada setiap *Pij* mulai dari yang terkecil menuju yang terbesar.

Tabel 2.3.10 langkah 3 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 3	-	-	-	-	-
Job 4	1080	230	150	160	240
Job 1	660	130	80	70	140
Job 2	300	60	35	30	55

d. Hitung $\sum C_i$ untuk setiap job yang ada.

Tabel 2.3.11 langkah 4 pada algoritma Pour

	M1	M2	M3	M4	M5	$\sum C_i$
Job 3	-	-	-	-	-	-
Job 4	1080	230	150	160	240	1860
Job 1	660	130	80	70	140	1080
Job 2	300	60	35	30	55	480

e. Mengurutkan $\sum C_i$ dimulai dari yang terkecil hingga yang terbesar dimana Job 3 sebagai urutan pertama dan diperoleh urutan sebagai berikut: Job 3 – Job 2 – Job 1 – Job 4.

f. Hitung *makespan*

Tabel 2.3.12 langkah 6 pada algoritma Pour

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 3	0	360	360	440	440	500	500	570	570	660
Job 2	360	660	660	720	720	755	755	785	785	840
Job 1	660	1020	1020	1090	1090	1135	1135	1175	1175	1260
Job 4	1020	1440	1440	1540	1540	1610	1610	1700	1700	1800

g. Dari data di atas diperoleh *makespan* 1.800 menit.

4. Job 4 ditempatkan sebagai urutan pertama

a. Job 4 dipilih untuk menduduki urutan pertama sehingga waktu proses Job 4 pada semua mesin dianggap 0. Tempatkan Job selain Job 4 sebagai urutan pertama pada urutan berikutnya.

Tabel 2.3.13 langkah 1 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 4	-	-	-	-	-
Job 1	360	70	45	40	85
Job 2	300	60	35	30	55
Job 3	360	80	60	70	90

b. Memilih waktu proses terkecil untuk masing-masing mesin yaitu:

M1 = 300 M2 = 60 M3 = 35 M4 = 30 M5 = 55

c. Lakukan penambahan waktu proses pada setiap *Pij* mulai dari yang terkecil menuju yang terbesar.

Tabel 2.3.14 langkah 3 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 4	-	-	-	-	-
Job 1	660	130	80	70	140
Job 2	300	60	35	30	55
Job 3	1020	210	140	140	230

d. Hitung $\sum C_i$ untuk setiap job yang ada.

Tabel 2.3.15 langkah 4 pada algoritma Pour

	M1	M2	M3	M4	M5	$\sum C_i$
Job 4	-	-	-	-	-	-
Job 1	660	130	80	70	140	1080

Job 2	300	60	35	30	55	480
Job 3	1020	210	140	140	230	1740

e. Mengurutkan $\sum C_i$ dimulai dari yang terkecil hingga yang terbesar dimana Job 4 sebagai urutan pertama dan diperoleh urutan sebagai berikut: Job 4 – Job 2 – Job 1 – Job 3.

f. Hitung *makespan*

Tabel 2.3.16 langkah 6 pada algoritma Pour

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 4	0	420	420	520	520	590	590	680	680	780
Job 2	420	720	720	780	780	815	815	845	845	900
Job 1	720	1080	1080	1150	1150	1195	1195	1235	1235	1320
Job 3	1080	1440	1440	1520	1520	1580	1580	1650	1650	1740

g. Dari data di atas diperoleh *makespan* 1.740 menit.

e. Menentukan urutan kedua

1. Job 1 ditempatkan sebagai urutan pertama

a. Job 1 dipilih untuk menduduki urutan pertama sehingga waktu proses Job 1 pada semua mesin dianggap 0. Tempatkan Job selain Job 1 sebagai urutan pertama pada urutan berikutnya.

Tabel 2.3.17 langkah 1 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 1	-	-	-	-	-
Job 2	300	60	35	30	55
Job 3	360	80	60	70	90

b. Memilih waktu proses terkecil untuk masing-masing mesin yaitu:
   M1 = 300 M2 = 60 M3 = 35 M4 = 30 M5 = 55
c. Lakukan penambahan waktu proses pada setiap *Pij* mulai dari yang terkecil menuju yang terbesar.

Tabel 2.3.18 langkah 3 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 1	-	-	-	-	-
Job 2	300	60	35	30	55
Job 3	660	140	95	100	145

d. Hitung $\sum C_i$ untuk setiap job yang ada.

Tabel 2.3.19 langkah 4 pada algoritma Pour

	M1	M2	M3	M4	M5	$\sum C_i$
Job 1	-	-	-	-	-	-
Job 2	300	60	35	30	55	480
Job 3	660	140	95	100	145	1140

e. Mengurutkan $\sum C_i$ dimulai dari yang terkecil hingga yang terbesar dimana Job 1 sebagai urutan pertama dan diperoleh urutan sebagai berikut: Job 1 – Job 2 –Job 3.
f. Hitung *makespan*

Tabel 2.3.20 langkah 6 pada algoritma Pour

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 1	0	360	360	430	430	475	475	515	515	600
Job 2	360	660	660	720	720	755	755	785	785	830
Job 3	660	1020	1020	1200	1200	1260	1260	1330	1330	1420

g. Dari data di atas diperoleh *makespan* 1.420 menit.

2. Job 2 ditempatkan sebagai urutan pertama

a. Job 2 dipilih untuk menduduki urutan pertama sehingga waktu proses Job 2 pada semua mesin dianggap 0. Tempatkan Job selain Job 2 sebagai urutan pertama pada urutan berikutnya.

Tabel 2.3.21 langkah 1 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 2	-	-	-	-	-
Job 3	360	80	60	70	90
Job 1	360	70	45	40	85

b. Memilih waktu proses terkecil untuk masing-masing mesin yaitu:

M1 = 360 M2 = 70 M3 = 45 M4 = 40 M5 = 85

c. Lakukan penambahan waktu proses pada setiap *Pij* mulai dari yang terkecil menuju yang terbesar.

Tabel 2.3.22 langkah 3 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 2	-	-	-	-	-
Job 3	720	150	105	110	175
Job 1	360	70	45	40	85

d. Hitung $\sum C_i$ untuk setiap job yang ada.

Tabel 2.3.23 langkah 4 pada algoritma Pour

	M1	M2	M3	M4	M5	$\sum C_i$
Job 2	-	-	-	-	-	-
Job 3	720	150	105	110	175	1260
Job 1	360	70	45	40	85	600

e. Mengurutkan $\sum C_i$ dimulai dari yang terkecil hingga yang terbesar dimana Job 2 sebagai urutan pertama dan diperoleh urutan sebagai berikut: Job 2 – Job 1 –Job 3.

f. Hitung *makespan*

Tabel 2.3.24 langkah 6 pada algoritma Pour

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 2	0	300	300	360	360	395	395	425	425	480
Job 1	300	660	660	730	730	775	775	815	815	900
Job 3	660	1020	1020	1100	1100	1160	1160	1230	1230	1320

g. Dari data di atas diperoleh *makespan* 1.320 menit.

3. Job 3 ditempatkan sebagai urutan pertama

a. Job 3 dipilih untuk menduduki urutan pertama sehingga waktu proses Job 3 pada semua mesin dianggap 0. Tempatkan Job selain Job 3 sebagai urutan pertama pada urutan berikutnya.

Tabel 2.3.25 langkah 1 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 3	-	-	-	-	-
Job 1	360	70	45	40	85
Job 2	300	60	35	30	55

b. Memilih waktu proses terkecil untuk masing-masing mesin yaitu:

M1 = 300 M2 = 60 M3 = 35 M4 = 30 M5 = 55

c. Lakukan penambahan waktu proses pada setiap *Pij* mulai dari yang terkecil menuju yang terbesar.

Tabel 2.3.26 langkah 3 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 3	-	-	-	-	-

Job 1	720	130	80	70	140
Job 2	300	60	35	30	55

d. Hitung $\sum C_i$ untuk setiap job yang ada.

Tabel 2.3.27 langkah 4 pada algoritma Pour

	M1	M2	M3	M4	M5	$\sum C_i$
Job 3	-	-	-	-	-	-
Job 1	720	130	80	70	140	1140
Job 2	300	60	35	30	55	480

e. Mengurutkan $\sum C_i$ dimulai dari yang terkecil hingga yang terbesar dimana Job 3 sebagai urutan pertama dan diperoleh urutan sebagai berikut: Job 3 – Job 2 –Job 1.

f. Hitung *makespan*

Tabel 2.3.28 langkah 6 pada algoritma Pour

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 3	0	360	360	440	440	500	500	570	570	630
Job 2	360	600	600	660	660	695	695	725	725	780
Job 1	600	960	960	1030	1030	1075	1075	1115	1115	1200

g. Dari data di atas diperoleh *makespan* 1.200 menit.

f. Menentukan urutan ketiga

1. Job 1 ditempatkan sebagai urutan pertama

a. Job 1 dipilih untuk menduduki urutan pertama sehingga waktu proses Job 1 pada semua mesin dianggap 0. Tempatkan Job selain Job 1 sebagai urutan pertama pada urutan berikutnya.

Tabel 2.3.29 langkah 1 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 1	-	-	-	-	-
Job 2	300	60	35	30	55

b. Memilih waktu proses terkecil untuk masing-masing mesin yaitu:
   M1 = 300 M2 = 60 M3 = 35 M4 = 30 M5 = 55
c. Lakukan penambahan waktu proses pada setiap *Pij* mulai dari yang terkecil menuju yang terbesar.

Tabel 2.3.30 langkah 3 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 1	-	-	-	-	-
Job 2	300	60	35	30	55

d. Hitung $\sum C_i$ untuk setiap job yang ada.

Tabel 2.3.31 langkah 4 pada algoritma Pour

	M1	M2	M3	M4	M5	$\sum C_i$
Job 1	-	-	-	-	-	-
Job 2	300	60	35	30	55	480

e. Mengurutkan $\sum C_i$ dimulai dari yang terkecil hingga yang terbesar dimana Job 1 sebagai urutan pertama dan diperoleh urutan sebagai berikut: Job 1 – Job 2.
f. Hitung *makespan*

Tabel 2.3.32 langkah 6 pada algoritma Pour

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 1	0	360	360	430	430	475	475	515	515	600
Job 2	360	660	660	720	720	755	755	785	785	830

g. Dari data di atas diperoleh *makespan* 830 menit.

2. Job 2 ditempatkan sebagai urutan pertama

a. Job 2 dipilih untuk menduduki urutan pertama sehingga waktu proses Job 2 pada semua mesin dianggap 0. Tempatkan Job selain Job 2 sebagai urutan pertama pada urutan berikutnya.

Tabel 2.3.33 langkah 1 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 2	-	-	-	-	-
Job 1	360	70	45	40	85

b. Memilih waktu proses terkecil untuk masing-masing mesin yaitu:

M1 = 360 M2 = 70 M3 = 45 M4 = 40 M5 = 85

c. Lakukan penambahan waktu proses pada setiap *Pij* mulai dari yang terkecil menuju yang terbesar.

Tabel 2.3.34 langkah 3 pada algoritma Pour

	M1	M2	M3	M4	M5
Job 2	-	-	-	-	-
Job 1	360	70	45	40	85

d. Hitung $\sum C_i$ untuk setiap job yang ada.

Tabel 2.3.35 langkah 4 pada algoritma Pour

	M1	M2	M3	M4	M5	$\sum C_i$
Job 2	-	-	-	-	-	-
Job 1	360	70	45	40	85	600

e. Mengurutkan $\sum C_i$ dimulai dari yang terkecil hingga yang terbesar dimana Job 2 sebagai urutan pertama dan diperoleh urutan sebagai berikut: Job 2 – Job 1.

f. Hitung *makespan*

Tabel 2.3.36 langkah 6 pada algoritma Pour

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 2	0	300	300	360	360	395	395	425	425	480
Job 1	300	660	660	730	730	775	775	820	820	860

g. Dari data di atas diperoleh *makespan* 860 menit.

Urutan terpilih adalah Job 4 – Job 3 – Job 1 – Job 2 sehingga diperoleh *makespan* sebagai berikut:

Tabel 2.3.37 Perhitungan *makespan* Job 4 – Job 3 – Job 1 – Job 2

	M1		M2		M3		M4		M5	
	S	E	S	E	S	E	S	E	S	E
Job 4	0	420	420	520	520	590	590	680	680	780
Job 3	420	780	780	960	960	1020	1020	1090	1090	1170
Job 1	780	1140	1140	1210	1210	1250	1250	1295	1295	1380
Job 2	1140	1440	1440	1500	1500	1535	1535	1565	1565	1620

B. Hasil Perhitungan Algoritma NEH

Langkah-langkah penyelesaian dengan mengunakan algoritma NEH adalah sebagai berikut:

Langkah 1:

1. Menjumlahkan waktu proses setiap job.

Tabel 2.4.1 Total waktu proses setiap job

	M1	M2	M3	M4	M5	Total
Job 1	360	70	45	40	85	600
Job 2	300	60	35	30	55	480
Job 3	360	80	60	70	90	660
Job 4	420	100	70	90	100	780

2. Mengurutkan job-job mulai dari waktu proses terbesar.

Tabel 2.4.2 Urutan job mulai dari waktu proses terbesar

	M1	M2	M3	M4	M5	Total
Job 4	420	100	70	90	100	780
Job 3	360	80	60	70	90	660
Job 1	360	70	45	40	85	600
Job 2	300	60	35	30	55	480

3. Membuat penjadwalan parsial Job 4 –Job 3 sehingga diperoleh *makespan* 1.080 menit.

Tabel 2.4.3 Penjadwalan parsial Job 4 – Job 3

M	Job	Durasi	Mulai	Siap
1	Job 4	420	0	420
1	Job 3	360	420	780

2	Job 4	100	420	520
2	Job 3	80	780	860
3	Job 4	70	520	590
3	Job 3	60	860	920
4	Job 4	90	590	680
4	Job 3	70	920	990
5	Job 4	100	680	780
5	Job 3	90	990	1080

4. Alternatif penjadwalan parsial Job 3 – Job 4 sehingga diperoleh *makespan* 1.140 menit.

Tabel 2.4.4 Penjadwalan parsial Job 3 – Job 4

M	Job	Durasi	Mulai	Siap
1	Job 3	360	0	360
1	Job 4	420	360	780
2	Job 3	80	360	440
2	Job 4	100	780	880
3	Job 3	60	440	500
3	Job 4	70	880	950
4	Job 3	70	500	570
4	Job 4	90	950	1040
5	Job 3	90	570	650
5	Job 4	100	1040	1140

5. Dari perhitungan penjadwalan parsial di atas, maka dipilih urutan Job 4 – Job 3 karena menghasilkan *makespan* terkecil, yaitu 1.080 menit.

Langkah 2:

1. Membuat alternatif penjadwalan parsial dari Job 4 – Job 3 – Job 1 sehingga diperoleh beberapa kemungkinan, yaitu:

a. Job 4 – Job 3 – Job 1
b. Job 4 – Job 1 – Job 3
c. Job 1 – Job 4 – Job 3

2. Menghitung *makespan* dari alternatif penjadwalan poin a, b dan c pada nomor 1.

a. Penjadwalan parsial Job 4 – Job 3 – Job 1 sehingga diperoleh *makespan* 1.380 menit.

Tabel 2.4.5 Penjadwalan parsial Job 4 – Job 3 – Job 1

M	Job	Durasi	Mulai	Siap
1	Job 4	420	0	420
1	Job 3	360	420	780
1	Job 1	360	780	1140
2	Job 4	100	420	520
2	Job 3	80	780	560
2	Job 1	70	1140	1210
3	Job 4	70	520	590
3	Job 3	60	560	620
3	Job 1	45	1210	1255
4	Job 4	90	590	680
4	Job 3	70	620	690
4	Job 1	40	1255	1295
5	Job 4	100	680	780
5	Job 3	90	690	780
5	Job 1	85	1295	1380

b. Penjadwalan parsial Job 4 – Job 1 – Job 3 sehingga diperoleh *makespan* 1.435 menit.

Tabel 2.4.6 Penjadwalan parsial Job 4 – Job 1 – Job 3

M	Job	Durasi	Mulai	Siap

1	Job 4	420	0	420
1	Job 1	360	420	780
1	Job 3	360	780	1140
2	Job 4	100	420	520
2	Job 1	70	780	850
2	Job 3	80	1140	1220
3	Job 4	70	520	590
3	Job 1	45	850	895
3	Job 3	60	1220	1280
4	Job 4	90	590	680
4	Job 3	40	895	935
4	Job 1	70	1280	1350
5	Job 4	100	680	780
5	Job 3	90	935	1025
5	Job 1	85	1350	1435

c. Penjadwalan parsial Job 1 – Job 4 – Job 3 sehingga diperoleh *makespan* 1.440 menit.

Tabel 2.4.7 Penjadwalan parsial Job 1 – Job 4 – Job 3

M	Job	Durasi	Mulai	Siap
1	Job 1	360	0	360
1	Job 4	420	360	780
1	Job 3	360	780	1140
2	Job 1	70	360	430
2	Job 4	100	780	880
2	Job 3	80	1140	1220
3	Job 1	45	430	475
3	Job 4	70	880	950
3	Job 3	60	1220	1280
4	Job 1	40	475	515

4	Job 4	90	950	1040
4	Job 3	70	1280	1350
5	Job 1	85	515	600
5	Job 4	100	1040	1140
5	Job 3	90	1350	1440

3. Membuat alternatif penjadwalan parsial dari Job 4 – Job 3 – Job 2 sehingga diperoleh beberapa kemungkinan, yaitu:
   a. Job 4 – Job 3 – Job 2
   b. Job 4 – Job 2 – Job 3
   c. Job 2 – Job 4 – Job 3
4. Menghitung *makespan* dari alternatif penjadwalan poin a, b dan c pada nomor 3.
   a. Penjadwalan parsial Job 4 – Job 3 – Job 2 sehingga diperoleh *makespan* 1.260 menit.

Tabel 2.4.8 Penjadwalan parsial Job 4 – Job 3 – Job 2

M	Job	Durasi	Mulai	Siap
1	Job 4	420	0	420
1	Job 3	360	420	780
1	Job 2	300	780	1080
2	Job 4	100	420	520
2	Job 3	80	780	860
2	Job 2	60	1080	1140
3	Job 4	70	520	590
3	Job 3	60	860	920
3	Job 2	35	1140	1175
4	Job 4	90	590	680
4	Job 3	70	920	990
4	Job 2	30	1175	1205
5	Job 4	100	680	780

5	Job 3	90	990	1080
5	Job 2	55	1205	1260

b. Penjadwalan parsial Job 4 – Job 2 – Job 3 sehingga diperoleh *makespan* 1.280 menit.

Tabel 2.4.9 Penjadwalan parsial Job 4 – Job 2 – Job 3

M	Job	Durasi	Mulai	Siap
1	Job 4	420	0	420
1	Job 2	300	420	720
1	Job 3	360	720	1080
2	Job 4	100	420	520
2	Job 2	60	720	780
2	Job 3	80	1080	1160
3	Job 4	70	520	590
3	Job 3	60	780	840
3	Job 2	35	1160	1195
4	Job 4	90	590	680
4	Job 3	70	840	910
4	Job 2	30	1195	1225
5	Job 4	100	680	780
5	Job 3	90	910	1000
5	Job 2	55	1225	1280

c. Penjadwalan parsial Job 2 – Job 4 – Job 3 sehingga diperoleh *makespan* 1.380 menit.

Tabel 2.4.10 Penjadwalan parsial Job 2 – Job 4 – Job 3

M	Job	Durasi	Mulai	Siap
1	Job 2	300	0	300
1	Job 4	420	300	720

1	Job 3	360	720	1080
2	Job 2	60	300	360
2	Job 4	100	720	820
2	Job 3	80	1080	1160
3	Job 2	35	360	395
3	Job 4	70	820	890
3	Job 3	60	1160	1220
4	Job 2	30	395	425
4	Job 4	90	890	980
4	Job 3	70	1220	1290
5	Job 2	55	425	480
5	Job 4	100	980	1080
5	Job 3	90	1290	1380

5. Dari perhitungan penjadwalan parsial di atas, maka dipilih urutan Job 4 – Job 3 – Job 2 karena menghasilkan *makespan* terkecil, yaitu 1.260 menit.

Langkah 3:

1. Membuat alternatif penjadwalan parsial dari Job 4 – Job 3 – Job 2 – Job 1 sehingga diperoleh beberapa kemungkinan, yaitu:
   a. Job 4 – Job 3 – Job 2 – Job 1
   b. Job 4 – Job 3 – Job 1 – Job 2
   c. Job 4 – Job 1 – Job 3 – Job 2
   d. Job 1 – Job 4 – Job 3 – Job 2
2. Menghitung *makespan* dari alternatif penjadwalan poin a, b, c dan d pada nomor 1.
   a. Penjadwalan parsial Job 4 – Job 3 – Job 2 – Job 1 sehingga diperoleh *makespan* 1680 menit.

Tabel 2.4.11 Penjadwalan parsial Job 4 – Job 3 – Job 2 – Job 1

M	Job	Durasi	Mulai	Siap

1	Job 4	420	0	420
1	Job 3	360	420	780
1	Job 2	300	780	1080
1	Job 1	360	1080	1440
2	Job 4	100	420	520
2	Job 3	80	780	860
2	Job 2	60	1080	1140
2	Job 1	70	1440	1510
3	Job 4	70	520	590
3	Job 3	60	860	920
3	Job 2	35	1140	1175
3	Job 1	45	1510	1555
4	Job 4	90	590	680
4	Job 3	70	920	990
4	Job 2	30	1175	1205
4	Job 1	40	1555	1595
5	Job 4	100	680	780
5	Job 3	90	990	1080
5	Job 2	55	1205	1260
5	Job 1	85	1595	1680

b. Penjadwalan parsial Job 4 – Job 3 – Job 1 – Job 2 sehingga diperoleh *makespan* 1.620 menit.

Tabel 2.4.12 Penjadwalan parsial Job 4 – Job 3 – Job 1 – Job 2

M	Job	Durasi	Mulai	Siap
1	Job 4	420	0	420
1	Job 3	360	420	780
1	Job 1	360	780	1140
1	Job 2	300	1140	1440

2	Job 4	100	420	520
2	Job 3	80	780	860
2	Job 1	70	1140	1210
2	Job 2	60	1440	1500
3	Job 4	70	520	590
3	Job 3	60	860	920
3	Job 1	45	1210	1255
3	Job 2	35	1500	1535
4	Job 4	90	590	680
4	Job 3	70	920	990
4	Job 1	40	1255	1295
4	Job 2	30	1535	1565
5	Job 4	100	680	780
5	Job 3	90	990	1080
5	Job 1	85	1295	1380
5	Job 2	55	1565	1620

c. Penjadwalan parsial Job 4 – Job 1 – Job 3 – Job 2 sehingga diperoleh *makespan* 1.620 menit.

Tabel 2.4.13 Penjadwalan parsial Job 4 – Job 1 – Job 3 – Job 2

M	Job	Durasi	Mulai	Siap
1	Job 4	420	0	420
1	Job 1	360	420	780
1	Job 3	360	780	1140
1	Job 2	300	1140	1440
2	Job 4	100	420	520
2	Job 1	70	780	850
2	Job 3	80	1140	1220
2	Job 2	60	1440	1500

3	Job 4	70	520	590
3	Job 1	45	850	895
3	Job 3	60	1220	1280
3	Job 2	35	1500	1535
4	Job 4	90	590	680
4	Job 1	40	895	935
4	Job 3	70	1280	1350
4	Job 2	30	1535	1565
5	Job 4	100	680	780
5	Job 1	85	935	1020
5	Job 3	90	1350	1440
5	Job 2	55	1565	1620

d. Penjadwalan parsial Job 1 – Job 4 – Job 3 – Job 2 sehingga diperoleh *makespan* 1.620 menit.

Tabel 2.4.14 Penjadwalan parsial Job 1 – Job 4 – Job 3 – Job 2

M	Job	Durasi	Mulai	Siap
1	Job 1	360	0	360
1	Job 4	420	360	780
1	Job 3	360	780	1140
1	Job 2	300	1140	1440
2	Job 1	70	360	430
2	Job 4	100	780	880
2	Job 3	80	1140	1220
2	Job 2	60	1440	1500
3	Job 1	45	430	475
3	Job 4	70	880	950
3	Job 3	60	1220	1280
3	Job 2	35	1500	1535

4	Job 1	40	475	515
4	Job 4	90	950	1040
4	Job 3	70	1280	1350
4	Job 2	30	1535	1565
5	Job 1	85	515	600
5	Job 4	100	1040	1140
5	Job 3	90	1350	1440
5	Job 2	55	1565	1620

3. Dari perhitungan penjadwalan parsial di atas, ada sebanyak 3 urutan alternatif job yang menghasilkan *makespan* sama besar yakni 1.620 menit. Urutan alternatif tersebut adalah Job 1 – Job 4 – Job 3 – Job 2 atau Job 4 – Job 1 – Job 3 – Job 2 atau Job 4 – Job 3 – Job 1 – Job 2.

C. Program

```
%MEAN FLOWTIME
J0=J;x2=0;
for x=1:J0-1

% ALGORITMA POUR
[J M]=size(data); % J: banyaknya Job, M= Banyaknya Mesin;
Data=data; Data_baru=[];urutan=[]; wkt_floatime=[];

for i=1:J % menentukan Makespan tiap awalan Job

% membuat data baru untuk data ke- i di anggap sebagai awalan
    i2=0;
for i1=1:J
if i==i1

continue;
elseif i1>i
            i2=i2+1;
            Data_baru(i2,:)=data(i1,:);
            urutan(i2)=i1;
end
if i1==J && i2~=J-1
           Data_baru(i2+1:J-1,:)=data(1:i-1,:);
           urutan(i2+1:J-1)=1:i-1;
end
end
    Data(1,:)=data(i,:);
    Data(2:J,:)=Data_baru;
```

```
    Urut(1)=i; Urut(2:J)=urutan;
if i==J
       Data_baru(1:J-1,:)=data(1:J-1,:);
    Data(2:J,:)=Data_baru;
    Urut(2:J)=1:J-1;
end
    Simpanan_urutan(i,:)=Urut;
% menentukan waktu minimum tiap mesin dan membuat data baru nya
for k=1:M
        waktu_min(k)=min(Data_baru(:,k));
        ab=sort(Data_baru(:,k));
        a=waktu_min(k); jml=0;
for i1=1:J-1
            a1=ab(i1);
for i2=1:J-1
            b=Data_baru(i2,k);
if a1==b
                Data_baru(i2,k)=jml+Data_baru(i2,k);
                jml=Data_baru(i2,k);
end% end nya if
end
end



end
% Menentukan Urutan Job
    Total=sum(Data_baru,2);
    Total1=sort(Total);urutan=Urut(2:J);
    urutan_baru(1)=i;
for i2=1:J-1
```

```
for i1=1:J-1
        a=Total(i1); b=Total1(i2);
if a==b
           urutan_baru(i2+1)=urutan(i1);
end


end
end
    Hasil(i,:)=urutan_baru;
for i1=1:J
%         i3=Hasil(i,i1);
        i3=urutan_baru(i1);
    Data(i1,:)=data(i3,:);
end
% Membuat Makespan
        [waktu1(i) total(i) Makespan1]=makespan(Data);


clear data_urut
end
data1=[];
% mencari total waktu minimum
minwkt=min(waktu1);
x1=0;
for z=1:J
    wkt=waktu1(z);
if minwkt==wkt
       x2=x2+1;
    S_data(x2,:)=data(z,:);
else
       x1=x1+1;
```

```
        data1(x1,:)=data(z,:);
end
end
data=[];data=data1;

end

data=get(tabel,'Data');
data_lama=data; ulang=0; i10=0;
[J M]=size(data_lama);
posisi=1:J;
algoritma_poor;
[J M]=size(data_lama);

% Mementukan urutan job
S_data(J,:)=data;
for i=1:J
    dat1=S_data(i,:);
for i1=1:J
   dat2=data_lama(i1,:);
   sama=0;
for i2=1:M
      a=dat1(i2);
      b=dat2(i2);
if a==b
        sama=sama+1;

if sama==M
           urutan(i)=i1;
end
```

```
end

end

end

end
Data_baru(1:J,:)=S_data(1:J,:);
[Waktu total Makespan]=makespan(Data_baru);
% disp(['Total Waktu : ' num2str(Waktu) ' Menit']);
kat=joinseq('Waktu: ', num2str(Waktu));
kat=joinseq(kat,' Menit');
set(label_waktu,'string',kat);
% disp(['Urutan Job : ' num2str(urutan)]);
kat1=joinseq('Urutan: ', num2str(urutan));
set(label_job,'string',kat1);
job1=cell(job,1);
for i=1:job;
    i1=num2str(urutan(i));
    kata=joinseq('Job ', i1);
    job1(i,1)={kata};
end
job1=char(job1);
cnames=job1;
set(tabel2,'Data',Makespan);
set(tabel2,'RowName',cnames);


x=Waktu;
```

```
jml1=0;a1=0;beda=1/J; beda2=1/M;
figure(2); hold off; plot(0);
for i1=1:J
Q=Makespan(i1,:);
jml=0;
for i=1:M
a=[i,i]; b=[Q(2*i-1) Q(2*i)];
area(b,a,'FaceColor',[1-i1*beda 0+i*beda2 1]);
if i>1
   area(b,[i-1 i-1], 'FaceColor',[1 1 1]);
end
   axis([0 x 0 M+1]);
   title('Algoritma POUR');
   xlabel('Waktu (Menit)'); ylabel('Job');
   pause(1);
   hold on
end
end
```

```
job=str2num(get(edit1,'string'));mesin=str2num(get(edit2,'string')
);
% nama job
job1=cell(job,1);
for i=1:job;
    i1=num2str(i);
    kata=joinseq('Job ', i1);
    job1(i,1)={kata};
end
```

```
mesin1=cell(mesin,1);
for i=1:mesin;
    i1=num2str(i);
    kata1=joinseq('Mesin ', i1);
    mesin1(i,1)={kata1};
    mesin2(2*i-1,1)='S';
    mesin2(2*i,1)='E';
end
job1=char(job1);
mesin1=char(mesin1);
cnames = {job1};  rnames={mesin1};  dat=zeros(job,mesin);
tabel = uitable('Data', dat,...
'Parent',win1,...
'units','point',...
'ColumnName',rnames,...
'RowName',cnames,...
'hitTest','on',...
'backgroundcolor',[1 1 .5; .5 1 1],...
'ColumnEditable',true,...
'fontname','times new roman',...
'foregroundcolor',[0 0 0],...
'fontsize',14,...
'Position',[190 135 335 105]);
dat2=zeros(job,mesin*2);
tabel2 = uitable('Parent',win1,...
'units','point',...
'Data', dat2,...
'ColumnName',mesin2,...
'RowName',cnames,...
'hitTest','on',...
```

```
'backgroundcolor',[1 1 .5; .5 1 1],...
'fontname','times new roman',...
'foregroundcolor',[0 0 0],...
'fontsize',14,...
'Position',[190 13 335 105]);

proses1=uicontrol('parent',win1,...
'units','points',...
'position',[18 130 130 20],...
'style','Pushbutton',...
'backgroundcolor',[.8 .8 .6],...
'string','Algoritma POUR',...
'HorizontalAlignment','left',...
'callback','DATA',...
'fontname','times new roman',...
'fontsize',14);

proses2=uicontrol('parent',win1,...
'units','points',...
'position',[18 107 130 20],...
'style','Pushbutton',...
'backgroundcolor',[.8 .8 .6],...
'callback','NEH',...
'HorizontalAlignment','left',...
'string','Algoritma NEH',...
'fontname','times new roman',...
'fontsize',14);


% Membuat Makespan
```

```
function [waktu total simpan]=makespan(data)
 [J M]=size(data); Data=data;
for i1=1:M


        S(1,1)=0;
if i1==1
            S(1,1)=0;
for it=2:J
               S(it,1)=S(it-1,1)+Data(it-1,i1);


end
            E(1:J-1,1)=S(2:J,1);
            E(J,1)=S(J,1)+Data(J,1);
else
for i2=1:J
if i2==1
                S(i2,i1)=E(i2,i1-1);
E(i2,i1)=S(i2,i1)+Data(i2,i1);
else
                S(i2,i1)=max(E(i2,i1-1),E(i2-1,i1));
                E(i2,i1)=S(i2,i1)+Data(i2,i1);
end


end
end


end


 waktu=E(J,M);
 total=sum(E(:,M));
```

```
for i=1:M
    simpan(:,2*i-1)=S(:,i);
    simpan(:,2*i)=E(:,i);
end


data=get(tabel,'Data');

[J M]=size(data);
% membuat urutan dari besar ke kecil

Total=sum(data,2);
total=sort(Total,'descend');

for i=1:J
   tot1=total(i);
for i1=1:J
       tot2=Total(i1);
if tot1==tot2
          urut(i)=i1 ;
end

end

end


for i=2:J-1
   urut1=urut(1:i-1);
```

```
   matrik=eye(i);

for x=1:i% baris
       x2=0;
       a=matrik(x,:);
for x1=1:i % kolom
         a1=a(x1);
if a1==1
               matrik(x,x1)=urut(i);
else
               x2=x2+1;
               matrik(x,x1)=urut1(x2);
end

end

end

for y=1:i % baris
       a=matrik(y,:);
for ii=1:i % kolom
           a1=a(ii);
           Data(ii,:)=data(a1,:);
end

       [Waktu(y) tot Makespan1]=makespan(Data);
end

   wkt=min(Waktu);
for i2=1:i
```

```
        wkt1=Waktu(i2);
if wkt1==wkt
            mtr=matrik(i2,:);
            urut(1:i)=mtr;
end

end




end

% disp(['Urutan Job : ' num2str(urut)]);

for i=1:J
    a=urut(i);
    Data(i,:)=data(a,:);

end

[Waktu1 tot Makespan]=makespan(Data);
% disp(['Total Waktu : ' num2str(Waktu1) ' Menit']);
kat=joinseq('Waktu: ', num2str(Waktu1));
kat=joinseq(kat,' Menit');
set(label_waktu,'string',kat);
kat1=joinseq('Urutan: ', num2str(urut));
set(label_job,'string',kat1);
job1=cell(job,1);
for i=1:job;
    i1=num2str(urut(i));
```

```
    kata=joinseq('Job ', i1);
    job1(i,1)={kata};
end
job1=char(job1);
cnames=job1;
set(tabel2,'Data',Makespan);
set(tabel2,'RowName',cnames);

x=Waktu1;
jml1=0;a1=0;beda=1/J; beda2=1/M;
figure(2); hold off;plot(0);
for i1=1:J
Q=Makespan(i1,:);
jml=0;
for i=1:M
a=[i,i]; b=[Q(2*i-1) Q(2*i)];
area(b,a,'FaceColor',[1-i1*beda 0+i*beda2 1]);
if i>1
   area(b,[i-1 i-1], 'FaceColor',[1 1 1]);
end
   axis([0 x 0 M+1]);
   xlabel('Waktu (Menit)'); ylabel('Job');
   title('Algoritma NEH');
   pause(1);
   hold on
end
end


clc; clear all;
```

```
close all;
win1=figure(...
'units','points',...
'position',[50 100 550 350],...
'color',[.8 .8 .9],...
'menubar','none',...
'resize','off',...
'numbertitle','off',...
'name','ALGORITMA POUR DAN ALGORITMA NEH');
%=======================================
grafik1=axes('parent',win1,...
'units','points',...
'position',[10 270 90 70],...
'fontsize',8,...
'color',[1 1 1]);
olmat=imread('unej.jpg');
    imshow(olmat);
        set(win1,'CurrentAxes',grafik1);
%=============================================
label1=uicontrol('parent',win1,...
'units','points',...
'position',[90 320 450 24],...
'style','Text',...
'string',' PENJADWALAN PRODUKSI KERTAS MENGGUNAKAN ',...
'backgroundcolor',[.8 .8 .9],...
'fontname','Times New Roman',...
'fontsize',15,...
'fontweight','bold',...
'foregroundcolor',[.0 .0 .0]);
```

```
label1=uicontrol('parent',win1,...
'units','points',...
'position',[90 296 450 24],...
'style','Text',...
'string',' ALGORITMA POUR DAN ALGORITMA NEH ',...
'backgroundcolor',[.8 .8 .9],...
'fontname','Times New Roman',...
'fontsize',15,...
'fontweight','bold',...
'foregroundcolor',[.0 .0 .0]);

label1=uicontrol('parent',win1,...
'units','points',...
'position',[90 272 450 24],...
'style','Text',...
'string','DI PT. KERTAS LECES PROBOLINGGO ',...
'backgroundcolor',[.8 .8 .9],...
'fontname','Times New Roman',...
'fontsize',15,...
'fontweight','bold',...
'foregroundcolor',[.0 .0 .0]);

label1=uicontrol('parent',win1,...
'units','points',...
'position',[0 260 700 1],...
'style','Text',...
'backgroundcolor',[.3 .3 .3],...
'foregroundcolor',[1 1 1]);

label1=uicontrol('parent',win1,...
```

```
'units','points',...
'position',[165 0 1 260],...
'style','Text',...
'backgroundcolor',[.3 .3 .3],...
'foregroundcolor',[1 1 1]);

%=========================================================

hp = uipanel('parent',win1,...
'Title','Input','FontSize',12,...
'units','points',...
'BackgroundColor',[.8 .8 .9],...
'Position',[13 170 140 90]);

label1=uicontrol('parent',win1,...
'units','points',...
'position',[18 215 70 25],...
'style','Text',...
'HorizontalAlignment','left',...
'string','Jumlah Job :',...
'backgroundcolor',[.8 .8 .9],...
'fontname','Times New Roman',...
'fontsize',12,...
'fontweight','bold');

edit1=uicontrol('parent',win1,...
'units','points',...
'position',[100 224 40 20],...
'style','edit',...
'string','0',...
```

```
'backgroundcolor',[.8 .8 .9],...
'fontname','comic',...
'fontsize',12,...
'fontweight','bold');

label1=uicontrol('parent',win1,...
'units','points',...
'position',[18 200 100 20],...
'style','Text',...
'HorizontalAlignment','left',...
'string','Jumlah Mesin :',...
'backgroundcolor',[.8 .8 .9],...
'fontname','Times New Roman',...
'fontsize',12,...
'fontweight','bold');

edit2=uicontrol('parent',win1,...
'units','points',...
'position',[100 204 40 20],...
'style','edit',...
'string','0',...
'backgroundcolor',[.8 .8 .9],...
'fontname','comic',...
'fontsize',12,...
'fontweight','bold');

proses=uicontrol('parent',win1,...
'units','points',...
'position',[30 177 100 20],...
'style','Pushbutton',...
```

```
'callback','Input_Data',...
'backgroundcolor',[.8 .8 .6],...
'string','Ok',...
'fontname','times new roman',...
'fontsize',14);
%================================================

hp = uipanel('parent',win1,...
'Title','Jenis Metode','FontSize',12,...
'units','points',...
'BackgroundColor',[.8 .8 .9],...
'Position',[13 95 140 75]);

proses1=uicontrol('parent',win1,...
'units','points',...
'position',[18 130 130 20],...
'style','Pushbutton',...
'backgroundcolor',[.8 .8 .6],...
'string','Algoritma POUR',...
'HorizontalAlignment','left',...
'fontname','times new roman',...
'fontsize',14);

proses2=uicontrol('parent',win1,...
'units','points',...
'position',[18 107 130 20],...
'style','Pushbutton',...
'backgroundcolor',[.8 .8 .6],...
'HorizontalAlignment','left',...
'string','Algoritma NEH',...
```

```
'fontname','times new roman',...
'fontsize',14);

proses2=uicontrol('parent',win1,...
'units','points',...
'position',[18 10 130 20],...
'style','Pushbutton',...
'backgroundcolor',[.8 .8 .6],...
'callback','POOR_vs_NEH',...
'HorizontalAlignment','left',...
'string','Reset',...
'fontname','times new roman',...
'fontsize',14);

%================================================

hp = uipanel('parent',win1,...
'Title','Output','FontSize',12,...
'units','points',...
'BackgroundColor',[.8 .8 .9],...
'Position',[13 35 140 60]);

label_waktu=uicontrol('parent',win1,...
'units','points',...
'position',[18 60 130 20],...
'style','Text',...
'HorizontalAlignment','left',...
'string','Waktu:  Menit',...
'backgroundcolor',[.8 .8 .9],...
'fontname','Times New Roman',...
```

```
'fontsize',12,...
'fontweight','bold');

label_job=uicontrol('parent',win1,...
'units','points',...
'position',[18 40 130 20],...
'style','Text',...
'HorizontalAlignment','left',...
'string','Urutan : ',...
'backgroundcolor',[.8 .8 .9],...
'fontname','Times New Roman',...
'fontsize',12,...
'fontweight','bold');

%================================================

hp = uipanel('parent',win1,...
'Title','Input Data','FontSize',12,...
'units','points',...
'BackgroundColor',[.8 .8 .9],...
'Position',[180 130 355 130]);

hp = uipanel('parent',win1,...
'Title','Makespan','FontSize',12,...
'units','points',...
'BackgroundColor',[.8 .8 .9],...
'Position',[180 5 355 130]);
```