APLIKASI PEMOGRAMAN INTEGER DALAM USAHA PENJADWALAN TENAGA KERJA PADA INDUSTRI TEKSTIL

Oleh : Moekarto Moeliono, Santoso
Balai Besar Tekstil, Jl. A. Yani No. 390 Bandung Telp. 022.7206214-5 Fax. 022.7271288;
E-mail : moekarto55@kemenperin.go.id

Tulisan diterima : 29 Desember 2010, Selesai diperiksa : 4 Maret 2011

ABSTRAK

Penelitian dan penerapan pemograman *integer* (*Integer Programming/IP 1-5*) dengan model *Solver XA* telah dilakukan disalah satu industri tekstil, khususnya industri pertenunan *(weaving)* yang ada di kota Bandung. Penggunaan bahasa pemograman matematika ini mengacu pada sistem pengefisienan biaya tenaga kerja dan penggunaan waktu yang terendah yang dapat digunakan di lapangan industri.

Maksud dan tujuan dari penelitian ini, adalah memanfaatkan aplikasi pemograman matematika dari *integer* model *solver XA* dalam pemecahan masalah dalam sistem penjadwalan kerja agar dihasilkan proses kerja di lapangan menjadi efisien, efektif, dan produktif yang dampaknya bagi tenaga kerja berkualifikasi lebih tinggi dapat menggantikan tenaga kerja yang berkualifikasi lebih rendah secara teratur dan berurut. Selain itu juga diharapkan proses pemutusan hubungan kerja (PHK) pada industri tekstil tersebut dapat dihindarkan, karena semua tenaga kerja dapat bekerja secara bergiliran termasuk hari liburnya masing-masing. Pada penerapan program ini juga dilaksanakan pengembangan model dan perhitungan dengan komputer melalui operasi kerja pada *microcomputer*, hal ini dilakukan untuk mempersingkat waktu pemecahan masalah dengan hasil yang maksimal.

Dari data penelitian dan penerapan program tersebut dapat menghasilkan penjadwalan kerja dan hari libur secara rinci, dicapai kerja yang efisien dan biaya tenaga kerja optimal juga tidak perlu melakukan pemutusan hubungan kerja (PHK).

Kata kunci : pemograman integer, IP 1-5, model solver XA, industri tekstil, pertenunan

ABSTRACT

A research and the implementation of 1-5 Integer Programming (IP 1-5) with Solver XA Model has been carried out in one of textile industry, particularly weaving industry in Bandung City area. The using of this mathematical programming language refers to the efficiency of labour cost and the using of the lowest time. Thus the program can be used for that industry.

The aim of this research is to use that program to solve work scheduling problem so that working process will more efficient and effective as well as higher productivity. It is therefore that resulted program perform a process of substitution from an unskilled employee by a skilled employee regularly and orderly. Moreover, a disconnected employment can be avoided due to the employees can be worked in turn including their off days. A developing of model and its computation has been performed by using a microcomputer to shorten of problem solving which maximum result.

Based on the implementation of the program result a detail schedule including of the day-off, higher working efficiency and lower cost as well as disconnected employment can be avoided.

Key words **:** *Integer programming, IP 1-5, solver model XA, textile industry, weaving*

PENDAHULUAN

Sesudah masa krisis ekonomi, hampir seluruh industri tekstil khususnya pada industri pertenunan (*Weaving*) sampai sekarang memgalami kesulitan dalam pengelolaan sumber daya manusia yang dalam hal ini tenaga kerja. Para tenaga kerja dikejar dan dihantui oleh ancaman pemutusan hubungan kerja (PHK) yang kerap kali dilakukan oleh pihak perusaan tanpa pemberitahuan alasan yang jelas, karena banyaknya kendala dan hambatan yang besar baik teknis maupun non teknis. Secara kebetulan sebagai konsultan di salah satu pabrik pertenunan yang ada di kota Bandung tersebut, dalam hal ini pihak produsen (perusahaan) meminta dan mendesak untuk segera menanggulangi problema pemutusan hubungan kerja (PHK) yang sedang dihadapi [1].

Untuk memecahkan masalah diatas tersebut dan mendapatkan solusi yang baik dan tepat, maka diputuskan untuk menerapkan suatu pemograman matematika melalui solusi di *microcomputer* terarah sesuai dengan metoda pemecahan logis, yang selanjutnya digunakan penerapan melalui pemograman *integer* model *solver XA* seperti dalam pertemuan dan pembicaraan yang pernah dilakukan penulis bersama Mr. Lingsui dan Mr. Hung R. dan kawan-kawan beberapa tahun lalu (periode tahun 2008-2010), mereka semua ini pernah menyarankan kepada penulis untuk menggunakan pemograman *integer* (*Integer Programming*) dengan aplikasi

penggunaan bahasa pemograman *linear solver XA* [2].

Penjadwalan merupakan kumpulan kebijaksanaan dan mekanisme di sistem operasi yang berkaitan dengan urutan kerja yang dilakukan sistem komputer. Penjadwalan bertugas memutuskan proses yang harus berjalan, kapan, dan selama berapa lama proses itu berjalan. Sasaran utama penjadwalan adalah adil, efisien, waktu tanggap, *turn arround time*, dan *throughput*. Adil merupakan proses-proses diperlukan sama yaitu mendapat jatah waktu pemroses yang sama dan tak ada proses yang tak kebagian layanan pemroses. Sasaran penjadwalan seharusnya menjamin tiap proses mendapat pelayanan dari pemroses yang adil. Efisien merupakan utilitas pemroses dihitung dengan perbandingan (rasio) waktu sibuk pemroses. Sasaran penjadwalan adalah menjaga agar pemroses tetap dalam keadaan sibuk sehingga efisiensi mencapai maksimum.Terdapat dua kedaan waktu tanggap, yaitu waktu tanggap pada sistem interaktif dan waktu tanggap pada sistem waktu nyata. Waktu tanggap pada sistem interaktif merupakan waktu yang dihabiskan dari saat karakter terakhir dari perintah dimasukan atau transaksi sampai hasil pertama muncul dilayar (terminal). Sedangkan waktu tanggap pada sistem waktu nyata merupakan waktu dari kejadian (internal atau eksternal) sampai instruksi pertama rutin layanan yang dimaksud dieksekusi disebut *event response time*.

Sasaran penjadwalan adalah meminimalkan waktu tanggap. *Turn arround time* adalah waktu yang dihabiskan dari saat program atau *job* mulai masuk ke sistem sampai proses diselesaikan sistem. Waktu yang dimaksud adalah waktu yang dihabiskan di dalam sistem, diekspresikan sebagai penjumlah waktu eksekusi (waktu pelayanan *job*) dan waktu menunggu. Throughput adalah jumlah kerja yang dapat diselesaikan dalam satu unit waktu. Cara untuk mengekspresikan throughput adalah dengan jumlah job pemakai yang dapat dieksekusi dalam satu unit/interval waktu. Sasaran penjadwalan adalah memaksimalkan jumlah *job* yang diproses per satu interval waktu. Lebih tinggi angka troughput, lebih banyak kerja yang dilakukan sistem.

Dikatakan sebagai penjadwalan berprioritas apabila tiap proses diberi prioritas dan proses yang berprioritas tertinggi mendapat jatah waktu lebih dulu *(running)*. Berasumsi bahwa masing-masing proses memiliki prioritas tertentu, sehingga akan dilaksanakan berdasar prioritas yang dimilikinya. Ilustrasi yang dapat memperjelas prioritas tersebut adalah dalam komputer militer, dimana proses dari jendral berprioritas 100, proses dari kolonel 90, mayor berprioritas 80, kapten berprioritas 70, letnan berprioritas 60 dan seterusnya. Dalam *UNIX* perintah untuk mengubah prioritas menggunakan perintah *nice.*

Sedangkan penjadwalan terjamin memberikan janji yang realistis (memberi daya pemroses yang sama) untuk membuat dan menyesuaikan performance. Jika ada N pemakai, sehingga setiap proses (pemakai) akan mendapatkan 1/N dari daya pemroses CPU. Untuk mewujudkannya, sistem harus selalu menyimpan informasi tentang jumlah waktu CPU untuk semua proses sejak login dan juga berapa lama pemakai sedang login. Kemudian jumlah waktu CPU, yaitu waktu mulai login dibagi dengan n, sehingga lebih mudah menghitung rasio waktu CPU. Karena jumlah waktu pemroses tiap pemakai dapat diketahui, maka dapat dihitung rasio antara waktu pemroses yang sesungguhnya harus diperoleh, yaitu 1/N waktu pemroses seluruhnya dan waktu pemroses yang telah diperuntukkan proses itu. Rasio 0,5 berarti sebuah proses hanya punya 0,5 dari apa yang waktu CPU miliki dan rasio 2,0 berarti sebuah proses hanya punya 2,0 dari apa yang waktu CPU miliki. Algoritma akan menjalankan proses dengan rasio paling rendah hingga naik ketingkat lebih tinggi diatas pesaing terdekatnya. Ide sederhana ini dapat diimplementasikan ke sistem *real-time* dan memiliki penjadwalan berprioritas dinamis. Selama survai dan mempelajari teori pemograman bersama rekan-rekan konsultan, maka dapalah dipilih model integer yang aplikatif mendekati *Turbo Pascal* yang cukup tepat, cocok dan efisien dalam proses pemecahan masalah penjadwalan tenaga kerja. Sedang untuk pelaksanaan operasi proses cukup menggunakan metode *solver XA*.

Adapun tujuan penelitian ini selain untuk mengembangkan model *pemograman* dalam penjadwalan tenaga kerja pada proses produksi dalam industri Tekstil yang mengacu pada model formulasi MPL, dan untuk menyelesaikan model tersebut menggunakan *software* optimasi metode *solver XA*. sehingga dapat menghasilkan jadwal dan pengaturan jadwal libur yang optimum, juga hasil dari penerapan program ini dapat menemukan biaya terendah tenaga kerja. Pada penelitian ini terbagi atas tiga tahap yaitu pertama adalah pengolahan data yang terdiri atas penyusunan *state-task network* (*STN*), penetapan fungsi tujuan, batasan-batasan dan variabel keputusan, pembuatan model MPL kedalam *software* optimasi. Tahap kedua adalah uji validasi model dengan *software Solver XA* dan tahap akhir adalah analisa solusi optimal. Dari solusi optimal yang didapat akan ditransformasikan menjadi jadwal tenaga kerja pada proses produksi

Pada sistem penjadwalan didekati melalui formula (IP 1 – 5) dengan menggunakan formulasi MPL. MPL ini merupakan suatu model menuntun para *programmer* untuk menyelesaikan model *linear programming (LP)* yang sulit, menyelesaikan ribuan peubah (*variable)* dan *constraint* secara jelas, tepat dan nampaknya cukup efisien. Peubah (*Variable*) pada *linear programming (LP)* mungkin berkesinambungan atau *integer*, dan untuk solusai optimal pelaksanaannya menggunakan metode solver XA [3dan 4].

Atas dasar alasan dan pemikiran lanjut, dalam hubungan penerapan di lapangan industri tekstil khususnya pertenunan telah dilakukan suatu upaya melalui sistem perhitungan formulasi 10

(sepuluh) [5] baris dan solusi waktu pemecahan yang di pakai pada *microcomputer* direncanakan memakan waktu kurang lebih selama 1 menit. Pada saat penulisan baris program di perangkat lunak (*software*), tidak digunakan baris-baris program yang khusus maupun cabang-cabangnya, hal ini dilakukan karena dengan sistem yang sederhana ini sudah cukup aplikatif dan diharapkan dapat mencapai hasil yang maksimal dalam menghasilkan urutan pengaturan jadwal kerja dari tenaga kerja pada industri pertenunan.

METODOLOGI

Agar pembahasan lebih bisa dimengerti, maka akan dijelaskan terlebih dahulu pengertian tentang pemograman. Pemograman berasal dari kata program yang berarti deretan instruksi yang digunakan untuk mengendalikan program komputer. Program dibuat menggunakan Bahasa Pemrograman (*programming language*) Kosa kata atau aturan-aturan gramatik untuk memberikan perintah kepada komputer untuk melaksanakan suatu tugas disebut *syntax*. Istilah bahasa pemrograman biasanya mengacu kepada bahasa tingkat-tinggi, seperti *BASIC, C, C++, COBOL, FORTRAN, Ada*, dan *Pascal*.

Klasifikasi Bahasa Pemrograman adalah sebagai berikut :

- *Generasi pertama.*
  Bahasa yang berorientasi pada mesin, dimana program disusun menggunakan bahasa mesin/kode mesin. Bahasa Mesin adalah bahasa tingkat rendah yang hanya dipahami oleh komputer. Bahasa mesin ini sangat sulit dipahami oleh orang awam sehingga programmer harus menguasai operasi komputer secara teknis. Abstraksi bahasa ini adalah kumpulan kombinasi kode biner "0" dan "1" yang sangat tidak alamiah bagi kebanyakan orang - kecuali insinyur pembuat mesin komputer. Karena tidak alamiah bagi kebanyakan orang, bahasa mesin juga disebut bahasa tingkat rendah.
- *Generasi Kedua.*
  Bahasa pemrograman yang menggunakan bahasa rakitan / *Assembly*. Bahasa *Assembly* adalah bahasa pemrograman yang menggunakan instruksi yang sama seperti pada bahasa mesin, tetapi instruksi dan variable yang digunakan mempunyai nama sehingga mempermudah proses pemrograman. Prosesnya tidak lagi menggunakan deretan kode *biner* untuk melakukan pemrograman.
- *Generasi Ketiga* .
  Bahasa pemrograman yang menggunakan pendekatan prosedural. Instruksi program ditulis menggunakan kata-kata yang biasa digunakan oleh manusia. Contoh : *WRITE* (untuk menampilkan kelayar), *READ* (untuk membaca data masukan dari *keyboard*). Bahasa pada generasi ini disebut juga Bahasa beraras tinggi / *High Level Language*. Contoh bahasa pemrogaman :*PASCAL, FORTRAN, C, COBOL, BASIC* dll.
  Pada generasi bahasa pemrograman terakhir sekarang ini, kedua cara interpretasi dan kompilasi digabungkan dalam satu lingkungan pengembangan terpadu (IDE = *integrated development environment*). Cara interpretasi memudahkan dalam pembuatan program secara interaktif dan cara kompilasi menjadikan eksekusi program lebih cepat.
  Pertama program dikembangkan interaktif, kemudian setelah tidak ada kesalahan keseluruhan program dikompilasi. Contoh bahasa program seperti ini adalah *Visual BASIC* yang berbasis BASIC dan Delphi yang berbasis *PASCAL*.
  Bahasa tingkat tinggi bersifat *portable*. Program yang dibuat menggunakan bahasa tingkat tinggi pada suatu mesin komputer bersistem operasi tertentu, hampir 100% bisa digunakan pada berbagai mesin dengan aneka sistem operasi. dan kalaupun ada perbaikan sifatnya kecil sekali.
- *Generasi keempat*.
  Merupakan Bahasa Non-Prosedural. Bahasa pemrograman Generasi Ke-4 dirancang untuk mengurangi waktu pemrogram untuk membuat program sehingga pembuatan program dibuat dengan waktu lebih cepat. Program ini dapat digunakan oleh pemakai yang kurang mengenal hal-hal teknis pemrograman tanpa perlu bantuan seorang programmer professional.
  Contoh : Membuat program database sederhana dengan Microsoft Access. Bahasa generasi ke-4 disebut juga dengan *Very High Level Language atau Problem Oriented Language* (bahasa yang berorientasi pada masalah) karena memungkinkan pemakai menyelesaikan masalah dengan sedikit penulisan kode pemrograman dibandingkan dengan bahasa prosedural.
  Fasilitas yang tersedia :
  – Program Generator (untuk membuat aplikasi mudah).
  – Report Generator (untuk membuat laporan dengan mudah dan cepat)
  – Bahasa Query (SQL).
  Dengan adanya fasilitas ini pekerjaan programmer menjadi lebih sedikit dalam menuliskan kode instruksi.
  Contoh Bahasa Generasi ke-4: *Oracle, Microsoft Access* dsb.
- *Generasi Ke Lima.*
  Merupakan bahasa pemrograman yang ditujukan untuk menangani kecerdasan buatan (*artificial intelligence*) (AI). AI adalah disiplin dari ilmu komputer yang mempelajari cara komputer meniru kecerdasan manusia.
  Contoh Aplikasi :
  - Pemrosesan Bahasa Alami è mengatur komputer agar bisa berkomunikasi dengan manusia melalui bahasa manusia.
  - Aplikasi Sistem Pakar è program komputer yang dapat menghasilkan pemikiran yang

setara dengan seorang pakar. Contoh Bahasa Pemrograman : *PROLOG* dan *LISP*.

Adapun untuk pembagian tingkatan Bahasa Pemrograman ada 3 (tiga) macam, yaitu :

- Bahasa Tingkat Rendah (*Low Level Language*), bahasa yang menggunakan bahasa mendekati bahasa mesin, contoh : bahasa *Assembly.*
- Bahasa Tingkat Menengah (*Middle Level Language*), bahasa pemrograman yang menggunakan aturan-aturan (sintax) dalam penulisan pernyataannya yang mudah dipahami dan memiliki instruksi/sintax tertentu yang dapat langsung diakses oleh komputer, contoh : Bahasa C.
- Bahasa Tingkat Tinggi (*High Level Language*), bahasa pemrograman yang penulisan pernyataannya (sintax) mudah dipahami secara langsung oleh manusia. (karena sintax nya menggunakan bahasa manusia), contoh dalam *PASCAL* : *WRITE* (untuk menampilkan ke layar). Bahasa Pemrograman tingkat Tinggi terdiri dari dua kelompok Bahasa, yaitu:.
  - *Procedure Oriented Language*. Terdiri dari Scientific/Science (Masalah Ilmiah) yang digunakan untuk memecahkan persoalan matematik/ alamiah (contoh : *PASCAL, FORTRAN, BASIC*) dan Bussiness yang digunakan untuk masalah bisnis (contoh : *Cobol, PL/1*).
  - *Problem Oriented Language,* contoh : *RPG (Report Program Generator*)

Dalam tulisan yang berhubungan dengan penjadwalan, program lebih mengarah ke *TURBO PASCAL* dan untuk lebih memahami lagi maka akan dikaji kembali dengan pengertian *INTEGER*.
INTEGER adalah *INTEGRAL* yang merupakan kebalikan dari differensial (anti differensial). Jika turunan dari F(x) adalah f(x), maka :
(c = konstanta)⇒ f(x) dx = F(x) + c ∫
Integral dapat digolongkan atas :
A. Integral tak tentu (tanpa batas).
B. Integral tertentu (dengan batas)

Rumus-Rumus Integer tertentu

1. Rumus

Fungsi Aljabar
-1≠ xn dx = 1/n+1 xn+1 + c ; n ∫

Fungsi *trigonometri*
sin x dx∫ = - cos x + c
cos x dx = sin x + c ∫

Sifat-sifat:
f(x) dx ∫c f(x) dx = c ∫a.
g(x) dx ∫f(x) dx ± ∫(f(x) ± g(x)) dx = ∫b.
jika f(x) dx = F(x) + c ∫ ,
maka 1/a F(ax) + c = f(ax) dx ∫
1/a F(ax+b) + c = f(ax+b) dx ∫

Perluasan :
(ax + b)n dx = 1/a 1/(n+1) (ax + b)n+1 + c ∫
sin (ax + b) dx = -1/a cos (ax + b) + c ∫
cos (ax + b) dx = 1/a sin (ax + b) + c ∫

Cara mengintegrir
a. Substitusi
f(x) dx ∫I =
substitusi : x = Q(u) ; dx = Q`(u) du
f(Q(u)) Q`(u) du ∫I =

Jika ruas kanan telah diintegrir, subtitusi kembali dengan fungsi invers dari x = Q(u) (keterangan : Prinsipnya adalah merubah variabel sehingga rumus dapat digunakan)

b. Substitusi *trigonometri*
a2 - x2√1. Bentuk = arc sin x/aθ → θ
misalkan x = a sinθ dθdx = (a cos)θ dθ (a
cos θ 1 - sin2√ ∫ a2 - x2 dx
= a √ ∫θ dθcos2 ∫= a2 θ) dθ(1 + cos2 ∫
= ½a2) + cθ cosθ + sinθ
= ½a2 (a2 - x2] + c√[arc sin x + x ∫
= ½a2 a a a
a2 - x2 + c√ a2 - x2 dx
= ½ a2 arc sin x/a + ½ x √ ∫a2 + b2x2√ ∫2.

Bentuk θ Gunakan substitusi : x = a/b tg θ
dθdx
= a/b sec2
b2x2 - a2√ ∫3.
Bentuk θ Gunakan substitusi : x = a/b secθ
sec2θdx = a/b tg

c. Parsiil
Yaitu mengenai integral dari suatu bentuk yang merupakan hasil perkalian antara suatu fungsi x dengan turunan dari suatu fungsi x yang lain.

I f(x) g(x) dx ∫=
Misalkan : u = f(x) ; dv = g(x) dx
du = ... dx ; g(x) dx = ... maka : ∫v =
v du ∫u du = u v - ∫v du jadi lebih
mudah ∫□ .
Pemisalan dibuat sedemikian sehingga bentuk untuk hal-hal khusus dapat digunakan cara TABULASI

Integer tertentu

1. Pengertian
Bila suatu fungsi F(x) mempunyai turunan f(x), maka bila f(x) diintegrasikan pada selang (a, b) menjadi
a a = F(b) - F(a)⏐ c dx = c(x) ∫ b b

2. Sifat

a. a = c(b - c) ⏐ c dx = c(x) ∫ c =konstanta a a b a
b. f(x) dx∫ f(x) dx = - ∫ c = batas ditukar a b a
c. f(x) dx =∫ 0 c = batas sama ab a b
d. f(x) dx∫ f(x) dx = ∫ f(x) dx∫+c = (a<c<b) a b c

Pembatasan dan formulasi masalah [6]

Untuk pembahasan dan memformulasikan pemecahan masalah yang sedang dihadapi, maka sebelumnya disiapkan dan diberikan suatu kondisi syarat dan model sebagai berikut :

1. Tenaga kerja staff yang bekerja dengan jumlah hari kerja selama seminggu atau 7 hari yaitu hari Senin sampai dengan hari Minggu, diberikan notasi hari dengan angka 1, 2, 3, 4, 5, 6, dan 7.
2. Semua tenaga kerja dengan waktu penuh dan diklasifikasikan ke dalam tipe *m* (jumlah tenaga kerja), dengan tipe 1 mempunyai kualitas paling bagus, tipe 2 mempunyai kualitas paling bagus berikutnya. Biaya dari tipe *k* tenaga kerja adalah c_k dan $c_1 > c_2 > \ldots > c_m$. Biaya ini dimasukkan kedalam jumlah perhitungan hari libur yang diambil oleh tenaga kerja setiap minggunya;
3. Untuk setiap harinya tenaga kerja memerlukan sejumlah tipe 1 pekerjaan untuk dikerjakan, juga tipe 2 pekerjaan untuk dikerjakan, …, dan tipe *m* pekerjaan untuk dikerjakan. Spesifikasinya, d_{lj}; dimana pekerjaan pada tipe 1 harus dikerjakan pada hari *j, j* = 1,2, …, 7. Setiap pekerjaan yang selesai diperlukan satu tenaga kerja dan tipe 1 pekerjaan dapat dikerjakan dengan tipe *k* tenaga kerja dimana $l \geq k$ (tenaga kerja yang berkualitas lebih tinggi dapat menggantikan tenaga kerja yang berkualitas lebih rendah, tetapi tidak sebaliknya).
4. Setiap tenaga kerja harus menerima *n* hari libur setiap minggunya, dimana n = 2, 3, 4 untuk 5 hari, 4 hari, dan 3 hari kerja seminggu secara berurut. Untuk lebih jelasnya apa yang dimaksud ini dapat disajikan seperti pada Tabel 1 berikut.

Tabel 1. Hubungan Jumlah Hari libur dan Jumlah Hari kerja

Jumlah Hari Libur (n)	Jumlah Hari Kerja	Total Hari
2	5	7
3	4	7
4	3	7

Model pemograman integer dan formulasi MPL

Untuk mendefinisikan suatu peubah (*variable*) keputusan, sebagai dasar diambil notasi $W_{k.}$ dimana nilai w_k (k = 1 … m) dan ini merupakan jumlah tenaga kerja yang bertipe *k* dan selanjutnya $x_{k.l.j}$ jumlah tenaga kerja bertipe *k* (k = 1 …m) dengan menentukan pekerjaan bertipe *1* ($1 \in$ {1 …m}, $1 \geq k$) pada hari *j*,*j* = 1, 2, …., 7. Dimana $y_{k.j}$ menjadi jumlah dari tenaga kerja bertipe *k*(k = 1, …, *m)* kemudian *j* dianggap hari libur (j = 1, …, 7).

Selanjutnya untuk perhitungan biaya minimal tenaga kerja yang harus dibayar dan penjadwalan yang sesuai dapat ditentukan dengan penyelesaian pemograman *integer* (*Integer Programming/IP*) [7] sebagai berikut :

(IP min) $\quad 1) \ \Sigma\, c_k w_k$

$$\sum_{k=1.m} x_{k.l.j} + y_{k.j} = w_k \quad (k = 1 \ldots m, j = 1, \ldots 7),$$

$$\sum_{k=l.m} x_{k.l.j} + y_{k.j} = w_k \quad (k = 1 \ldots m, j = 1, \ldots 7),$$

$$\sum_{j} y_{k.j} \geq w_k n \quad (k = 1, \ldots m),$$

$$\sum_{k<l} x_{k.l.j} = d_{l.j} \quad (l = 1, \ldots m, j = 1, \ldots 7),$$

$x_{k.l.j}, y_{k.j}, w_k$ integer (k=1,…,m,l=k,…..m, j=1, …,7),

Pelaksanaan solusi optimal dari (IP1) telah dihitung, selanjutnya akan lebih mudah menemukan penjadwalan. Kita ketahui $y_{k.j.}$ adalah jumlah tenaga kerja bertipe *k* (k = 1, …, *m*) dimana *j* dianggap hari libur. Sesudah menjadwalkan hari libur pada masing-masing tenaga kerja dari setiap tipe. Untuk memberikan tipe pertama-tama kita ambil 1 hari libur untuk setiap tenaga kerja, selanjutnya 2 hari libur untuk setiap tenaga kerja dan seterusnya. Pada masalah ini, setiap tenaga kerja bertipe *k* akan menerima sedikitnya *n* hari libur, dan hal ini dimungkinkan terjadi pada jumlah total dari hari libur ditempatkan pada tenaga kerja bertipe $k(\Sigma_j y_{k.j})$ yang lebih besar atau sama dengan jumlah tenaga kerja bertipe k (w_k) dikalikan dengan jumlah hari libur yang harus diterima tenaga kerja setiap minggu (*n*) disebabkan oleh bagian ke 2 dari (IP1).

Contoh 1 : m = 3 (terdapat 3 tipe tenaga kerja), n = 2 (5 hari kerja dalam seminggu), c_1 = 12, c_2 = 8, c_3 = 6 dan selanjutnya pada Tabel 2 merupakan rincian jumlah pekerja yang bekerja pada hari j.

Tabel 2. Jumlah Pekerja yang Bekerja pada Hari j

	Jumlah Orang						
	Senin	Selasa	Rabu	Kamis	Jum'at	Sabtu	Minggu
d_{1j}	2	3	1	4	2	5	5
d_{2j}	2	1	2	2	2	1	1
d_{3j}	4	4	4	1	6	4	4

Solusi optimal dari (IP1) menghasilkan nilai w_k *dan* $y_{kj,}$ w_1 = 5, w_2 = 2, w_3 = 5 dan selanjutnya pada Tabel 3 merupakan rincian jumlah pekerja yang libur pada hari j [8].

Tabel 3. Jumlah Pekerja yang Libur pada Hari j

	Jumlah Orang						
	Senin	Selasa	Rabu	Kamis	Jum'at	Sabtu	Minggu
y_{1j}	3	1	4	1	1	0	0
y_{2j}	0	2	0	0	0	1	1
y_{3j}	1	1	1	4	1	1	1

Untuk selanjutnya penjadwalan contoh 1 dengan solusi optimal (IP1). Setiap hari, setiap tenaga kerja diberikan tenaga kerjaan dari level 1 (1) atau hari libur (x), dan hasil penjadwalan tersebut disajikan seperti pada Tabel 4 berikut.

Tabel 4. Hasil Penjadwalan Contoh 1

		Senin	Selasa	Rabu	Kamis	Jum'at	Sabtu	Minggu
K=1	1	X	1	X	1	1	1	1
	2	X	1	X	1	1	1	1
	3	X	1	X	1	2	1	1
	4	1	X	1	X	2	1	1
	5	1	2	X	1	X	1	1
K=2	1	2	X	2	2	3	X	2
	2	2	X	2	2	3	2	X
K=3	1	X	3	3	X	3	3	3
	2	3	X	3	X	3	3	3
	3	3	3	X	3	X	3	3
	4	3	3	3	X	3	X	3
	5	3	3	3	X	3	3	X

Selanjutnya diberikan contoh lain dalam penyelesaian *solusi optimal IP1*, yaitu Contoh 2 : m = 3 (terdapat 3 tipe tenaga kerja), n = 2 (5 hari kerja dalam seminggu), $c_1 = 12$, $c_2 = 8$, $c_3 = 6$ dan selanjutnya pada Tabel 5 berikut ini merupakan rincian jumlah pekerja yang bekerja pada hari j.

Tabel 5. Jumlah Pekerja yang Bekerja pada Hari j

	Jumlah Orang						
	Senin	Selasa	Rabu	Kamis	Jum'at	Sabtu	Minggu
d_{1j}	4	0	5	3	3	3	3
d_{2j}	0	3	5	4	5	2	2
d_{3j}	5	0	4	2	0	0	1

Solusi optimal dari (IP1) menghasilkan nilai w_k *dan* y_{kj}, $w_1 = 5$, $w_2 = 5$ $w_3 = 4$ dan selanjutnya pada Tabel 6 merupakan rincian jumlah pekerja yang libur pada hari j.

Tabel 6. Jumlah Pekerja yang Libur pada Hari j

	Jumlah Orang						
	Senin	Selasa	Rabu	Kamis	Jum'at	Sabtu	Minggu
y_{1j}	1	5	0	2	0	2	2
y_{2j}	0	2	0	1	2	3	2
y_{3j}	4	4	0	2	4	4	4

Untuk selanjutnya penjadwalan contoh 2 dengan solusi optimal (IP1). Setiap hari, setiap tenaga kerja diberikan tenaga kerjaan dari level 1 (1) atau hari libur (x), dan hasil penjadwalan tersebut disajikan seperti pada Tabel 7 berikut.

Tabel 7. Hasil Penjadwalan Contoh 2

		Senin	Selasa	Rabu	Kamis	Jum'at	Sabtu	Minggu
K= 1	1	1	X	1	1	1	1	X
	2	1	X	1	X	1	1	X
	3	1	X	1	X	1	1	1
	4	1	X	1	1	2	X	1
	5	1	X	1	1	2	X	1
K= 2	1	3	X	2	2	2	X	2
	2	3	X	2	2	2	X	2
	3	3	2	2	X	2	X	3
	4	3	2	2	2	X	2	X
	5	3	2	2	2	X	2	X
K= 3	1	X	X	3	X	X	X	X
	2	X	X	3	X	X	X	X
	3	X	X	3	3	X	X	X
	4	X	X	3	3	X	X	X

Penjadwalan tenaga kerja [9]

```
TITLE       PENJADWALAN HIRARKHI TENAGA KERJA;
DATA        m = 3;n = 2;
INDEX       k = 1..m;l = 1..m;j = 1..7;
DATA        X[k, l, j] where l ≥ k; y[k,j]; w[k];
DECISION    d[l, j] = (4,0,5,3,3,3,3,0,3,5,4,5,2,2,5,0,4,2,0,0,1);
            c[k] = (12,8,6);
MODEL       min sum(k:c*w);
SUBJECT TO  C1[k,j]; sum (l : x) + y
            = w; C2[k]: sum(j : y) ≥ w*n C3[l,j]: sum(k:x)=d;
INTEGER     w,x,y;
END
```

HASIL DAN PEMBAHASAN

Hasil perhitungan komputer [10 dan 11]

Dari hasil perhitungan dan penjelasan tentang IP1, dilanjutkan dengan pengujian (Uji Model) untuk 48 *random* sekaligus memecahkan masalah dengan jumlah tipe *range* antar 2 – 5 dan jumlah hari libur mempunyai *range antar* 2 – 4. d_{lj} diacak dan didistribusi antar 0 – 8 dan *vector* **c** adalah (12,8,6,5,2). Untuk setiap nilai dari pasangan {jumlah tipe, jumlah hari libur} diselesaikan 4 masalah dengan menggunakan bahasa permodelan MPL dan LP – IP *solver* XA. Tabel 8 berikut memberikan suatu informasi tentang nilai rata-rata, untuk 4 permasalahan, dari perbedaan antara LP dan IP optimal dan nilai rata-rata, dari waktu yang diperlukan untuk menghitung IP optimal.

Hasil yang diperoleh menunjukkan fakta, bahwa 46 IP dari 48 dapat diselesaikan kurang dari 1 menit pada *microcomputer*. Dua masalah diperlukan lebih dari 1 menit ; satu masalah dengan m = 4 dan n = 3 memerlukan waktu 2 menit 30 detik dan satu masalah lagi dengan m = 5 dan n = 3 memerlukan waktu 3 menit 6 detik. Dari kondisi yang dapat dicapai ini, membuktikan bahwa metoda yang telah dilakukan merupakan metoda yang cukup cepat dalam pemecahan masalah, juga *integer programming* merupakan pendekatan yang cukup baik; implementasinya sangatlah mudah dan solusi waktunya masuk diakal pada semua *microcomputer.*

Seperti tersaji pada Tabel 8, bahwa batas relatif antara nilai optimal dari relaksi kontinyu dari (IP1) dan nilai optimum dari (IP1) menunjukkan hasil perhitungan waktu dan detik juga menghasilkan perhitungan optimum dari (IP1) selalu kecil (kurang dari 2 %). Semua ini memberikan arti dan penjelasan bahwa tercapainya efisiensi dengan pendekatan pemograman *integer* (*Integer Programming*).

Tabel 8. Relative Gap dalam Prosentase antara Nilai Optimum dari Relaksasi Kontinyu dari (IP1) dan Nilai Optimum dari (IP1)

	Libur		
Tipe	**2 Hari**	**3 Hari**	**4 Hari**
2	0 %	1.4 %	1.5 %
	0.61 s	1.75 s	1.87 s
3	0.6 %	1.5 %	1.2 %
	1.65 s	6.08 s	4.51 s
4	0.5 %	2 %	1.4 %
	4.91 s	63.80 s	25.53 s
5	0.3 %	1.4 %	1.12 %
	5.43 s	72.93 s	19.56 s

Pengembangan dari model

- Kualifikasi dan Penjadwalan

Untuk pembahasan lebih lanjut, model dapat disaring dan sebagai contoh dalam hal ini dapat dicari perhitungan biaya optimal tenaga kerja dengan cara meminimalkan jumlah tenaga kerja yang ditetapkan setiap hari untuk bekerja yang mana kualifikasinya lebih rendah dibandingkan dengan kualifikasi mereka. Pemograman *integer* (*Integer Programming*) yang sesuai (IP2) diperoleh dari (IP1) dengan menambahkan fungsi objektif kedua**,** $\varepsilon \sum^{m-i}_{k=1}\sum^{7}_{j=1}\sum^{m}_{l=k-1} X_{k,l,j,}$ yang dihitung kedalam jumlah tenaga kerja dan ditetapkan setiap harinya pada tipe Tenaga Kerjaan yang kualifikasinya lebih rendah daripada kualifikasi mereka. Koefisien ε harus dipilih yang nulainya kecil. Modifikasi untuk menjalankan formula MPL harus dilakukan secepatnya, dan perhitungan secara eksperimen dapat dilihat bahwa (IP2) yang dapat diselesaikan dengan waktu perhitungan yang sama seperti yang telah dilakukan dengan (IP1). Dari hasil perhitungan solusi optimal dari (IP2) menunjukkan penyelesaian waktu yang tidak jauh berbeda dengan yang dilakukan ole IP1, dan hal ini juga dengan mempertimbangkan lagi contoh 2 dari sesi solusi optimal dari (IP2) yang memberikan hasil $w_1 = 5$, $w_2 = 5$, $w_3 = 4$ (lihat Tabel 6).

Penjadwalan yang sesuai diberikan sesuai dengan solusi tersebut, menghasilkan biaya minimal yang dapat diberikan kepada satu tenaga kerja yang mempunyai kualitas kerja rendah: tenaga kerja no 5 tipe 2 bekerja pada level 3 pada hari senin. Pada solusi ditampilkan pada Tabel 2, 8 tenaga kerja bekerja pada level terendah dibandingkan dengan kualifikasi mereka.

- Penjadwalan hari libur

Kita telah melihat bagaimana membuat hari libur untuk masing-masing tenaga kerja dengan metoda yang sangat sederhana. Pada umumnya, ketika nilai optimum $y_{k,j}$ *(*k = l…m : j = 1…7) sudah ditentukan, maka penjadwalan hari libur beberapa orang dapat dilaksanakan dan dapat terjadi. Metoda yang telah dilakukan memberikan suatu pengertian bahwa tidak memperbolehkan untuk menentukan dan memilih "jadwal terbaik" diantara semua kemungkinan yang terjadi. Pada penjelasan berikut ini dapat diperlihatkan bagaimana formulasi pemograman *integer* memilih jadwal hari libur yang cukup menarik, dimana hari libur untuk tenaga kerja diberikan berdasarkan nilai maksimum dari hari libur yang berurutan. Dari kondisi ini cukup jelas, bahwa untuk setiap tipe dari tenaga kerja dapat dipelajari secara bebas yang selanjutnya harus dipertimbangkan disini hanya satu tipe tenaga kerja. Kasus dimana setiap tenaga kerja harus menerima 2 hari libur setiap minggunya dan didefinisikan mengikuti nilai 0 – 1 peubah (*variable*) :

$t_{i,j}$ = 1 jika dan hanya jika j hari dan $j + 1$ mod 7 adalah hari libur untuk tenaga kerja i ;

$u_{i,j}$ = 1 jika dan hanya jika j hari adalah hari libur untuk tenaga kerja i tapi $j + 1$ mod 7 bukan hari libur untuk tenaga kerja.

Data adalah N, jumlah tenaga kerja v_j, dan jumlah dari tenaga kerja yang mengambil hari libur adalah j. Solusi dari pemograman *integer* memberikan jadwal dari hari libur dengan maksimum hari libur yang berurutan.

$$\text{(IP3)} \quad \max \sum_{i=1\ldots N} \sum_{j=1\ldots 7} t_{i,j}$$

$$\text{s.t} \quad \sum_{i=1\ldots N} u_{i,j} + t_{i,j-1} + t_{i,j-1} = v_j \quad (i = 1,\ldots,7) \qquad (1)$$

$$\sum_{j=1\ldots 7} u_{i,j} + 2t_{i,j-1} = 2 \quad (i = 1,\ldots,N) \qquad (2)$$

$$u_{i,j}\,,\ t_{i,j} \in \{0,1\} \quad (i = 1,\ldots,N;\ j = 1,\ldots,7) \qquad (3)$$

Setiap tenaga kerja menerima 2 hari libur dan diasumsikan jumlah total hari libur adalah **2N** ($\Sigma_{j=1..7} \quad v_j = 2N$)**.** Fungsi utamanya untuk mengukur jumlah pasangan dari urutan hari libur. Batasan (1)
menentukan hari j , jumlah hari libur tenaga kerja sama dengan v_j (jika $j = 1$ maka $j - 1 = 7$), sedangkan batasan **(2)** menentukan dari setiap tenaga kerja N tepatnya dua hari libur. Dengan mempertimbangkan contoh 2 dari sesi 3 dan jadwal dari hari libur pada tenaga kerja tipe ke-2 akan memperoleh maksimum hari libur yang berurutan.

Seperti dapat disajikan pada Tabel 9 diperlihatkan penjadwalan, dimana semua tenaga kerja kecuali tenaga kerja no. 1 dan 3 menerima dua hari libur yang berurutan. Solusi ini ditunjukkan juga pada Tabel 3, tenaga kerja bertipe 2 mempunyai hari libur yang berurutan dan solusi ini menunjukkan pada tabel 5 tenaga kerja bertipe 2 no. 4 dan 5 tidak mempunyai hari libur yang berurutan. Kita dapat menandai pada akhir solusi bahwa beberapa tenaga kerja bertipe 2 menerima lebih dari 2 hari libur. Solusi optimal ditentukan dengan (IP1) atau (IP2) yang mungkin dengan $\Sigma_j\, y_{k,j} \geq w_k\, n$ untuk beberapa nilai k. Untuk kasus ini beberapa tipe k tenaga kerja akan menerima lebih dari n hari libur. Untuk jadwal hari libur pada kasus ini kita selesaikan dengan program (IP4) yang mana pembatas (1) dari (IP3) diganti dengan $\Sigma_{i=1}^{n} H_{i,j} + t_{i,j} - 1 + t_{i,j} \geq v_j$. Tabel 10 memberikan jadwal hari libur untuk tenaga kerja bertipe 2 diberikan dengan solusi dari (IP2) (lihat Tabel 5).

Pada kasus Tabel 10 setiap tenaga kerja menerima dua hari libur secara berurutan, dan untuk atribut hari Kamis sebagai satu hari libur untuk lima tenaga kerja dan Sabtu sebagai hari libur untuk dua tenaga kerja yang terpilih diantara 1, 2, 3 dan 4. Sekali hari libur dapat diperbaiki untuk setiap tenaga kerja (beberapa tenaga kerja menerima lebih dari dua hari libur) penjadwalan seluruhnya diperhitungkan.

Pada beberapa kasus kita harus memilih tenaga kerja bertipe-k untuk bekerja pada level $l \geq k$,dan berikut ini merupakan formulasi MPL untuk programming *integer programming* (IP3).

Tabel 9. Contoh Jadwal Hari Libur yang Mempunyai Maksimum Hari Libur Secara Berurutan untuk Solusi (IP1)

		Senin	Selasa	Rabu	Kamis	Jum'at	Sabtu	Minggu
K=2	1	3	X	2	2	2	2	X
	2	3	2	2	2	X	X	2
	3	3	X	2	2	2	X	2
	4	3	2	2	X	X	2	3
	5	3	2	2	2	2	X	X

Tabel 10. Contoh Jadwal Hari Libur yang Mempunyai Maksimum hari Libur Secara Berurutan untuk Solusi (IP1)

		Senin	Selasa	Rabu	Kamis	Jum'at	Sabtu	Minggu
K=	1	X	X					
2	2	X						X
	3	X	X					
	4	X						X
	5						X	X

```
TITLE        PERENCANAAN 2 HARI LIBUR;
DATA         N = 5;
INDEX        i = 1 ..N, j = 1 ..7;
DATA         v[j] = (0,2,0,1,2,3,2);
DECISION     t[I,j];u[I,j];
MODEL        max sum (i.j:t);
SUBJECT TO
      Cl [j > l]  sum(i:u[i,j]+t[i,j:= j-l]=v[j]
          C2 :    sum(i:u[i,j=l]+t[i,j:= 1]=v[j:=1];
       C3 [i] :   sum(j: 2t + u) = 2;
BINARY       t,u;
END
```

Percobaan disini melibatkan perhitungan untuk sejumlah tenaga kerja 30 orang, dan proses penyelesaian ditentukan oleh solusi optimal dari (IP3) kurang dari 30 detik waktu CPU (lihat Tabel 11).

Masalah penjadwalan hari libur dapat diformulasikan ke dalam beberapa cara, tapi sejauh pengalaman penulis dalam penerapan pemograman nampaknya *integer programming* merupakan model pemecahan yang tepat dalam hal penjadwalan tenaga kerja. Seperti sudah diketahui kebanyakan algoritma *integer programming* memerlukan sedikit batasan pada nilai fungsi utama dan efisiensi algoritma sangat bergantung kepada ketajaman dari batasan itu sendiri. Batasan yang rendah sering kali diperoleh dari relaksasi kontinu dari pemograman *integer*, oleh karena itu untuk pemecahannya digunakan *software* (*XA Solver*).

Untuk pemecahan masalah penjadwalan hari libur, formula sebelumnya lebih baik dari yang berikut ini yang nampak lebih natural. Pencarian untuk penugasan hari libur yang memaksimalkan urutan hari libur dapat diformulasikan dengan mengikuti kuadratik program 0 – 1 yang mana peubah (*variable*) *Boolean* f_{i-j} = 1 jika dan hanya jika j adalah hari libur untuk tenaga kerja i.

$$\text{(QIP)}\quad \max \sum_{i=1..N}\sum_{j=1..7} f_{i,j}$$

$$\text{s.t}\quad \sum_{i=1..N} u_{i,j} + t_{i,j-1} + t_{i,j-1} = v_j \quad (i = 1,\ldots,7) \qquad (1)$$

$$\sum_{j=1..7} u_{i,j} + 2t_{i,j-1} = 2 \quad (i = 1,\ldots,N) \qquad (2)$$

$$u_{i,j},\ t_{i,j} \in \{0,1\} \quad (i = 1,\ldots,N;\ j = 1,\ldots,7) \qquad (3)$$

Sesuai dengan penjelasan formulasi sebelumnya, N jumlah dari tenaga kerja dan v_j, jumlah dari tenaga kerja yang mengambil j hari libur. Linearisasi klasik dari (**QIP**) diperoleh dengan mengganti h_{i-j} setara dengan f_{i-j}.. f_{i-j+1} pada optimum. Solusi dengan mengikuti *linear integer programming* (**IP5**) memberikan jadwal dari hari libur dengan maksimum hari libur yang berurutan.

$$\text{(IP5)}\quad \max \sum_{i=1..N}\sum_{j=1..7} t_{i,j}$$

$$\text{s.t}\quad \sum_{i=1..N} u_{i,j} + t_{i,j-1} + t_{i,j-1} = v_j \quad (i = 1,\ldots,7) \qquad (1)$$

$$\sum_{j=1..7} u_{i,j} + 2t_{i,j-1} = 2 \quad (i = 1,\ldots,N) \qquad (2)$$

$$u_{i,j},\ t_{i,j} \in \{0,1\} \quad (i = 1,\ldots,N;\ j = 1,\ldots,7) \qquad (3)$$

Penggunaan formulasi ini sangat sederhana dan umum untuk setiap jumlah hari libur. Walaupun begitu perhitungan waktu yang dibutuhkan untuk memperoleh jadwal optimal dari hari libur lebih besar dari yang dibutuhkan pada formula (IP3). Percobaan perhitungan telah diperlihatkan bahwa solusinya mungkin cukup baik yang diperoleh lebih cepat (dalam waktu sedikit) tapi bukti bahwa optimalisasi memerlukan sejumlah cabang yang sangat besar pada *branch-bound procedure*. Dari kondisi ini memperlihatkan bahwa masalah penjadwalan dan perhitungan waktu ini merupakan ilustrasi yang terbaik dan cukup penting dari formulasi *integer programming* (lihat Tabel 11).

Dengan melihat pada pengembangan model, maka metode tersebut cukup efisien untuk menyelesaikan masalah dan untuk ini diberikan satu syarat yang sangat penting yaitu $(b_i, i = 1,2,\ldots,m)$, dan i dinotasikan sebagai urutan periode waktu, ukuran *shift* x_i dimulai dari i dan berakhir sampai s periode waktu, seperti pada *total service* yang diperlukan untuk penyelesaian fungsi linear minimum *shift*. Sayangnya, kita tidak dapat menggunakan meode ini karena masalahnya sangat berbeda, hari libur yang diberikan pada tenaga kerja yang mempunyai *maksimum hari libur yang berurutan*, tetapi solusi yang ada tidak yakin apakah

hari libur untuk setiap tenaga kerja semuanya berurutan.

Tabel 11. Perbandingan dari Perhitungan Waktu dalam Detik dua Hari Libur untuk Tenaga Kerja dengan Penyelesaian (IP3) atau (IP5).

Jumlah Tenaga Kerja		2 Hari Libur (IP3)	2 Hari Libur (IP5)
N = 5	$v_{j=}$ (2,0,3,1,1,2,1)	0.4 s	55.95 s
N = 10	$v_{j=}$ (0,7,3,2,0,5,3)	3.22 s	-
N = 15	$v_{j=}$ (2,0,10,3,9,3,3)	15.72 s	-
N = 20	$v_{j=}$ (8,0,5,10,5,8,4)	17.30 s	-
N = 25	$v_{j=}$ (9,2,12,11,5,2,9)	19.35 s	-
N = 30	$v_{j=}$ (15,12,0,15,0,15,3)	28.47 s	-

Catatan: N adalah jumlah dari tenaga kerja dan **(-)** berarti solusi optimal yang tidak diperoleh dalam **10** menit dalam waktu CPU.

KESIMPULAN

Dari uraian dan pembahasan sebelumnya, dapatlah disimpulkan hal - hal sebagai berikut:

1. Pemograman *integer* (*Integer Programming*) merupakan teknik yang cukup efisien dalam pemecahan masalah penjadwalan tenaga kerja di industri tekstil, yang dalam penelitian ini dilakukan di pabrik pertenunan (*Weaving Mill*).
2. Solusi penjadwalan tenaga kerja dapat dikerjakan dengan cepat melalui penggunaan pemograman *integer* model *solver XA*, dan melalui sistem ini perubahan dan perencanaan penjadwalan dapat diantisipasi hasilnya secara efisien dan efektif.
3. Adanya penerapan pemograman *integer* ini menghasilkan penjadwalan tenaga kerja yang cukup efisien termasuk pengaturan hari libur yang berurut untuk setiap tipe tenaga kerja, yang dampaknya pemutusan hubungan kerja (PHK) di perusahaan dapat dihindarkan.
4. Hasil penerapan program menunjukkan bahwa makin banyak data yang diproses maka secara langsung memerlukan waktu yang makin banyak lagi, tapi biarpun demikian dengan aplikasi pemograman ini tetap merupakan suatu proses pemecahan masalah yang cukup efisien.

DAFTAR PUSTAKA

1. Barbara, A., R., "*Problem Solving and Work load*", Perganon Publishing, New York, 1997.
2. Anderson, B., D., "*A Qualitative Introduction to Basic Language Programming", The Institute of Radio & Electricity*", Australia, 1996.
3. Barnard, G., A., 1979, "*Control Charts and Stochastic Processes*", Journal of The Royal Statistical Society, Series 1321, hal. 239 - 271.
4. Gilbert, O., "*Solver XA-Program Methode*", Rocckled Publishing, Canada, 1986.
5. Chang, S., H., 1991, "*Estimation of Forecast Errors for Seasonal Style Goods Sales*", Management Science, 18, no. 2, hal. 1389 – 1396, 1991.
6. Duncan, D., B., et al., 1972, "*Linear Dynamic Recursive Estimation from The Viewpoint of Regression Analysis*", Journal of The America Statistical Association, 17, no. 340, hal. 815 - 821.
7. Gelb, G., "*Applied Optimal Estimation-Time and Motion Study*", 3rd, Canbridge, 1997.
8. Giovonno, R., A., "*Programming System and Applications*", Stuttgart University, Stuttgart, Deutschland, 2006.
9. Glass, G., V., et al., "*Design and Analysis of Time Series Experiments*", 3rd Edition, Colorado, Colorado University, 2005.
10. Gracia, Smith, "*Computing System for Work Load*", Barkey Co., Ltd., Muenchen, Deutschland, 1997.
11. Hung, R., et al., "*Programming System in Microcomputer*", Lions Co., Ltd., Peking, 1998.