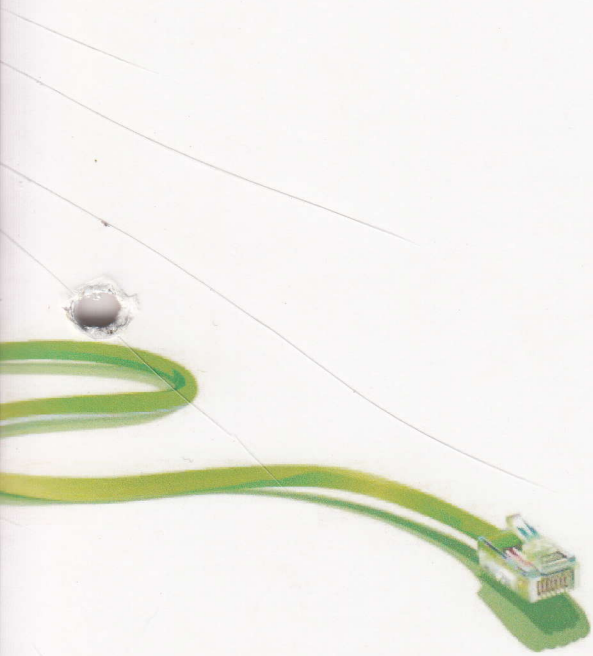
ISSN
9 772303 075009

LAMPIRAN B 5



Jurnal Teknik Informatika

Desember 2012

Vol. I #1

Rekursif

Rekursif

Jurnal
Teknik
Informatika

Volume I Nomor 1 Desember 2012

Alamat Redaksi

Jurnal Rekrusif Program Studi Teknik Informatika
Fakultas Teknik - Kampus Universitas Bengkulu
Jl. W.R Supratman Kandang
Limun Bengkulu 38371
Telp. (0736) 344087, 21170 – 227
Email : rekursifunib@gmail.com

www.ti.ft.unib.ac.id

DAFTAR ISI

ISSN
9 772303 075009

ISSN 9 772303 075009

RANCANG BANGUN APLIKASI METODE *TABU SEARCH* PADA PENYELESAIAN *ASSIGNMENT PROBLEM*

Boko Susilo[1], Rusdi Efendi[2], Ernawati[3], Safta Hastini[4]

Program Studi Teknik Informatika, Fakultas Teknik, Universitas Bengkulu
Jl. W.R Supratman Kandang Limun Bengkulu 38371 Telp. (0736) 344087, 21170 – 227

[1)] email: bksusilo@gmail.com
[2)] email: r_efendi@yahoo.com
[3)] email: w_ier_na@yahoo.com
[4)] email: safta_g1a006012@yahoo.com

ABSTRAK

juan dari penelitian ini adalah untuk membangun aplikasi metode Tabu Search pada penyelesaian signment Problem dan untuk mengetahui sejauh mana pengaruh perubahan parameter n, yaitu nyaknya sumber daya dan tugas terhadap waktu. Assignment problem merupakan masalah alokasi mber daya yang berkaitan dengan pemilihan sumber daya yang tepat dalam menyelesaikan tugas rtentu, sehingga hasil yang didapat akan optimal.Aplikasi ini dapat membantu para pelaku bisnis industri lam mengambil kebijakan dan tindakan dalam memilih sumber daya yang terbatas untuk menyelesaikan atu tugas. Pada penelitian ini, sistem dibangun dengan menggunakan metode Tabu Search dengan hasa pemrograman Delphi 7, metodepengembangan sistem waterfall, dan perancangan sistem DFD. sil pengujian menunjukkan bahwa ada pengaruh perubahan jumlah sumber daya, jumlah tugas dan maksimum iterasi terhadap waktu.

***Kata Kunci**: Assignment Problem, Metode Tabu Search, Delphi, Waterfall, DFD*

PENDAHULUAN

Perkembangan ilmu pengetahuan saat menuntut para pelaku industriuntuk pat mengambil keputusan menentukan bijaksanaan dan tindakan secara iah, bagaimana merancang dan njalankan sistem manusia-mesin secara timal walaupun menghadapi masalah ngalokasian sumber daya yang terbatas. salah alokasi ini muncul bilamana harus ilih tingkat pekerjaan tertentu yang saing dengan sumber daya manusia g terbatas untuk melaksanakan erjaan-pekerjaan tersebut sehingga il yang didapat akan optimal.

Pada umumnya tingkat keterampilan, galaman kerja, latar belakang pendidikan, dan latihan setiap pekerja berbeda-beda.Sehingga dalam waktu penyelesaian pekerjaan yang sama itu berbeda juga. Masalah manajemen adalah bagaimana menetapkan pemberian sejumlah tugas kepada pekerja atas dasar satu-satu dengan cara yang optimal (satu tugas dikerjakan oleh satu pekerja), dimana setiap pekerja mempunyai tingkat efisiensi yang berbeda-beda untuk tugas yang berbeda-beda pula dan pekerja mempunyai tingkat efisiensi yang sama daiam satu tugas namun berbeda untuk tugas yang lain.

"Riset Operasi (*Operations Research)* merupakan suatu teknik pemecahan masalah yang menerapkan metode-

metode ilmiah terhadap masalah-masalah rumit yang muncul dalam pengarahan dan pengelolaan dari suatu sistem yang terdiri dari manusia, mesin, bahan dan uang dalam industri, bisnis, pemerintahan dan pertahanan"[5].Pada Riset Operasiterdapat permasalahan dalam pemilihan personal yang tepat dalam mengerjakan suatu tugas tentu yang berpengaruh untuk suksesnya suatu usaha. Masalah ini disebut *assignment problem* ataumasalah penugasan yang merupakan suatu kasus khusus dari masalah *linear programming* tentang pembagian sumberdaya dan tugas, dimana sejumlah *m* sumber daya ditugaskan kepada sejumlah *n* tugas. Sumber daya adalah pekerjaan (pekerja/ agen) sedangkan yang dimaksud tugas adalah mesin-mesin. Jadi, *assignment problem*akan mencakup sejumlah *m* sumber daya yang mempunyai *n* tugas dengan C_{ij} biaya penugasan yang bertujuan menghemat biaya/waktu pada kasus minimasi atau mendapatkan keuntungan maksimum pada kasus maksimasi. Dalam halini, jumlah *m* sumber daya sama dengan jumlah *n* tugas.

Dari sudut pandang masalah optimasi, penyelesaian*assignmentproblem*bertujuan untuk memetakan *m*sumber daya terhadap *n* tugas berdasarkan penugasan satu-ke-satu, karena secara umum satu orang hanya bisa mengerjakan satu tugas dalam waktu bersamaan. Walaupun untuk menyelesaikan masalah ini bisa menggunakan metode umum yaitu dengan cara permutasi dari sejumlah *n* sumber daya dengan sejumlah *n* tugas, sehingga akan diperoleh *n!(n faktorial)* alternatif. Metodeini mudah dilakukan kalau *n* kecil, tetapi kalau sudah menyangkut untuk *n* yang besar cara ini kurang efisien, karena harus mencari alternatif dari *n!*buah kemungkinan yang harus dipilih. Sehingga besarnya sejumlah *n* yang harus dipetakan akan berdampak terhadap waktu yang digunakan untuk proses pengalokasian tugas tersebut.

Untuk melakukan suatu tugas, sumber daya harus dipilih berdasarkan *skill* yang dimiliki, yaitu dipilih sesuai dengan probabilitas sumber daya yang tinggi dimana banyaknya sumber daya yang harus dipilih untuk mengerjakan tugas tertentu berpengaruh terhadap waktu pengalokasian tugas tersebut. Karena itu, permasalahan ini merupakan permasalahan optimasi kombinatorial yang kompleks atau permasalahan *np-hard*, yaitu suatu permasalahan yang *pencarian solusinya (waktu komputasinya)* akan naik seiring dengan naiknya skala parameter *n* secara linear. Oleh karena itu, pada penelitian ini digunakan metode heuristik untuk pemecahan *assignment problem*. Metode heuristik adalah metode pencarian solusi permasalahan optimasi kombinatorial yang cukup sulit dan berskala besar dengan cara mencari *good solution* yang dapat mencapai semua kriteria dengan waktu yang relatif kecil[2].

Salah satu metode heuristik yang dapat digunakan adalah metode *Tabu Search*.

"Metode *Tabu Search* merupakan suatu metode optimasi yang menggunakan *short-term memory* untuk menjaga agar proses pencarian tidak terjebak pada nilai *optimum local*"[4]. Metode ini bertujuan untuk mengefektifkan proses pencarian solusi terbaik dari suatu permasalahan optimasi kombinatorial yang berskala besar.Untuk menjaga agar solusi terbaik tidak hilang, maka metode *Tabu Search* menyimpan solusi terbaik ke dalam *memory* yang disebut *tabu list,* serta terus mencari solusi terbaik dari setiap iterasi, kemudian dilakukan perbandingan antara solusi terbaik yang baru dievaluasi dengan solusi terbaik sebelumnya. Selain itu, metode ini mengingat atau melakukan pengecekan solusi yang pernah ditemui pada *memory* (*tabu list*) dan melarang untuk menggunakan solusi yang telah dievaluasi untuk menghindari pengulangan yang sia-sia. Hal ini yang membuat metode *Tabu Search* menjadi lebih efisien dalam hal waktu[8].

Penelitian ini bertujuan untuk menerapkan metode *Tabu Search*dalam menyelesaikan *assignmentproblem* yang bisa menangani sejumlah *n* dengan waktu yang relatif singkat.Oleh karena itu, untuk mencapai tujuan tersebut, maka dibangun aplikasi metode *Tabu Search*pada penyelesaian *assignment problem.*

2. LANDASAN TEORI

2.1 Assigment Problem

Assignment problem atau masalah penugasanpertama kali dikenalkan oleh seorang ahli matematika dari Hongaria yang bernama D Konig pada tahun 1916. "*Assignment problem* adalah jenis khusus *linear programming* dimana sumber-sumber dialokasilan kepada kegiatan-kegiatan atas dasar satu-satu *(one-to-one-basis)...*"[3].

Assignment problem merupakan masalah pengalokasian sejumlah *m* sumber daya yang ditugaskan kepada sejumlah *n* tugas (satu sumber daya untuk satu tugas) sehingga diperoleh biaya/waktu total yang minimum atau keuntungan total yang maksimum. Dalam hal ini, sumber daya adalah pekerjaan (pekerja/agen), sedangkan yang dimaksud dengan tugas adalah mesin-mesin. Jadi, ada *m* sumber daya yang ditugaskan kepada *n* tugas, dimana sumber daya *i (i = 1, 2, ... , m)* ditugaskan kepada tugas *j (j = 1, 2, ... , n)* dengan C_{ij} biaya penugasan.

Pada *assignment problem* satu sumber daya hanya ditugaskan kepada satu tugas, maka a_i kapasitas penawaran pada setiap sumber daya yang tersedia hanya 1. Demikian pula halnya dengan tugas, karena satu tugas hanya dikerjakan oleh satu sumber daya, maka *bj* kapasitas permintaan pada setiap tugas yang tersedia hanya 1.Secara matematis dapat dinyatakan sebagai berikut

$$X_{ij} = \begin{cases} 0, \text{sumber daya ke} - i, \text{tidak melakukan tugas ke - } j \\ 1, \text{sumber daya ke} - i, \text{melakukan tugas tugas ke - } j \end{cases}$$

dimana $i = 1, 2, ..., m,\ j = 1, 2, ..., n$

Berikut ini gambaran umum penyelesaian *assignment problem* dapat dilihat pada Tabel 1 di bawah ini:

Tabel 1. Gambaran Umum PenyelesaianAssignment Problem

Sumber daya	Tugas				
	1	2	...	*n*	a_i
1	C_{11} X_{11}	C_{12} X_{12}	...	C_{1n} X_{1n}	1
2	C_{21} X_{21}	C_{22} X_{22}	...	C_{2n} X_{2n}	1
...	...	...	...	...	...
m	C_{m1} X_{m1}	C_{m2} X_{m2}	...	C_{mn} X_{mn}	1
b_j	1	1	...	1	

Dengan model *assignment problem*sebagai berikut:

Fungsi tujuan (maksimum/minimum)

$$Z = \sum_{i=1}^{m}\sum_{j=1}^{n} C_{ij}X_{ij} \qquad (1)$$

Batasan-batasan

$$\sum_{i=1}^{m} x_{ij} = 1;\ j = 1,2,\ldots,m\ . \quad (2)$$

$$\sum_{i=1}^{n} x_{ij} = 1; j = 1,2,\ldots,n; \quad (3)$$

dan x_{ij}=0 atau 1 untuk *i=1, 2, ..., m* dan *j=1, 2,..., n.*

Keterangan dari model *assignment problem* di atas, yaitu:

1. Fungsi tujuan adalah *Z* sebagai total C_{ij}biaya penugasan yang menyatakan apakah *assignment problem* ini mencari nilai minimum atau maksimum.
2. Variabel keputusan adalah X_{ij}. Nilai dariX_{ij} = *0* atau 1 untuk *i = 1, 2, ..., m* dan *j = 1, 2, ..., n.*Bernilai 1, apabila *i*sumber daya ditugaskan untuk *j* tugas atau bernilai 0, apabila *i*sumber daya tidak ditugaskan untuk *j* tugas.
3. Tetapan biaya yang telah ditentukan adalahC_{ij} dari *m*sumber daya yang ditugaskan kepada *n*tugas.
4. Jumlah sumber daya adalah *m.*
5. Jumlah tugas yang akan diselesaikan adalah *n.*

Variabelkeputusan X_{ij} bernilai 0 atau 1. Dalam hal ini, untuk menyelesaikan *assignment problem* dicari letak X_{ij}yang bernilai 1 dimana variabel X_{ij} bernilai 1 apabila *m* sumber daya ditugaskan ke *n* tugas yang didapat berdasarkan keputusan saat X_{ij} terpilih menjadi langkah penugasan pada setiap pencarian solusi dengan C_{ij} biaya penugasan, sehingga angka tersebut dikalikan dengan biaya terpilih dari sel tersebut (X_{ij}* C_{ij}).Sedangkan untuk sel yang tidak terpilih maka nilai X_{ij} bernilai 0 karena *m* sumber daya tidak terpilih untuk melakukan *n* tugas. Sehingga fungsi tujuan *Z* dapat digambaran sebagai berikut:

$$\begin{aligned} Z = [(C_{11}.X_{11}) + (C_{12}.X_{12}) \\ + (C_{1n}.X_{1n})^2 \\ + [(C_{21}.X_{21}) + (C_{22}.X_{22}) \\ + \ldots + (C_{2n}.X_{2n})^2 \\ + [(C_{m1}.X_{m1}) + (C_{m2}.X_{m2}) \\ (C_{mn}.X_{mn})^2 \ \ldots \ (4) \end{aligned}$$

Berdasarkan Persamaan (4),

misalnya baris pertama terpilih penugasanmaka X_{11}bernilai 1 seda pertama atau kolo bernilai 0 karena a setiap sumber da permintaan setiap 1 karena satu sum tugas dan sebalik satu sumber da untuk baris kedu seterusnya. Sehing

$Z = [(0) + C \_ Z$
$+ [(0)$
$* X\_mn$

Berdasarkan dapat disimpulkan *problem* adalah m (maksimum/minim yang telah ditentu sumber daya yan kepada *n* tugas penawaran pada dan *bj* kapasitas setiap tugas be variabel keputus atau 1. Dalam ha terpilih dengan v yang bernilai 1 yait mengerjakan tugas kapasitas penawaran daya dan *bj* kapasi setiap tugas dan tidak perlu dicantu gambaran umum pe *problem* dapat dise tabel ongkos berikut

misalnya baris pertama kolom pertama terpilih dengan C_{11} biaya penugasanmaka variabel keputusan X_{11} bernilai 1 sedangkan X_{ij} pada baris pertama atau kolom pertama lainnya bernilai 0 karena a_i kapasitas penawaran setiap sumber daya dan *bj* kapasitas permintaan setiap tugas hanya bernilai 1 karena satu sumber daya untuk satu tugas dan sebaliknya satu tugas untuk satu sumber daya. Demikian juga untuk baris kedua kolom kedua dan seterusnya. Sehingga nilai *Z* menjadi:

$$Z = [(0) + C\_22 * X\_22) + \cdots + (0)] + [(0) + (0) + \cdots (C\_mn * X\_mn)] \ldots (5)$$

Berdasarkan penjelasan di atas, dapat disimpulkan bahwa *assignment problem* adalah mencari *Z* total biaya (maksimum/minimum) dari C_{ij} biaya yang telah ditentukan untuk setiap *m* sumber daya yang akan ditugaskan kepada *n* tugas dengan a_i kapasitas penawaran pada setiap sumber daya dan *bj* kapasitas permintaan pada setiap tugas bernilai 1 sehingga variabel keputusan X_{ij} bernilai 0 atau 1. Dalam hal ini, dicari C_{ij} yang terpilih dengan variabel keputusan X_{ij} yang bernilai 1 yaitu sumber daya yang mengerjakan tugas.Oleh karena itu, a_i kapasitas penawaran pada setiap sumber daya dan *bj* kapasitas permintaan pada setiap tugas dan variabel keputusan X_{ij} tidak perlu dicantumkan. Sehingga tabel gambaran umum penyelesaian *assignment problem* dapat disederhanakan menjadi tabel ongkos berikut ini:

Tabel 2. Tabel Ongkos pada *Assignment Problem*

	1	2	...	n
1	C_{11}	C_{12}	..	C_{1n}
2	C_{21}	C_{22}	..	C_{2n}
.	.	.	..	.
.	.	.	..	.
.	.	.	..	.
M	C_{m1}	C_{m2}	..	C_{mn}

2.2 Metode *Tabu Search*

"Metode *Tabu Search* adalah suatu metode optimasi matematis yang termasuk ke dalam kelas *local search*. Metode *Tabu Search* memperbaiki performansi *local search* dengan memanfaatkan struktur *memory*[8]. Definisi lainnya menyatakan bahwa metode *Tabu Search* adalah suatu metode optimasi yang menggunakan *short-term memory* untuk menjaga agar proses pencarian tidak terjebak pada nilai *optimum local*"[4].

Konsep dasar penyelesaian metode *Tabu Search* adalah pengefektifan proses pencarian solusi dengan cara mencari *best solution* pada setiap tahap pelacakan. Padabeberapa tahap pelacakan dapat dikategorikan sebagai langkah *tabu* (dilarang) karena akan menghasilkan *localoptimal* dan juga karena akan mengakibatkan langkah pengulangan kembali pencarian ke solusi yang pernah ditemukan sebelumnya *(entrapment).*

Langkah-langkahini kemudian dimasukkan ke dalam daftar yang disebutdengan *tabu list*. *Tabu list* merupakant empat untuk menyimpan sekumpulan solusi yang baru saja dievaluasi. Proses pencariannya dilakukan dengan cara menentukan solusi awal dankemudian dilakukan gerakan *(move)* ke solusi-solusiberikutnya dan baru berhenti sampai kriteria penghentian *(stopping conditions)* tercapai. Kriteria penghentian metode *Tabu Search*, misalnya sejumlah iterasi yang ditentukan oleh *user*, sejumlah waktu CPU tertentu, atau sejumlah iterasi berurutan tanpa peningkatan nilai fungsi objektif terbaik, dan sebagainya [8].

Metode *Tabu Search* bekerja secara iteratif menggunakan algoritma *local search* pada setiap iterasi untuk mencari solusi diantara sebagian tetangga dari solusi terbaik saat ini. Pada setiap iterasi, algoritma *local search* memilih solusi tetangga dengan total biaya/waktu yang minimal pada kasus minimasi atau total keuntungan maksimum pada kasus maksimasi tergantung dari kasus yang dihadapi.

Untuk efesiensi *memory* dan waktu proses, *tabu list* hanya menyimpan *move* yang merupakan kebalikan dari langkah yang telah digunakan pada iterasi-iterasi sebelumnya dengan panjang *list* yang dibatasi oleh *user*. Metode *Tabu Search* mempunyai struktur *memory* yang menyimpan solusi terbaik ke dalam *tabu list*. Selain itu, metode ini mengingat solusi yang ada dengan cara melakukan pengecekan pada *tabu list* dan melarang menggunakan solusi yang pernah ditemui untuk menghindari pengulangan solusi yang sudah ada. Oleh karena itu, dengan menggunakan *tabu list*, metode *Tabu Search* dapat menerima solusi yang tidak memberikan peningkatan kualitas, sehingga metode *Tabu Search* bisakeluardari *optimum local*. Akan tetapi, terdapat pengecualian dalam metode *Tabu Search*.Jika terdapat *move* yang sudah berada dalam *tabu list* (terlarang untuk dipilih) tetapi memberikan solusi yang lebih baik dibandingkan semua solusi terbaik yang pernah dievaluasi, maka *move* tersebut bisa diterima dan *move* tersebut harus dikeluarkan dari *tabu list* (dibebaskan dari larangan). Hal ini merupakan prioritas khusus pada *tabu list* yang disebut kriteria aspirasi atau kondisi aspirasi *(aspiration conditions)*.Algoritma *Tabu Search* secara garis besar dapat ditulis sebagai berikut[4]:

```
Langkah 0. Tetapkan:
     X=Matriks input berukuran
     mxn.
      MaxIter=maksimum iterasi.
Langkah 1. S=bangkitkan solusi
secara random
Langkah 2. GlobalMin=FCost(S)
Langkah 3. Best=S
Langkah 4. TabuList=[ ]
Langkah 5. Kerjakan dari k=1
          sampai MaxIter:
    Langkah 6.
BestSoFar=FCost(S)
Langkah 7.  BestMove=S
Langkah 8.  Kerjakan dari i=1
sampai(n-1):
Langkah 9.   Kerjakan dari j=i
sampai n:
Langkah 10. L=Tukar(S[i],S[j]).
Langkah 11.    Cost=FCost(L)
Langkah 12. Jika(L∈TabuList)atau
            st<GlobalMin),
```

```
      kerjakan:
Langkah 13. Cost<BestSoFar),
                  kerjakan
Langkah 14. BestSoFar=Cost.
Langkah 15. BestMove=L.
Langkah 16. S=BestMove.
Langkah 17. Tambahkan S ke
TabuList
Langkah 18. Jika
            tSoFar<GlobalMin,
            kerjakan:
Langkah 19. GlobalMin=BestSoFar.
Langkah 20.    Best=BestMove.

Solusi akhir adalah Best, dengan
cost sebesar GlobalMin.
```

2.3 PengukuranWaktu Kerja Sistem

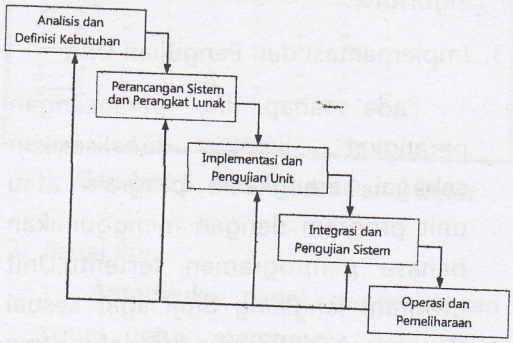
Pengukuran kerja sistem dilakukan untuk mendapatkan ukuran-ukuran tentang kinerja dan beban yang berlaku pada suatu sistem kerja. Kinerja adalah kecepatan atau pengukuran waktu kerja sistem yang akan dibuat (ukuran kuantitas untuk kualitas tertentu), sedangkan beban adalah yang dialami sumber daya (beban fisik dan beban psikososiologik)[7].

Pada penelitian ini, pengukuran waktu kerja sistem merupakan pengukuranwaktu sejauh mana terdapat pengaruhparameter *n*, yaitu banyaknya sumber daya dan tugas, terhadap waktu pencarian yang digunakan untuk proses pengalokasian tugas kepada sumber daya dengan tujuan untuk mengetahui kinerja dari sistem yang telah dibangun. Selain itu, akan dilakukan analisis beban kerja. Dalam penelitian ini, beban kerja adalah banyaknya sumber daya yang akan dialokasikan kepada tugas untuk mendapatkan solusi optimal. Waktu akan dicatat dengan menggunakan *clock* fisik yang terdapat pada komputer sebagai alat pengukur waktu dalam penyelesaian suatu aktivitas yang diamati.

Clock adalah alat elektronik yang menghitung osilasi yang terjadi pada frekuensi tertentu, dan menyimpannya dalam *counter register*. Sistem operasi membaca *clock* fisik tersebut dan menerjemahkannya ke *software clock*. *Software clock* tidak selalu akurat sehingga perhitungan waktu *hardware* dan *software* memiliki perbedaan walaupun sangat kecil. Namun, *software clock* tetap menjadi acuan pencatatan waktu setiap kejadian proses[1].

2.4 Metode Pengembangan Sistem

Metodepengembangan sistem menggunakan model SDLC (*System Development Life Cycle*) atau Daur Hidup Pengembangan Sistem dengan salah satu modelnya yaitumodel *sequensial linier* atau model *waterfall*. Model ini mengambil kegiatan proses dasar seperti spesifikasi, pengembangan, validasi, evolusi, dan merepresentasikannya sebagai tahap-tahap proses yang berbeda seperti definisi kebutuhan, perancangan perangkat lunak, implementasi, pengujian, pengoperasian dan pemeliharaan. Model *waterfall* untuk pengembangan sistem ini dapat dilihat pada gambar di bawah ini [6]:



Gambar 1. Model *Waterfall*

Tahapan yang digunakan dalam *waterfall* adalah sebagai berikut:

1. Analisis dan Definisi Kebutuhan

   Mengumpulkan data yang dibutuhkan oleh sistem secara lengkap dengan teknik pengumpulan data menggunakan teknik studi pustaka yang bersumber dari literatur-literatur berupa buku-buku, laporan penelitian, karangan-karangan ilmiah, dan sebagainya. Kemudian menganalisis dan mendefinisikan kebutuhan tersebut sehingga sistem yang akan dibangun dapat memenuhi semua kebutuhan. Dengan hasil analisis yang digambarkan secara terstruktur.

2. Perancangan Sistem dan Perangkat Lunak

   Tahap ini menterjemahkan analisis kebutuhan ke dalam bentuk rancangan sebelum dilakukan penulisan program. Kegiatan ini berfokus pada rancangan struktur data, arsitektur perangkat lunak, representasi *interface*, dan prosedur algoritma.

3. Implementasi dan Pengujian Unit

   Pada tahap ini, perancangan perangkat lunak direalisasikan sebagai serangkaian program atau unit program dengan menggunakan bahasa pemrograman tertentu.Unit program langsung diuji agar sesuai dengan spesifikasinya atau algoritma yang telah ditentukan.

4. Integrasi dan Pengujian Sistem

   Unit-unit program diintegrasikan dan diuji secara keseluruhan (*system testing*) sebagai sistem yang lengkap untuk menjamin bahwa kebutuhan sistem telah dipenuhi.Memeriksa apakah perangkat lunak sudah sesuai dengan yang diharapkan atau belum.

5. Operasi dan Pemeliharaan

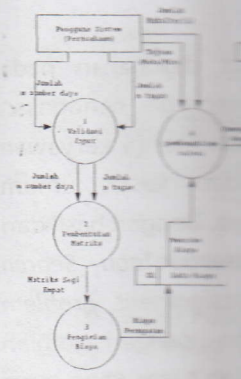
   Tahap ini merupakan fase siklus hidup yang paling lama.Sistem diinstal dan dipakai.Pemeliharaan mencakup koreksi dari berbagai *error* yang tidak ditemukan pada tahap-tahap terdahulu, perbaikan atas implementasi unit sistem dan pengembangan pelayanan sistem, sementara kebutuhan-kebutuhan baru ditambahkan.

Sistem ini akan dikembangkan dengan metode pengembangan sistem dengan analisis dan desain menggunakan pendekatan terstruktur. Adapun alat pengembangan sistem yang akan digunakan adalah *Data Flow Diagram*(DFD). DFD merupakan diagram yang menggunakan notasi-notasi (simbol-simbol) untuk menggambarkan arus data. DFD, sering digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika.

3. PERANCANGAN SISTEM

3.1 Perancangan *Data Flow Diagram* (DFD)

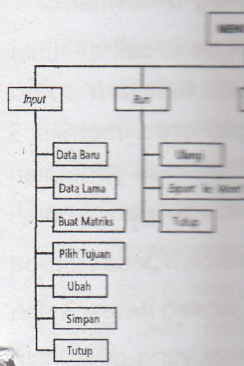
Perancangan sistem secara umum dilihat pada diagram level 0 dibawah ini:



Gambar

3.2 Perancangan A

Perancangan ar
menggunakan satu
panel utama, yaitu
Gambar 3 dapat d
yang terdapat pada

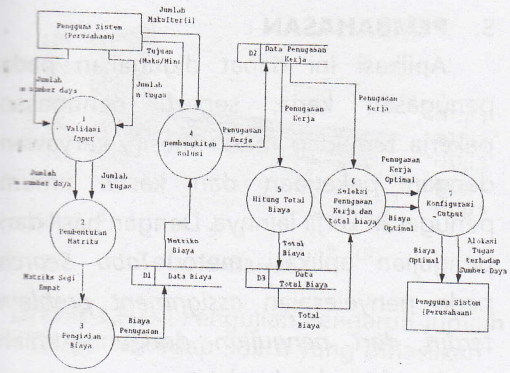


Gambar 3. Ranca

4. IMPLEMENTAS

4.1 Antarmuka

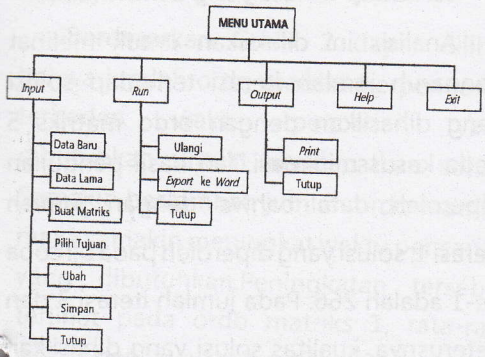
Antarmuka menu u
menampilkan pane
memilih tombol me
gambar antarmuka



Gambar 2. Diagram Level 0

3.2 Perancangan Antarmuka

Perancangan antarmuka pada aplikasi menggunakan satu form dengan satu panel utama, yakni menu utama. Dari Gambar 3 dapat dilihat beberapa panel yang terdapat pada panel menu utama.

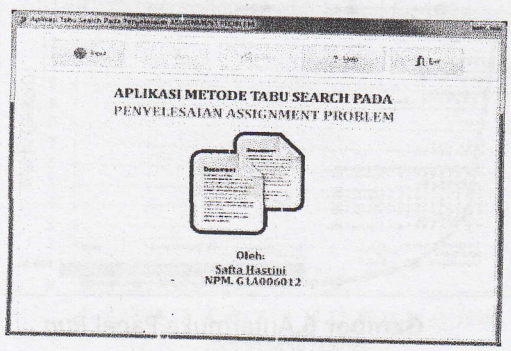


Gambar 3. Rancangan Struktur Menu

4. IMPLEMENTASI ANTARMUKA

4.1 Antarmuka Menu Utama

Antarmuka menu utama berfungsi untuk menampilkan panel data dengan cara memilih tombol menu. Berikut ini adalah gambar antarmuka menu utama pada sistem yang telah dibangun:



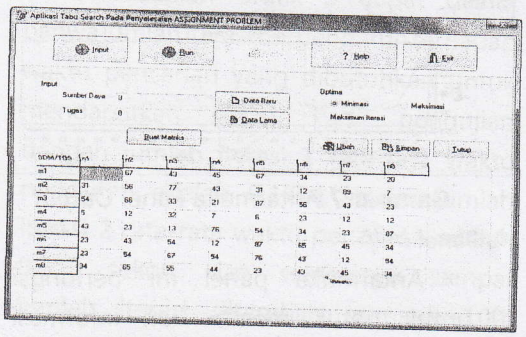
Gambar 4. Antarmuka Menu Utama

4.2 AntarmukaPanel Data

Antarmuka panel data berisi form yang berkaitan dengan menu yang tersedia. Adapun tampilan dari masing-masing panel adalah sebagai berikut:

1. Panel *Input*

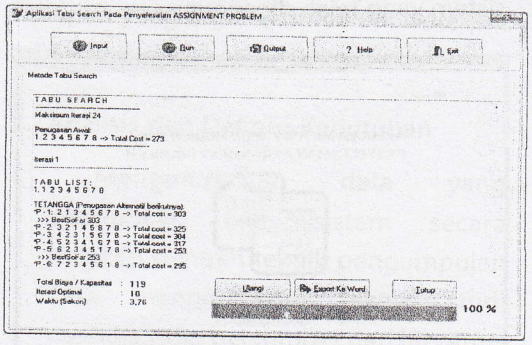
   Antarmuka panel *input* merupakan tampilan yang berfungsi untuk menampilkan form isian bagi pengguna sistem.Tampilannya dapat dilihat pada Gambar 5.



Gambar 5. Antarmuka Panel *Input*

2. Panel *Run*

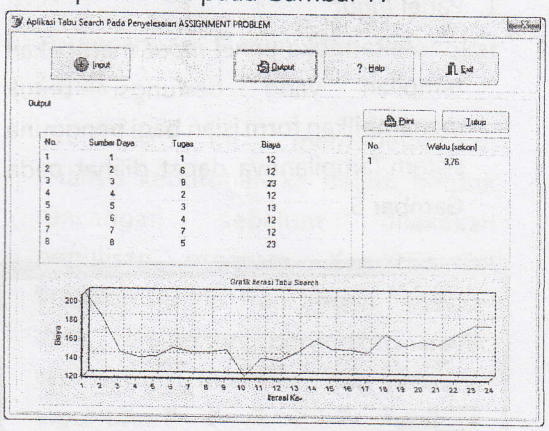
   Antarmuka panel *run*merupakan panel untuk memproses data pada tabel matriks. Proses dilakukan menggunakan algoritma *Tabu Search.*



Gambar 6. Antarmuka Panel *Run*

3. Panel *Output*

Antarmuka panel*output* berfungsi untuk menampilkan hasil proses perhitungan algoritma *Tabu Search*yang telah dilakukan sebelumnya pada panel *run*. Adapun tampilan panel *output* dapat dilihat pada Gambar 7.



Gambar 7. Antarmuka Panel *Output*

4. Panel *help*

Antarmuka panel ini berfungsi untuk untuk memberikan informasi mengenai aplikasi dan petunjuk pemakaian dari aplikasi yang telah dibangun.

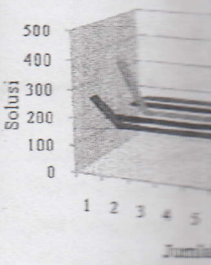
5. Menu *exit*berupa pesan yang berfungsi untuk menutup aplikasi secara keseluruhan.

5. PEMBAHASAN

Aplikasi ini dapat digunakan pada penugasan kerja, seperti penugasan pekerja terhadap mesin-mesin, karyawan dengan pekerjaan dan kasus umum penugasan kerja lainnya. Dengan hasil dari pengujian aplikasi metode*Tabu Search* pada penyelesaian *assignment problem terdiri dari pengujian* dengan jumlah sumber daya dan jumlah tugas yang terus bertambah untuk mengetahui solusi yang dihasilkandan pengujian jumlah ordo matriks, jumlah iterasi, jumlah sumber daya serta jumlah tugas untuk mengetahui waktu pencarian yang dibutuhkan sampai proses berhenti. Pengujian yang telah dilakukan sebanyak 5 kali.

5.1 Analisis Pengaruh Jumlah Iterasi terhadap Solusi yang Dihasilkan

Analisis ini dilakukan untuk melihat pengaruh jumlah iterasi terhadap solusi yang dihasilkan dengan ordo matriks 5 pada kasus minimasi.Dari hasil pengujian diperoleh data bahwa dengan jumlah iterasi 1, solusi yang diperoleh pada uji coba ke-1 adalah 266. Pada jumlah iterasi 3 dan seterusnya, kualitas solusi yang dihasilkan semakin baik, yaitu 169. Sehingga dapat disimpulkan bahwa bertambahnya jumlah iterasi mampu memperbaiki kualitas solusi yang dihasilkan.Hal ini disebabkan *Tabu Search* memiliki kesempatan yang lebih banyak untuk melakukan pembangkitan solusi.Hasil pengujian dapat dilihat padaGrafik1.

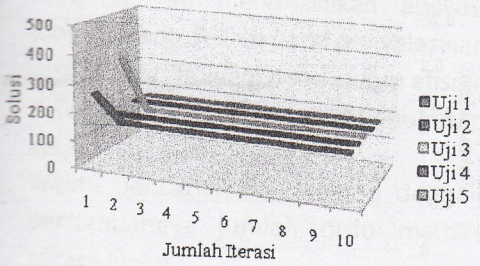


Grafik 1. Hasil Pen
Iterasi terhadap

5.2 Analisis
Ordo Matri
Pencarian

Analisis ini di
pengaruh jumlah
sampai 100 terh
yang dibutuhkan
solusi.Pengujian i
iterasi.

Berdasarkan
bahwa jumlah ord
terhadap waktu
dibutuhkan untuk
Semakin bertamba
maka semakin me
yang dibutuhkan
terlihat pada ord
waktu pencarian y
0,038 sekon. Ord
waktu pencarian y
0,0468 sekon dan
matriks ditambah
100 dengan wa
dibutuhkan meni
Waktu pencarian t
linear karena wa
untuk melakukan
dalam satu iterasi

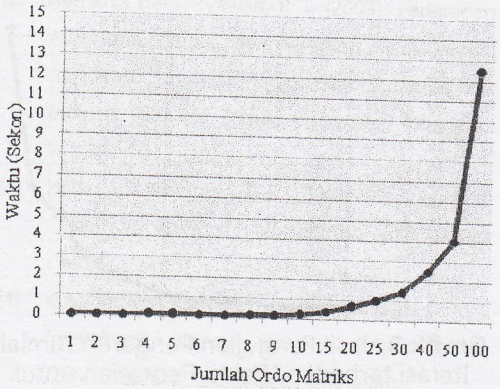


Grafik 1. Hasil Pengujian Pengaruh Jumlah Iterasi terhadap Solusi yang Dihasilkan

5.2 Analisis Pengaruh Jumlah Ordo Matriks terhadap Waktu Pencarian

Analisis ini dilakukan untuk melihat pengaruh jumlah ordo matriks dari 1 sampai 100 terhadap waktu pencarian yang dibutuhkan untuk mendapatkan solusi.Pengujian ini dilakukan dengan 1 iterasi.

Berdasarkan Grafik 2 dapat dilihat bahwa jumlah ordo matriks berpengaruh terhadap waktu pencarian yang dibutuhkan untuk mendapatkan solusi. Semakin bertambah jumlah ordo matriks, maka semakin meningkat waktu pencarian yang dibutuhkan.Peningkatan tersebut terlihat pada ordo matriks 1, rata-rata waktu pencarian yang dibutuhkan adalah 0,038 sekon. Ordo matriks 2, rata-rata waktu pencarian yang dibutuhkan adalah 0,0468 sekon, dan sampai jumlah ordo matriks ditambah secara linear menjadi 100 dengan waktu pencarian yang dibutuhkan meningkat 12,5054 sekon. Waktu pencarian terus meningkat secara linear karena waktu yang diperlukan untuk melakukan pembangkitan solusi dalam satu iterasi menjadi bertambah.

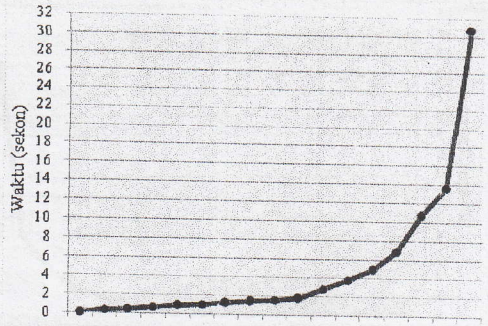


Grafik 2. Hasil Pengujian Pengaruh Jumlah Ordo Matriks terhadap Waktu Pencarian untuk 1 Iterasi

5.3 Analisis pengaruh jumlah iterasi terhadap waktu pencarian

Analisis ini dilakukan dengan melihat pengaruh jumlah iterasi terhadap waktu pencarian yang dibutuhkan sampai mendapatkan solusi.Dalam pengujian ini data yang digunakan adalah ordo matriks 10.

Berdasarkan Grafik 3 dapat dilihat bahwa jumlah iterasi berpengaruh terhadap waktu pencarian yang dibutuhkan untuk mendapatkan solusi. Pada pengujian dengan jumlah iterasi 1 rata-rata waktu pencarian adalah 0,1774 sekon, jumlah iterasi 2 rata-rata waktu pencarian adalah 0,359 sekon, dan seterusnya sampai jumlah iterasi ditambah menjadi 100, rata-rata waktu pencarian adalah 30,7602 sekon. Sehingga dari data tersebut, dapat disimpulkan bahwa bertambahnya jumlah iterasi akanmeningkatkan waktu pencarian yang dibutuhkan untuk mendapatkan solusi. Hal ini dikarenakan, jumlah iterasi yang dikerjakan semakin banyak.

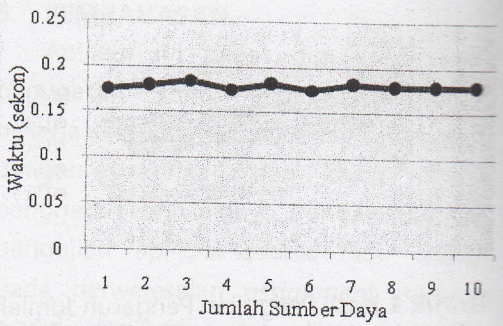


Grafik 3. Hasil Pengujian Pengaruh Jumlah Iterasi terhadap Waktu Pencarian untuk Ordo Matriks 10

5.4 Analisis pengaruh jumlah sumber daya dan tugas terhadap waktu pencarian

Analisis ini dilakukan untuk mengetahui pengaruh sumber daya dan tugas terhadap waktu pencarian untuk mendapatkan solusi.Untuk mewakili analisis ini, maka dapat dilihat pada Grafik 4, yaitu analisis pengaruh *dummy* pada sumber daya terhadap waktu pencarian. Dengan jumlah sumber daya bertambah dari 1 sampai 10 dan jumlah tugas tetap, yakni 10 tugas.

Dari Grafik tersebut dapat dilihat bahwa jika jumlah sumber daya bertambah dan tugas tetap, maka akan mempengaruhi waktu pencarian yang dibutuhkan untuk mendapatkan solusi optimal. Semakin bertambah jumlah sumber daya, maka waktu pencarian yang dibutuhkan hampir sama dan tidak stabil. Hal ini terlihat pada jumlah sumber daya 1 sampai jumlah sumber daya 3, jumlah sumber daya 4 sampai 5, jumlah sumber daya 6 sampai 7, dan jumlah sumber daya 7 sampai 10 terjadi peningkatan dan penurunan waktu pencarian solusi (tidak stabil).



Grafik 4. *Hasil Pengujian Jika Jumlah Sumber Daya Bertambah dan Jumlah Tugas Tetap*

Dari seluruh pengujian di atas ditarik kesimpulan bahwa pada analisis solusi diketahui jumlah iterasi mempengaruhi solusi yang dihasilkan. Dengan bertambahnya jumlah iterasi akan memperbaiki kualitas solusi dari *algoritma Tabu Search*. Sedangkan pada analisis kerja sistem disimpulkan bahwa jumlah ordo matriks, jumlah iterasi, jumlah sumber daya, dan jumlah tugas mempengaruhi waktu pencarian untuk mendapatkan solusi optimal.Semakin bertambah jumlah ordo matriks dan jumlah iterasi, maka waktu pencarian yang dibutuhkan akan meningkat. Begitu juga dengan jumlah sumber daya dan jumlah tugas, semakin bertambah jumlah sumber daya atau jumlah tugas, maka waktu pencarian yang dibutuhkan untuk mendapatkan solusi optimal hampir sama atau tidak stabil.

6. KESIMPULAN

Berdasarkan analisis dari aplikasi metode *Tabu Search* pada penyelesaian *assignment problem* yang telah dibangun, maka terdapat beberapa hal yang dapat disimpulkan, yaitu:

1. Penelitian ini menghasilkan aplikasi metode *Tabu Search* pada penyelesaian *assignment problem* yang cukup efisien dalam menyelesaikan kasus penugasan kerja.
2. Waktu pencarian naik seiring dengan bertambahnya jumlah ordo matriks secara linear.
3. Penyelesaian ordo matriks 1000 atau lebih dengan perangkat uji (komputer) yang digunakan pada penelitian ini belum bisa menyelesaikan *assignment problem*.
4. Kualitas solusi *algoritma Tabu Search*dipengaruhi jumlah iterasi.

PUSTAKA

[1] Ardani,I, Hamdani,T, dan W Herusetyo,A. (2011). *Time and Global State*.http://te.ugm.ac.id/~risanuri/distributed/ringk/Bab10.pdf. Terakhir diakses: 6 Maret 2011

[2] Betrianis dan Aryawan, P. T. (2003). Penerapan Algoritma *Tabu Search*dalam Penjadwalan *Job Shop. Jurnal Teknologi Departemen Teknik Industri, Fakultas Teknik, Universitas Indonesia.* 7(3), 107-112

[3] Hillier, F., Lieberman, G.J. (1990). *Introduction to Operations Research, fifth edition.* (alih bahasa Ellen Gunawan) Jakarta:Erlangga

[4] Kusumadewi, Sri dan Purnomo, Hari. (2005). *Penyelesaian Masalah Optimasi dengan Teknik-Teknik Heuristik.*Yogyakarta: Graha Ilmu

[5] Mulyono, Sri. (2004). *Riset Operasi. Edisi Revisi.* Jakarta: FEUI

[6] Sommerville, Ian. (2003). *Software Engineering (Rekayasa Perangkat Lunak) Edisi 6* (terjemahan). Jilid 1. Jakarta: Erlangga

[7] Sutalaksana, I.Z. (2010). *Pengukuran Sistem Kerja 1.* Lab Perancangan Sistem Kerja & Ergonomi. Departemen Teknik Industri ITB: Tidak Diterbitkan

[8] Suyanto. (2010). *Algoritma Optimasi Deterministik atau Probabilitik.* Yogyakarta: Graha Ilmu