BAB IV

IMPLEMENTASI DAN EVALUASI

4.1 Implementasi

Dalam tahap ini dibahas mengenai kebutuhan peragkat keras, implementasi *Update trail*, implementasi *DrawEdge,* penjelasan pemakaian program dan evaluasi dari aplikasi yang telah dibuat.

4.1.1 Kebutuhan Perangkat Keras

Spesifikasi Minimum

Untuk menjalankan aplikasi tugas akhir ini, spesifikasi perangkat keras minimum yang digunakan adalah komputer Pentium I-33 Mhz, memory 65 MB, VGA 8 MB, Hard disk 6 GB.

Spesifikasi yang disarankan

Untuk menerapkan Optimasi Masalah Transportasi menggunakan Ant Colony Algorithm ini agar berjalan baik, penulis menyarankan menggunakan komputer Pentium II keatas, memory 65 MB keatas, VGA 8 MB, Hard disk 40 GB. Hal ini dikarenakan agar dalam proses pencarian dapat lebih cepat.

4.1.2 Implementasi Update Trail

Mengacu ke rancangan pada bab 3.5, pada proses ini menggambarkan pengubahan jejak (*update trail*) tiap melakukan perjalanan.

```
// procedure to update pheromone trail each looping
procedure TFAntColonySystem.updateTrail();
var _from, _to, _ant : integer;
  I, num, ii, ij: Integer;
begin
try
  //  Pheromone evaporation
  for _from := 0 to MAX_CITIES -1  do    // Iterate
  begin
```

```
    for _to := 0 to MAX_CITIES -1   do    // Iterate
    begin
      TPheromone[_from][_to] := TPheromone[_from][_to] * (1.0 - RHO);
      if ( TPheromone[_from][_to] <= 0.0 ) then TPheromone[_from][_to] :=
INIT_PHEROMONE;
    end;    // for
  end;    // for
```

Menambahkan jejak feromon baru, dengan melihat perjalanan pada tiap semut

```
  for _ant := 0 to MAX_ANTS - 1 do    // Iterate
  begin

//update each leg of the tour given the tour length
    for I := 1 to MAX_CITIES  do    // Iterate
    begin

      if ( i <= MAX_CITIES) then
      begin
        _from := TAnts[_ant].path[i];
        _to := TAnts[_ant].path[i+1];
      end else
      begin
        _from := TAnts[_ant].path[i];
        _to := TAnts[_ant].path[0];
      end;

      if _ant >= MAX_ANTS  then MessageDlg('Kakean Semut sing metu',
mtError, [mbOK], 0);
      if _to < 0  then begin
      //MessageDlg('Nang njobo vektor', mtError, [mbOK], 0);
      _to:= MAX_CITIES -1 ; end;

      if ((_from > MAX_CITIES-1) or  (_from < 0 )) then _From :=
random(MAX_CITIES-1);
      if ((_to > MAX_CITIES-1) or  (_to < 0 )) then _to :=
random(MAX_CITIES-1);
```

Apabila diluar jangkauan, kembalikan lagi

```
//      ShowMessage('Pheromone From  := '+intToStr(_from)+' to
'+intToStr(_to));
(*      if (_from >= MAX_CITIES) or (_to >= MAX_CITIES)   then
        ShowMessage('Pheromone From  := '+intToStr(_from)+' to
'+intToStr(_to))
      else begin
*)
//      ShowMessage('Pheromone := '+FloatToStr(TPheromone[_from][_to]) +'
'+FloatToStr(TPheromone[_to][_from])) ;

      TPheromone[_from][_to] := TPheromone[_from][_to] + (QVAL /
TAnts[_ant].tourLength);
      TPheromone[_to][_from] := TPheromone[_from][_to];
floattostr(TPheromone[_from][_to]));
      // update to real pheromone 2D
  num := 1;


      {if _from <> _to then begin
      TPheromone[_to][_from] := TPheromone[_from][_to];
      end;
```

```
      {end;}
    end;    // for
  end;    // for
except
  //ExitThread(0);
  MessageDlg('ERROR on updatetrail Pheromone From  := '+intToStr(_from)+'
to '+intToStr(_to), mtError, [mbOK], 0);
end;
  for _from := 0 to MAX_CITIES -1  do    // Iterate
  begin
    for _to := 0 to MAX_CITIES -1   do    // Iterate
    begin
      TPheromone[_from][_to] := TPheromone[_from][_to] * RHO;
      TPheromone[_to][_from] := TPheromone[_from][_to];
    end;    // for
  end;    // for
```

4.1.3 Implementasi DrawEdges / ReturnPath / jalurnya

Mengacu ke rancangan pada bab 3.5, pada proses ini menggambarkan penempatan jejak (*trail*) yang pada setiap kota yang telah dilalui.

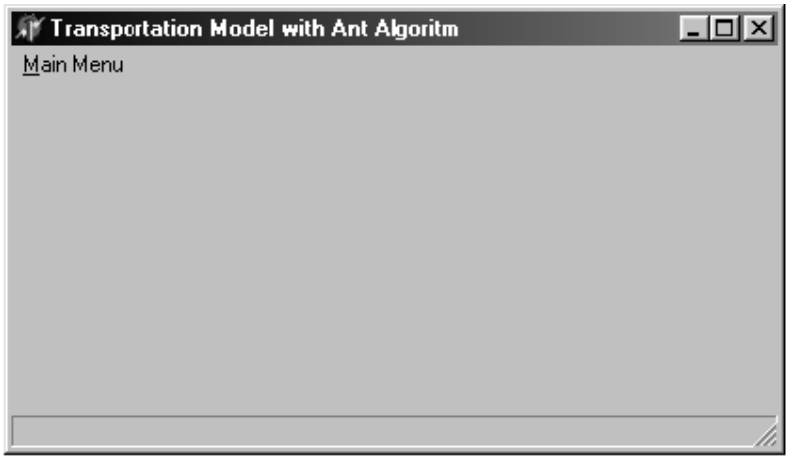
```
// Procedure to draw egdes each city
procedure TFAntColonySystem.drawEdges();
var _colStep, i, _from, _to, _pher : integer;
    _phLow, _phHi, _phTemp, _phStep  : double;
begin
  _colStep := 255;
  _phLow := 1110;
  _phHi := 0;
```

Iterasi feromon selesai dan mencari feromon yang tertinggi dan terendah.

```
  for _from := 0 to MAX_CITIES - 1  do    // Iterate
  begin
    for _to := 0 to MAX_CITIES - 1  do    // Iterate
    begin
      if (_from <> _to) then
      begin
        _phTemp := TPheromone[_from][_to];
        if _phTemp > _phHi then _phHi := _phTemp;
        if _phTemp < _phLow then _phLow := _phTemp;
      end;
    end;
  end;    // for
```

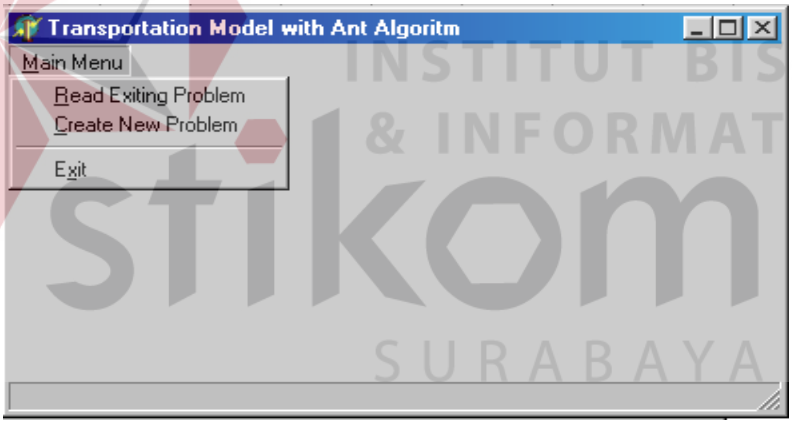
4.1.4. Penjelasan Pemakaian Program

Dalam bagian ini akan dijelaskan cara pemakaian program aplikasi yang sebelumnya telah diinstalasi ke dalam media penyimpanan. Penjelasan pemakaian program dimulai dari cara pemanggilan program hingga tampilan hasil proses penyelesaian masalah transportasi dengan menggunakan *Ant colony algorithm.*



Gambar 4.1 Tampilan awal program

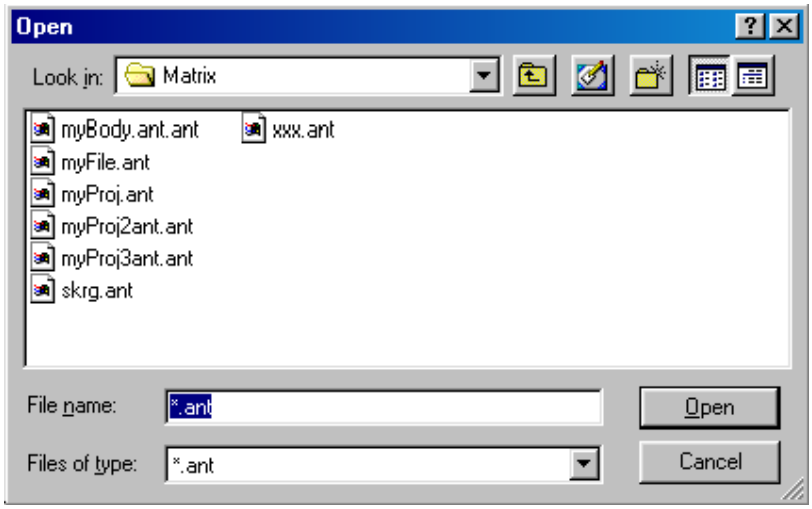
Setelah pemanggilan program aplikasi, akan muncul tampilan awal program seperti pada Gambar 4.1. Selanjutnya dapat dilakukan pemilihan pada menu utama (Main Menu): Read Exiting Problem, Create New Problem dan Exit.



Gambar 4.2. Menu utama

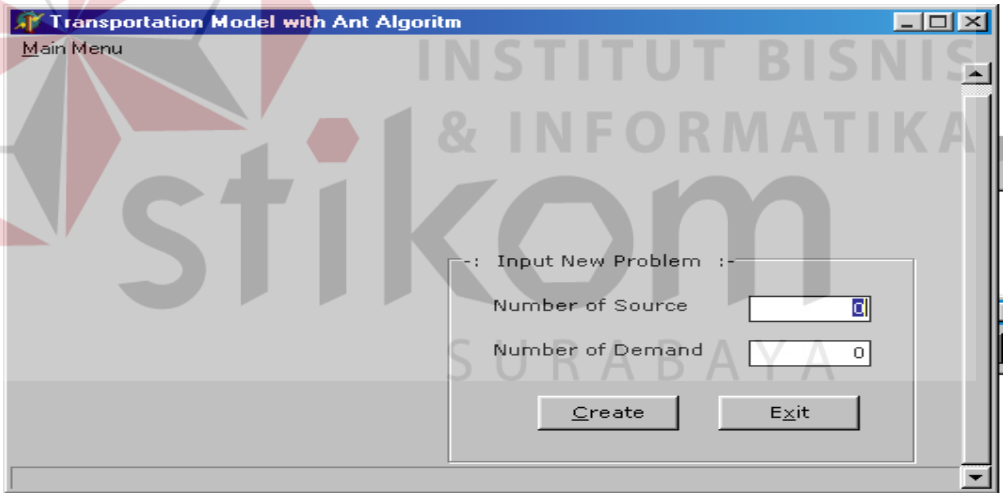
Pada Gambar 4.2. menu utama masing-masing dapat dijelaskan sebagai berikut:

1. Read Exiting Problem, pilihan ini digunakan untuk menampilkan form Open file, dimana pada form ini user dapat memanggil data masukkan masalah transportasi dari sebuah file yang telah disimpan sebelumnya (file *.ant) seperti pada Gambar 4.3.



Gambar 4.3. Form Open File

2. Create New Problem, pilihan ini digunakan untuk masukkan masalah transportasi baru. Apabila dipilih akan muncul form masukkan (Gambar 4.4.).



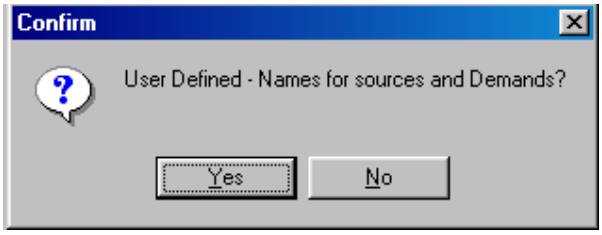
Gambar 4.4. Input New Problem

2.a. Number of Destination, digunakan untuk menginputkan jumlah dari tujuan (destination).

2.b. Number of Demand, digunakan untuk menginputkan jumlah dari sumber (source).

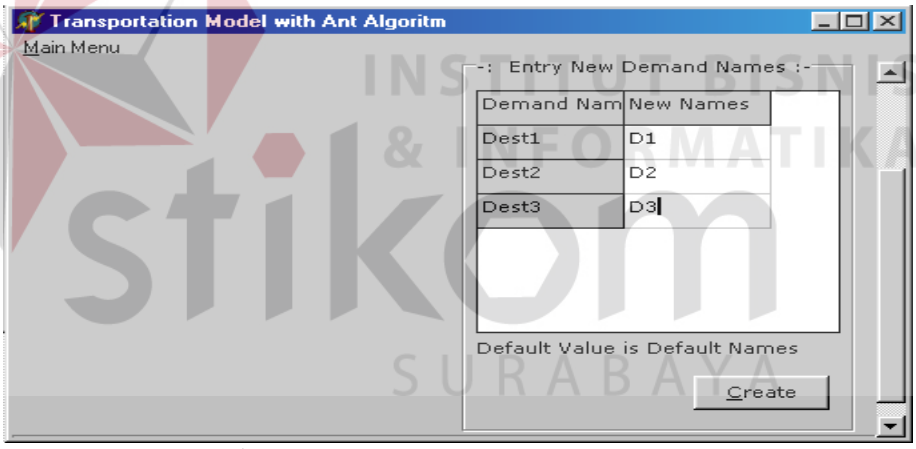
2.c. Tombol Create, tombol ini berfungsi untuk membuat matrik sesuai dengan tujuan dan sumber yang telah dimasukkan. Apabila tombol ini dipilih akan

muncul form pesan Confirm (Gambar 4.5.), untuk memastikan apakah user ingin medefinisikan nama sumber dan tujuan sesuai dengan keinginannya atau tidak.



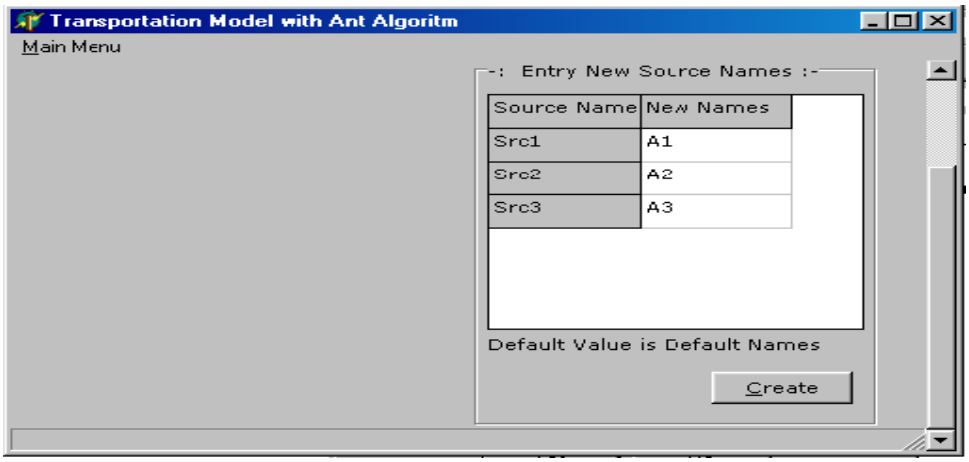
Gambar 4.5. Konfirmasi Nama sumber dan tujuan

Jika ya, akan muncul form Entry New Source Name dan Form Entry New Destination (Gambar 4.6. dan Gambar 4.7.), jika tidak maka nama sumber dan tujuan akan otomatis sesuai default.



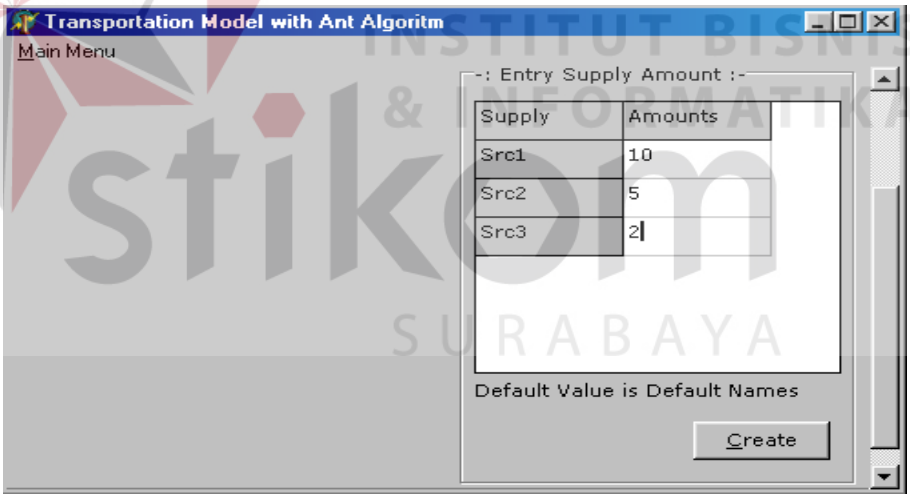
Gambar 4.6. Form Entry New Source

Pada Gambar 4.6. user akan mengisikan nama *source* baru sesuai keinginannya. Setelah tombol create dipilih, akan muncul form Entry New Destination.



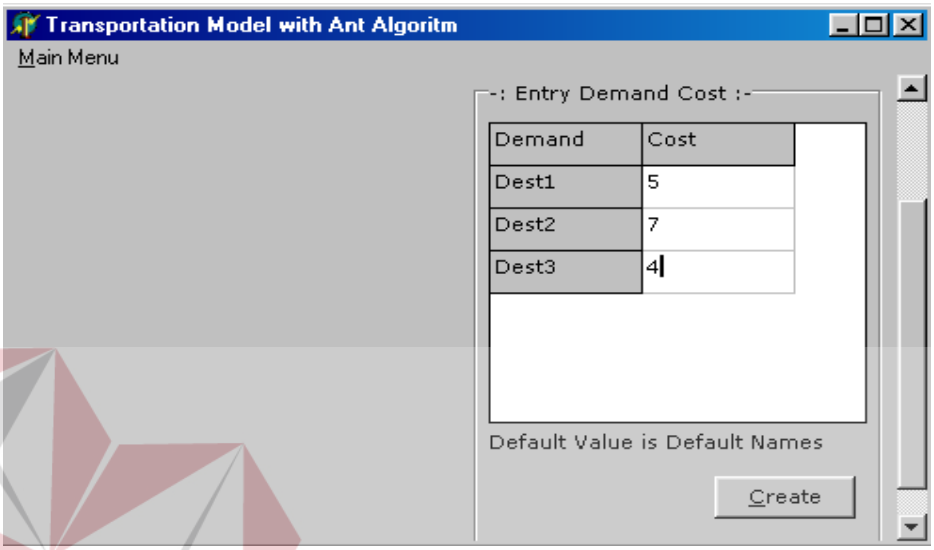
Gambar 4.7. Form Entry New Destination Names

User mengisikan nama *destination* baru sesuai dengan keinginannya. Setelah tombol Create dipilih akan muncul form masukkan besarnya supply dan demand (Gambar 4.8. dan 4.9)



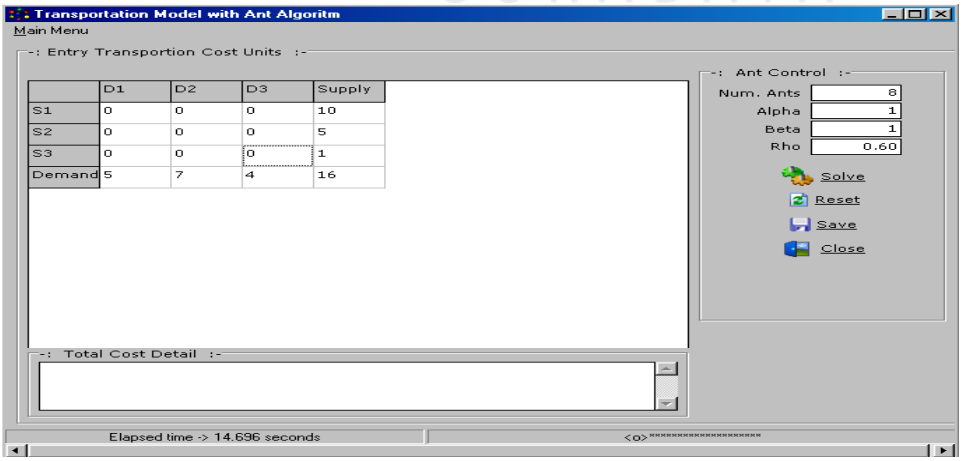
Gambar 4.8. Form Entry Supply Amount

Pada form ini, user memasukkan batasan besarnya supply dan demand dari masalah transportasi. Jumlah batasan yang dimasukkan sesuai dengan jumlah dari destination dan source yang sebelumnya telah dimasukkan.



Gambar 4.9. Form Entry Demand Cost.

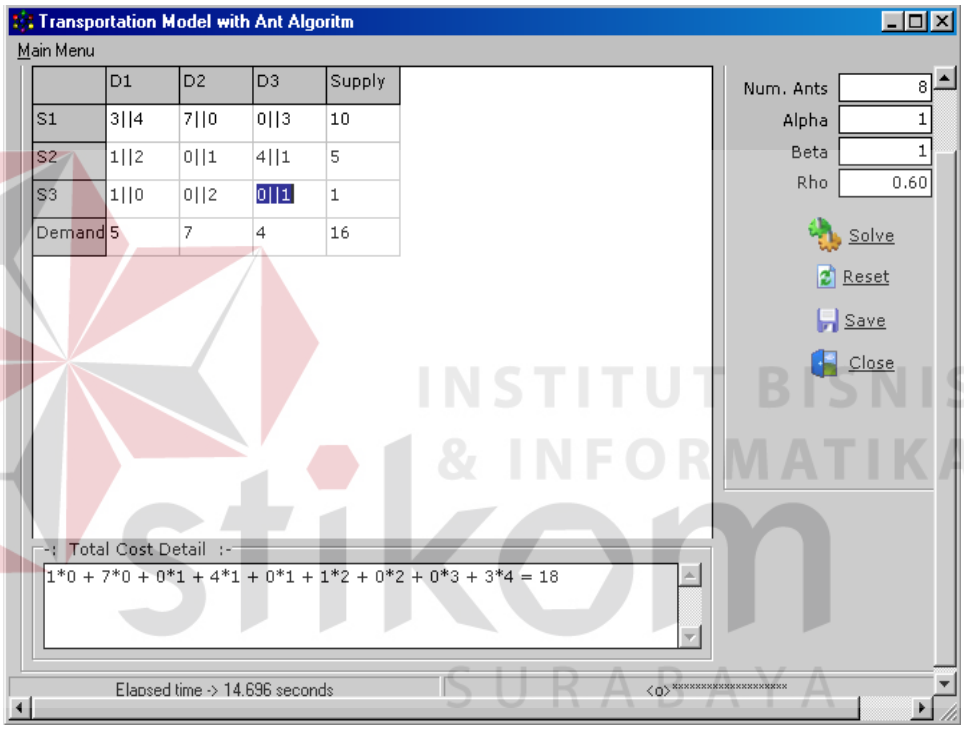
Setelah dilakukan pengisian kedua form tersebut dan tombol dreate dipilih, selanjutnya akan muncul form Transportation Problem (Gambar 4.10.) dimana user harus mengisikan cost tiap unitnya dan juga parameter dari ant colony sistem yang terdiri dari Num ant, Beta, Alpha dan Rho.



Gambar 4.10. Transportation Problem

Pada Gambar 4.10 juga terdapat tombol Solve, Reset, Save dan Close. Dimana masing-masing dapat dijelaskan sebagai berikut:

1. Solve, tombol ini untuk memulai perhitungan masalah transportasi yang telah diinputkan dengan menggunakan Ant Colony Algorithm. Sehingga akan muncul form Hasil seperti pada Gambar 4.11, yang merupakan hasil akhir proses pencarian dari masalah transportasi yang dimasukkan.



Gambar 4.11. Form Hasil Transportation Problem

Hasil akhir dari proses pencarian ini berupa tabel transportasi, total cost dan waktu proses.

2. Save, tombol ini digunakan untuk menyimpan masalah transportasi ke dalam file *.ant yang sewaktu-waktu dapat dibuka kembali pada menu utama Read Exiting Problem.
3. Close, tombol ini digunakan untuk keluar dari form dan kembali ke menu utama.

2.d. Tombol Exit, tombol ini berfungsi untuk keluar dari form Input New Problem dan kembali ke menu utama.

3. Exit, pilihan ini digunakan untuk keluar dari sistem.

4.2 Evaluasi

Pada tahap ini peneliti melakukan evaluasi untuk mengetahui apakah hasil yang diperoleh telah optimal atau tidak, dan membandingkan hasil yang didapat dengan pemecahan masalah transportasi biasa (konvensional). Selain itu evaluasi juga dilakukan pada proses pencarian dan jumlah semut. Data-data yang digunakan pada uji coba diperoleh dari literatur-literatur yang digunakan yaitu Tjutju (2002:124), Johannes Supranto (1988:178) dan data yang dibuat sendiri oleh penulis.

Data Uji coba tersebut adalah:

1. Matrik 3x3 (Tjutju, 2002:124)

Tabel 4.1. Uji coba matrik 3x3

Tujuan / Sumber	A	B	C	Supply
I	6	8	10	150
II	7	11	11	175
III	4	5	12	275
Demand	200	100	300	600

Pada tabel diatas, sebuah perusahaan tegel mempunyai 3 pabrik dan 3 gudang. Pabrik I bisa memproduksi 150 m^2/hari, pabrik II bisa memproduksi 175 m^2/hari, pabrik III 275 m^2/hari. Setelah diproduksi tegel-tegel tersebut akan disimpan ke 3 gudang sesuai dengan kapasitasnya. Gudang A kapasitasnya 200 m^2/hari, gudang B kapsditasnya100 m^2/hari dan gudang C 300 m^2/hari. Biaya untuk mengangkut tegel dari pabrik ke gudang seperti nampak pada Tabel 4.1 (Ket: biaya angkut

barang dalam satuan rupiah). Tentukan jumlah tegel yang harus didistribusikan dari tiap pabrik ke tiap gudang agar biaya pengangkutan minimal.

Penyelesaian:

Dari data yang dimasukkan didapatkan tabel penyesaian sebagai berikut:

Tabel 4.2. Penyelesaian matrik 3x3

Tujuan / Sumber	A	B	C	Supply
I	0 || 6	25 || 8	125 || 10	150
II	0 || 7	0 || 11	175 || 11	175
III	200 || 4	75 || 5	0 || 12	275
Demand	200	100	300	600

Tegel yang harus didistribusikan dari masing-masing pabrik ke masing-masing gudang per hari agar ongkos minimum dapat dilihat dari tabel sebagai berkut:

Tabel 4.3.Pengiriman barang 1

Dari pabrik	Ke gudang	Distribusi barang (Unit)	Biaya/unit (Rp.)
I	A	0	6
	B	25	8
	C	125	10
II	A	0	7
	B	0	11
	C	175	11
III	A	200	4
	B	75	5
	C	0	12

Jadi, total biaya (total cost least) distribusi

= (6x0) + (8x25) + (10x125) + (7x0) + (11x0) + (11x175) + (4x200) +

(5x75) + (12x0)

= 0 + 200 +1250 + 0 + 0+ 1925 + 800 + 375 + 0

= Rp. 4.550

2. Matrik 5x3 (Johannes Supranto, 1988:178)

Tabel 4.4. Uji coba matrik 5x3

Tujuan / Sumber	G1	G2	G3	G4	G5	Supply
P1	50	80	60	60	30	800
P2	40	70	70	60	50	600
P3	80	40	60	60	40	1100
Demand	400	400	500	400	800	2500

Pada tabel diatas terdapat 5 tujuan yaitu gudang G1, G2, G3, G4, G5 dan 3 sumber yaitu pabrik PI, PII, PIII. Dengan batasan bahwa pabrik I dapat menyuplai sebasar 800/hari, pabrik II sebesar 600/hari dan pabrik III sebesar 1100/hari. Sedangkan besar kapasitas (demand) gudang G1 sebesar 400/hari, gudang G2 sebesar 400/hari, gudang G3 sebesar 500/hari, gudang G4 sebesar 400/hari dan gudang G5 sebesar 800/hari. Biaya angkut perunit dari produk seperti nampak pada Tabel 4.2. (Ket: biaya angkut per unit barang dalam satuan rupiah). Berapakah alokasi pengiriman barang dari tiap sumber ke tiap gudang agar didapatkan biaya transportasi yang minimal.

Penyelesaian:

Dari data yang dimasukkan didapatkan tabel penyesaian sebagai berikut:

Tabel 4.5. Penyelesaian Matrik 5x3

Tujuan / Sumber	G1	G2	G3	G4	G5	Supply
P1	0 || 50	0 || 80	0 || 60	0 || 60	800 || 30	800
P2	400 || 40	0 || 70	0 || 70	200 || 60	0 || 50	600
P3	0 || 80	400 || 40	500 || 60	200 || 60	0 || 40	1100
Demand	400	400	500	400	800	2500

Alokasi pengiriman barang dari tiap-tiap sumber ke tiap-tiap gudang adalah:

Tabel 4.6. Pengiriman barang 2

Dari pabrik	Ke gudang	Distribusi barang (Unit)	Biaya/unit (Rp.)
P1	G1	0	50
	G2	0	80
	G3	0	60
	G4	0	60
	G5	80	30
P2	G1	400	40
	G2	0	70
	G3	0	70
	G4	200	60
	G5	0	50
P3	G1	0	80
	G2	400	40
	G3	500	60
	G4	200	60
	G5	0	40

Jadi, total biaya (total cost least) pengiriman

= (50x0) + (80x0) + (60x0) + (60x0) + (30x800) + (40x400) + (70x0) + (70x0) + (60x200) + (50x0) +(80x0) + (40x400) + (60x500) + (60x200) + (40x0)

= 0 + 0 +0 + 0 + 24.000 + 16.000 + 0 + 0 + 12.000 + 0 + 0 + 16.000 + 30.000 + 12.000 + 0

= Rp. 110.000

3. Matrik 10x10

Tabel 4.7. Uji coba matrik 10x10

Tuj / Sum	A	B	C	D	E	F	G	H	I	J	Sup
1	3	5	1	2	2	8	1	2	4	9	100
2	10	2	4	1	1	7	5	1	3	1	75
3	2	1	2	4	4	1	2	5	1	3	125
4	1	2	6	6	6	1	1	1	2	4	50
5	2	2	3	3	1	1	1	6	8	9	25
6	1	5	4	4	2	3	4	2	3	1	75
7	6	3	11	6	11	4	1	8	5	1	100
8	8	1	6	8	6	5	1	7	6	2	50
9	7	2	1	1	1	6	2	1	1	3	25
10	1	2	1	2	1	1	2	6	2	5	125
Dm	125	25	50	100	75	25	50	125	75	100	800

Pada tabel diatas terdapat 10 sumber dan 10 tujuan, dimana masing-masing mempunyai batasan dan nilai biaya per unit sendiri-sendiri seperti terlihat pada tabel tersebut. (Ket: biaya angkut per unit barang dalam satuan rupiah). Tentukan biaya transportasi pengiriman barang dari masing-masing sumber ke masing-masing tujuan.

Penyelesaian:

Dari data yang dimasukkan didapatkan tabel penyesaian sebagai berikut:

Tabel 4.8. Penyelesaian Matrik 10x10

Tuj / Sum	A	B	C	D	E	F	G	H	I	J	Sup
1	0 || 3	0 || 5	50 || 1	0 || 2	0 || 2	0 || 8	50 || 1	0 || 2	0 || 4	0 || 9	100
2	0 || 10	0 || 2	0 || 4	75 || 1	0 || 1	0 || 7	0 || 5	0 || 1	0 || 3	0 || 1	75
3	0 || 2	25 || 1	0 || 2	0 || 4	0 || 4	25 || 1	0 || 2	0 || 5	75 || 1	0 || 3	125
4	50 || 1	0 || 2	0 || 6	0 || 6	0 || 6	0 || 1	0 || 1	0 || 1	0 || 2	0 || 4	50
5	0 || 2	0 || 2	0 || 3	0 || 3	25 || 1	0 || 1	0 || 1	0 || 6	0 || 8	0 || 9	25
6	75 || 1	0 || 5	0 || 4	0 || 4	0 || 2	0 || 3	0 || 4	0 || 2	0 || 3	0 || 1	75
7	0 || 6	0 || 3	0 || 11	0 || 6	0 || 11	0 || 4	0 || 1	0 || 8	0 || 5	100 || 1	100
8	0 || 8	0 || 1	0 || 6	0 || 8	0 || 6	0 || 5	0 || 1	50 || 7	0 || 6	0 || 2	50
9	0 || 7	0 || 2	0 || 1	25 || 1	0 || 1	0 || 6	0 || 2	0 || 1	0 || 1	0 || 3	25
10	0 || 1	0 || 2	0 || 1	0 || 2	50 || 1	0 || 1	0 || 2	75 || 6	0 || 2	0 || 5	125
Dm	125	25	50	100	75	25	50	125	75	100	800

Biaya pengangkutan dari tiap-tiap sumber ke tiap-tiap gudang adalah:

Tabel 4.9. Pengiriman barang 3

Dari pabrik	Ke gudang	Distribusi barang (Unit)	Biaya/unit (Rp.)
1	A	0	3
	B	0	5
	C	50	1
	D	0	2
	E	0	2
	F	0	8
	G	50	1
	H	0	2
	I	0	4
	J	0	9
2	A	0	10
	B	0	2
	C	0	4
	D	75	1
	E	0	1
	F	0	7
	G	0	5
	H	0	1
	I	0	3
	J	0	1
3	A	0	2
	B	25	1
	C	0	2
	D	0	4
	E	0	4
	F	25	1
	G	0	2
	H	0	5

	I	75	1
	J	0	3
4	A	50	1
	B	0	2
	C	0	6
	D	0	6
	E	0	6
	F	0	1
	G	0	1
	H	0	1
	I	0	2
	J	0	4
5	A	0	2
	B	0	2
	C	0	3
	D	0	3
	E	25	1
	F	0	1
	G	0	1
	H	0	6
	I	0	8
	J	0	9
6	A	75	1
	B	0	5
	C	0	4
	D	0	4
	E	0	2
	F	0	3
	G	0	4
	H	0	2
	I	0	3

	J	0	1
7	A	0	6
	B	0	3
	C	0	11
	D	0	3
	E	0	11
	F	0	4
	G	0	1
	H	0	8
	I	0	5
	J	100	1
8	A	0	8
	B	0	1
	C	0	6
	D	0	8
	E	0	6
	F	0	5
	G	0	1
	H	50	7
	I	0	6
	J	0	2
9	A	0	7
	B	0	2
	C	0	1
	D	25	1
	E	0	1
	F	0	6
	G	0	2
	H	0	1
	I	0	1
	J	0	3

10	A	0	1
	B	0	2
	C	0	1
	D	0	2
	E	50	1
	F	0	1
	G	0	2
	H	75	6
	I	0	2
	J	0	5

Jadi, total biaya (total cost least) transportasi

= (3x0) + (5x0) + (1x50) + (2x0) + (2x0) + (8x0) + (1x50) + (2x0) + (4x0) + (9x0) + (10x0) + (2x0) + (4x0) + (75x1) + (1x0) + (7x0) + (5x0) + (1x0) + (3x0) + (1x0) + (2x0) + (25x1) + (2x0) + (4x0) + (4x0) + (25x1) + (2x0) + (5x0) + (75x1) + (3x0) + (50x1) + (2x0) + (6x0) + (6x0) + (6x0) + (1x0) + (1x0) + (1x0) + (2x0) + (4x0) + (2x0) + (2x0) + (3x0) + (3x0) + (25x1) + (1x0) + (1x0) + (6x1) + (8x0) + (9x0) + (75x1) + (5x0) + (4x0) + (4x0) + (2x0) + (3x0) + (4x0) + (2x0) + (3x0) + (1x0) + (6x0) + (3x0) + (11x0) + (6x0) + (11x0) + (4x0) + (1x0) + (8x1) + (5x0) + (1x100) + (8x0) + (1x0) + (6x0) + (8x0) + (6x0) + (5x0) + (1x0) + (7x50) + (6x0) + (2x0) + (7x0) + (7x0) + (2x0) + (1x0) + (1x25) + (1x0) + (6x0) + (2x0) + (1x0) + (1x0) + (3x0) + (1x0) + (2x0) + (1x0) + (2x0) (50x1) + (1x0) + (2x0) + (6x75) + (2x0) + (5x0)

= Rp. 1.425

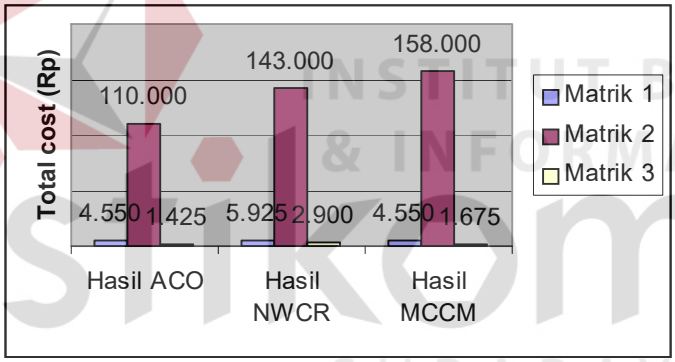
a. Uji coba untuk hasil pencarian optimal

Uji coba ini dilakukan untuk mengetahui hasil optimal dari ACO dan hasilnya akan dibandingkan dengan hasil optimal cara konvensional. Dari serangkaian uji coba yang telah dilakukan maka didapat hasil sebagai berikut:

Tabel 4.10. Hasil pencarian optimal

Matrik	Hasil ACO	Hasil NWCR	Hasil MCCM
1	4.550	5.925	4.550
2	110.000	143.000	158.000
3	1.425	2.900	1.675

Pada tabel diatas terlihat bahwa hasil pemecahan masalah transportasi menggunakan ACO lebih optimal dibandingkan dengan penyelesaian cara biasa (konvensional).



Gambar 4.12. Perbandingan hasil ACO dengan konvensional.

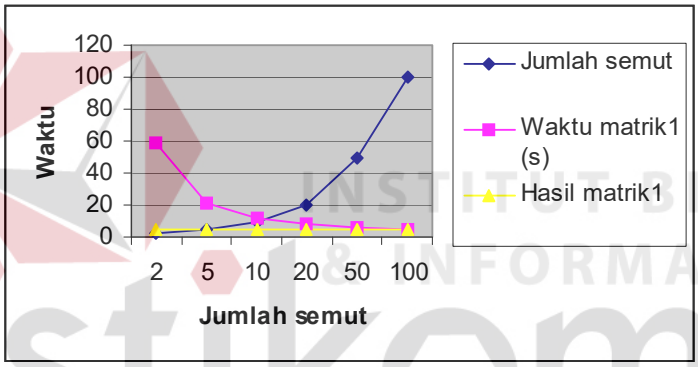
b. Uji coba untuk jumlah semut terhadap waktu dan hasil.

Uji coba ini dilakukan untuk mengetahui waktu proses pencarian terhadap parameter jumlah semut. Dari serangkaian uji coba yang telah dilakukan maka didapat hasil sebagai berikut:

Tabel 4.11. Perbandingan jumlah semut terhadap waktu dan hasil.

Jumlah semut	Waktu matrik1 (s)	Hasil matrik1	Waktu matrik2 (s)	Hasil matrik2	Waktu matrik3 (s)	Hasil matrik3
2	58.690	4.550	45.070	110.000	57.414	1.425
5	20.690	4.550	20.780	110.000	31.549	1.425
10	12.106	4.550	12.701	110.000	22.926	1.425
20	8.044	4.550	6.231	110.000	18.78	1.425
50	5.621	4.550	5.436	110.000	16.066	1.425
100	4.794	4.550	4.545	110.000	15.221	1.425

Pada tabel diatas dapat dilihat bahwa jumlah semut yang dimasukkan akan berpengaruh terhadap waktu proses pencarian tetapi tidak berpengaruh terhadap hasil cost least. Ini berarti pencarian dengan jumlah semut sedikitpun telah mendapatkan hasil yang optimal.



Gambar 4.13. Pengaruh jumlah semut terhadap waktu pada matrik 1

c. Uji coba ketetapan hasil optimal

Uji coba ini dilakukan untuk mengetahui apakah hasil yang diperoleh sudah teroptimal atau tidak, dengan melakukan pengujian berulang-ulang pada data yang sama. Dari data uji coba yang telah diinputkan maka didapatkan hasil sebagai berikut:

Tabel 4.12. Hasil optimal

Matrik	Banyak percobaan	Hasil ACO	Hasil NWCR	Hasil MCCM
1	1	4.550	5.925	4.550
	5	4.550	5.925	4.550
	10	4.550	5.925	4.550
	15	4.550	5.925	4.550
	20	4.550	5.925	4.550
2	1	110.000	143.000	158.000
	5	110.000	143.000	158.000
	10	110.000	143.000	158.000
	15	110.000	143.000	158.000
	20	110.000	143.000	158.000
3	1	1.425	2900	1.675
	5	1.425	2900	1.675
	10	1.425	2900	1.675
	15	1.425	2900	1675
	20	1.425	2900	1675

Dari tabel diatas dapat dilihat bahwa pengulangan percobaan pada data yang sama menghasilkan nilai yang sama, ini berarti hasil yang diperoleh dari penyelesaian masalah transportasi dengan menggunakan *Ant Colony Algorithm* adalah optimal.