UNIVERSITAS INDONESIA

PENDEKATAN ALGORITMA *AUCTION* PADA MASALAH PENJADWALAN KENDARAAN ANGKUTAN UMUM KOTA DENGAN WAKTU PERJALANAN DINAMIK

SKRIPSI

NITA ASTUTI

0806452261

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

PROGRAM SARJANA MATEMATIKA

DEPOK

NOVEMBER 2012

UNIVERSITAS INDONESIA

PENDEKATAN ALGORITMA *AUCTION* PADA MASALAH PENJADWALAN KENDARAAN ANGKUTAN UMUM KOTA DENGAN WAKTU PERJALANAN DINAMIK

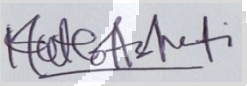
SKRIPSI
Diajukan sebagai salah satu syarat untuk memperoleh gelar sarjana sains

NITA ASTUTI
0806452261

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
PROGRAM SARJANA MATEMATIKA
DEPOK
NOVEMBER 2012

HALAMAN PERNYATAAN ORISINALITAS

Skripsi ini adalah hasil karya saya sendiri dan semua sumber baik yang dikutip maupun dirujuk telah saya nyatakan dengan benar.

Nama	: Nita Astuti
NPM	: 0806452261
Tanda Tangan	: 
Tanggal	: 20 November 2012

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh
Nama : Nita Astuti
NPM : 0806452261
Program Studi : Matematika
Judul Skripsi : Pendekatan Algoritma *Auction* pada Masalah Penjadwalan Kendaraan Angkutan Umum Kota dengan Waktu Perjalanan Dinamik

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Sains pada Program Studi Sarjana Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Indonesia

DEWAN PENGUJI

Pembimbing : Helen Burhan,M.Si ( )

Penguji : Arie Wibowo, M.Si ( )

Penguji : Dra. Denny Riama Silaban, M.Kom ( )

Penguji : Dhian Widya, M.Kom ( )

Ditetapkan di : Depok
Tanggal : 20 November 2012

KATA PENGANTAR

Alhamdulillah, puji syukur kepada Allah SWT atas segala rahmat dan karunia yang Ia berikan sehingga penulis dapat menyelesaikan skripsi ini. Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Sains jurusan matematika pada Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Indonesia. Tanpa bantuan dan bimbingan dari berbagai pihak, dari masa perkuliahan sampai pada penyusunan skripsi ini, sangatlah sulit bagi penulis untuk menyelesaikan skripsi ini. Oleh karena itu, penulis mengucapkan terima kasih kepada:

(1) Orang tua (Ayah dan Mama) penulis yang tidak henti-hentinya mendoakan untuk kesuksesan dan kelancaran penulis dalam menghadapi segala urusan, serta kesabaran mereka untuk menunggu penulis lulus di waktu yang telah ditakdirkan oleh Allah SWT. Semoga kalian selalu hadir disaat-saat bahagia penulis.

(2) Ibu Helen, M.Si. selaku pembimbing skripsi penulis. Terima kasih atas bimbingan, diskusi, kesabaran, serta perhatian Bu Helen dalam menghadapi penulis dalam menyelesaikan skripsi ini.

(3) Ibu Dra. Saskya Mary Soemartodjo, M.Si. selaku pembimbing akademik penulis. Tanpa lelah memberikan nasihat-nasihat perkuliahan yang sangat membantu penulis dalam menjalani perkuliahan dan tetap bertahan menjalani perkuliahan di matematika.

(4) Ibu Dra. Denny Riama Silaban, M.Kom; Ibu Dhian Widya, M.Kom; Bapak Arie Wibowo, M.Si sebagai penguji kolokium yang telah memberikan saran membangun untuk penulis mengenai penulisan skriipsi ini.

(5) Seluruh staf pengajar di Departemen Matematika UI. Terima kasih atas ilmu yang telah diajarkan. Semoga penulis dapat memanfaatkan dan mengaplikasikan ilmu tersebut dengan baik.

(6) Seluruh karyawan di Departemen Matematika UI. Terima kasih atas bantuan dan kemudahan yang diberikan kepada penulis selama penulis kuliah di Matematika UI.

(7) Uda, Abang, dan Uni yang merupakan kakak-kakak dari penulis yang selalu

mendukung penulis dalam setiap kegiatan positif dan memberikan semangat kepada penulis untuk menyelesaikan skripsi ini. Terima kasih juga penulis ucapkan untuk anak kecil bergigi *ompong* yang selalu dapat dijaili dan memberikan keceriaan di rumah, yaitu keponakan penulis, sasa.

(8) Laili Miftahur Rizqi (QQ) sebagai teman seperjuangan terkasih. Tidak ada kata lelah walau cobaan dan air mata itu selalu ada, selalu semangat memberikan energi positif bagi penulis untuk menyelesaikan tulisan ini. Semoga sesuatu yang indah itu akan segera datang. Aamiin.

(9) Bang Andy yang sudah sangat membantu penulis dalam menyelesaikan program matlab dalam tulisan ini, menjadi tutor yang baik, dan ketua angkatan yang loyal kepada angkatannya.

(10) Koh Hendry, Kak Edy Setiawan, dan Om Dedet yang telah memberikan masukkan membangun kepada penulis mengenai penulisan dalam skripsi ini.

(11) Ahmad Maimun dan kak Ajat yang membantu penulis dalam memahami materi perkuliahan yang sulit untuk dipahami. Terima kasih atas kesabaran kalian dalam menghadapi penulis dimasa perkuliahan.

(12) Kak Dessanti yang telah menjadi tempat bercerita, membimbing, dan mengajarkan banyak hal kepada penulis sejak SMA hingga saat ini.

(13) Teman-teman Matematika UI 2008, Uni Achi, Ade, Adhi, Agnes, Agy, Bang Andy, Awe, Anggi, Anisah, Arief, Arkies, Arman, Bowo, Cindy, Cici Citra, Danis, Dede, Dhea, Dheni, Dhila, Dian, Dini, Mbak Ega, Eka, Emy, Juni, Fani, Koh Hen, Hindun, Icha, Ifah, Ines, Janu, Lian, Mbak Luthfa, Maimun, Masykur, Maul, Mayta, Mei, Nadia, Nora, Numa, Oyin, Puput, Purwo, QQ, Ramos, Resti, Risya, Sita, Mbak Siwi, Tute, Uchid, Ucil, Umbu, Vika, Wulan, dan Yulial. Terima kasih telah menjadi teman-teman yang baik, selalu membantu penulis dalam menjalani perkuliahan, dan semangat, serta motivasi yang sangat membangun. Peluk cium untuk kalian.

(14) Cahyo Baskoro yang tidak pernah berhenti dalam memberikan semangat kepada penulis dalam menyelesaikan skripsi ini.

(15) Saras Ayu Heidiana sebagai adik asuh dan Elvin Marwady sebagai adik asuh angkat yang selalu berdoa dan memberikan semangat bagi penulis dalam menyelesaikan skripsi ini.

(16) Kak Arif Auzandi yang telah menganggap penulis seperti adik sendiri dan selalu memotivasi penulis dalam menjalani perkuliahan, serta nasihat-nasihat perkuliahan yang membangun bagi penulis. Terima kasih atas segala kebaikan dan perhatian kakak.

(17) Kakak-kakak angkatan 2004, 2005, 2006, dan 2007 yang telah memberikan semangat dan bantuan kepada penulis.

(18) Teman-teman dan adik-adik angkatan 2009, 2010, dan 2011 yang telah memberikan semangat, dukungan, dan doa kepada penulis.

(19) Kawan-kawan *wardep* Icha, Mia, Rifa, Ratna, Oya, Adam, Ramandha, Chandra, Madan, Eki, Ojiek, Fachrie, Popon, Achip, Acid, Baba, Vito yang selalu memberikan kenangan indah dan kekonyolan masa SMA hingga saat ini.

(20) Kisun dan The G-BLAQ : Ika, Adhani, dan Ade. Terima kasih atas dukungan kalian kepada penulis dan keceriaan yang diberikan kepada penulis dalam menyelesaikan skripsi ini.

(21) Semua pihak yang telah memberikan doa dan semangat kepada penulis yang tidak bisa disebutkan satu per satu.

Akhir kata, penulis mohon maaf apabila ada kesalahan atau kekurangan dalam skripsi ini. Penulis berharap semoga tulisan ini bermanfaat bagi para pembaca.

Penulis

2012

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

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

Nama	: Nita Astuti
NPM	: 0806452261
Program Studi	: Sarjana Matematika
Departemen	: Matematika
Fakultas	: Matematika dan Ilmu Pengetahuan Alam
Jenis karya	: Skripsi

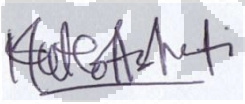
demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty Free Right*) atas karya ilmiah saya yang berjudul :

Pendekatan Algoritma *Auction* pada Masalah Penjadwalan Kendaraan Angkutan Umum Kota dengan Waktu Perjalanan Dinamik

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalih media/format-kan, mengelola dalam bentuk pangkalan data (database), merawat dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok
Pada tanggal : 20 November 2012
Yang menyatakan



(Nita Astuti)

ABSTRAK

Nama : Nita Astuti
Program Studi : Sarjana Matematika
Judul : Pendekatan Algoritma *Auction* pada Penjadwalan Angkutan Umum Kota dengan Mempertimbangkan Waktu Perjalanan Dinamik

Salah satu tahap dalam proses perencanaan pengoperasian angkutan umum adalah mengatur jadwal tugas kendaraan terhadap sejumlah perjalanan (*trip*) yang berasal dari jam keberangkatan (*timetable*). Kondisi lalu lintas yang padat di perkotaan menyebabkan waktu tempuh perjalanan yang tidak tetap (dinamik), hal ini dapat mengakibatkan keterlambatan jam keberangkatan angkutan umum dari jadwal yang telah ada. Oleh karena itu perlu dibuat suatu penjadwalan kendaraan dengan mempertimbangkan situasi yang dinamik.Pada skripsi ini akan dibahas masalah penjadwalan kendaraan angkutan umum dengan mempertimbangkan waktu perjalanan yang dinamik atau *dynamic vehicle scheduling problem* (D-VSP). D-VSP akan menggunakan model matematika berbentuk masalah penugasan *quasi*. Kemudian masalah penjadwalan ini akan diselesaikan dengan algoritma *auction*. Algoritma ini berprinsip pada metode pelelangan dimana pemberi *bid* terbesar akan dipilih dalam menentukan barisan perjalanan yang dilalui kendaraan.

Kata Kunci : Masalah Penjadwala Kendaraan Umum, Masalah Penjadwalan Kendaraan Umum Dinamik, Masalah Penugasan *Quasi*, Algoritma *Auction*.
xiv+80 halaman : 12 gambar + 4 tabel
Daftar Pustaka : 13 (1990-2007)

ABSTRACT

Name : Nita Astuti
Study Program : Sarjana Matematika
Title : Auction Algorithm Approach to Urban Public Transport Scheduling Considering The Non Fixed Travel Time

One of stages in planning process of operating public transport is arranging task schedule of vehicle towards amount of trips from the departure time (timetable). The congested traffic condition in urban area leads to travel time that is not fixed (dynamic), so it can cause delay the departure time from the schedule that was determined. Therefore it is imperative to set a scheduling to overcome the dynamic condition. In this *skripsi* will discuss about vehicle scheduling problem by considering the dynamic travel time or dynamic vehicle scheduling problem (D-VSP). The DVSP will apply the mathematical model of quasi assignment problem. Afterwards, this problem will solve with auction algorithm. This algorithm is based on auction method, where the biggest bidder will be chosen to determine the path that is taken by vehicle.

Keywords : Vehicle Scheduling Problem, Dynamic Vehicle Scheduling Problem, Quasi Assignment Problem, Auction Algorithm.
xiv+80 pages : 12 pictures + 4 tables
Bibliography : 13 (1990-2007)

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Salah satu moda angkutan umum yang menjadi sarana transportasi di Jakarta adalah bus kota. Sebagian besar bus kota yang beroperasi merupakan milik perorangan (pribadi), contohnya Metromini dan Kopaja. Sistem pengoperasian dalam kepemilikan perorangan disebut sebagai sistem setoran atau sistem sewa. Pada sistem setoran, pemilik atau pengusaha angkutan umum menyewakan kendaraan pada pengemudi, lalu mendapatkan bayaran berupa uang setoran sebagai biaya sewa. Pemilik kendaraan menetapkan tarif setoran dengan kebijakan masing-masing, bukan dengan sistem kesepakatan. Oleh karena itu, banyak pengemudi yang mengoperasikan kendaraan secara berlebihan dengan ketidakteraturan waktu keberangkatan.

Selain pengoperasian bus secara berlebihan, banyaknya jumlah trayek yang saling tumpang tindih dan perilaku pengemudi bus dalam menaikkan atau menurunkan penumpang tidak pada tempat pemberhentian yang seharusnya menyebabkan pengoperasian angkutan umum di Jakarta tidak optimal. Sehingga diperlukan suatu perencanaan pelayanan pengoperasian angkutan umum yang disesuaikan dengan keadaan kota Jakarta untuk mengoptimalkan pengoperasian kendaraan angkutan umum.

Perencanaan pelayanan pengoperasian angkutan umum yang baik mempunyai beberapa tahapan, dimana hasil dari tahapan sebelumnya digunakan untuk tahapan berikutnya. Tahap awal adalah pengaturan rute angkutan (*routing*) dan perhitungan banyaknya (frekuensi) angkutan yang melewati suatu rute. Tahap berikutnya adalah pengaturan jadwal jam keberangkatan (*timetabling*) masing-masing rute. Berdasarkan jam keberangkatan yang telah didapat akan diperoleh sejumlah perjalanan (*trip*) dan dilanjutkan dengan tahap penugasan kendaraan terhadap *trip* atau disebut dengan *vehicle scheduling*. Tahap terakhir adalah penugasan *crew* terhadap kendaraan angkutan yang akan beroperasi (Huisman, 2004).

Ppenjadwalan kendaraan atau *scheduling problem* (VS) yang biasa digunakan adalah masalah penjadwalan kendaraan statik atau s*tatic vehicle scheduling problem* (S-VSP) dimana masalah penjadwalan ini mengasumsikan waktu perjalanan yang tetap. Sementara situasi lalu lintas yang padat di perkotaan, seperti di kota Jakarta, dapat menyebabkan waktu tempuh perjalanan yang tidak tetap (dinamik). Hal ini dapat mengakibatkan keterlambatan jam keberangkatan angkutan umum yang telah ada. Oleh sebab itu, untuk mengatasi waktu perjalanan tidak tetap dilakukan penjadwalan kendaraan dinamik atau *dynamic vehicle scheduling problem* (D-VSP) (Huisman, 2004). Pada penulisan skripsi ini akan dibahas masalah penjadwalan angkutan umum dengan mempertimbangkan keadaan dinamik dan diselesaikan dengan algoritma *auction*.

1.2 Perumusan Masalah dan Pembatasan Masalah

Perumusan Masalah:

1. Bagaimana model penjadwalan kendaraan angkutan umum kota dengan mempertimbangkan waktu perjalanan yang tidak tetap (dinamik) sehingga dapat mengoptimalkan biaya operasional penggunaan kendaraan?
2. Bagaimana pendekatan algoritma *auction* dalam mendapatkan solusi pada model penjadwalan kendaraan angkutan umum kota yang telah didapatkan?
3. Bagaimana implementasi pendekatan algoritma *auction* untuk menyelesaikan masalah penjadwalan kendaraan angkutan umum Jakarta dengan bantuan perangkat lunak?

Pembatasan Masalah:

Masalah penjadwalan kendaraan angkutan umum dinamik dalam skripsi ini akan diaplikasikan pada penjadwalan kendaraan angkutan umum kota Jakarta dengan batasan sebagai berikut:

1. Terdapat satu *depot*, baik sebagai tempat keberangkatan awal kendaraan maupun pemberhentian terakhir kendaraan.
2. Kendaraan yang digunakan seragam (homogen), yaitu bus ukuran sedang dengan kapasitas maksimum 40 orang.

3. Rute angkutan umum yang digunakan hanya dua rute, yaitu:
    - Dari Manggarai ke Blok M atau sebaliknya.
    - Dari Manggarai ke Pasar Minggu atau sebaliknya.
4. Pada rute yang digunakan dimungkinkan terjadi penukaran rute (*interlining*) di Manggarai.

1.3 Jenis Penelitian dan Metode Penelitian

Jenis penelitian yang digunakan dalam pembuatan tugas akhir ini adalah studi literatur. Metode yang digunakan untuk menyelesaikan masalah penjadwalan angkutan umum kota dengan mempertimbangkan waktu perjalanan dinamik adalah algoritma *auction* dan untuk memperoleh solusi dari masalah penjadwalan tersebut digunakan bantuan perangkat lunak MATLAB R2009a.

1.4 Tujuan Penulisan

Tujuan penulisan dalam skripsi ini adalah:

- Mengetahui model penjadwalan kendaraan angkutan umum kota dengan mempertimbangkan waktu perjalanan yang tidak tetap (dinamik) sehingga dapat mengoptimalkan biaya operasional penggunaan kendaraan.
- Menjelaskan pendekatan algoritma *auction* untuk menyelesaikan model penjadwalan kendaraan angkutan umum kota yang telah didapatkan.
- Implementasi pendekatan algoritma *auction* untuk menyelesaikan masalah penjadwalan kendaraan angkutan umum Jakarta dengan bantuan perangkat lunak.

BAB 2
LANDASAN TEORI

Pada bab ini akan dibahas mengenai beberapa teori yang digunakan dalam menyelesaikan masalah penjadwalan kendaraan atau *vehicle scheduling problem* (VSP) dan VSP dengan mempertimbangkan waktu perjalanan yang dinamik atau *dynamic vehicle scheduling problem* (D-VSP). Beberapa teori tersebut meliputi pemrograman linear, pemrograman linear bilangan bulat, pemrograman linear bilang biner, teori graf, model masalah penugasan *quasi*, teori transportasi untuk masalah penjadwalan kendaraan dan algoritma *auction* yang akan digunakan untuk menyelesaian model D-VSP.

2.1 Pemrograman Linear

Masalah pemrograman linear dapat didefinisikan sebagai masalah mengoptimalkan suatu fungsi tujuan yang linear terhadap fungsi kendala yang linear (Hillier dan Lieberman, 1995). Masalah pemprogram linear secara umum merupakan bagian dari riset operasional yang didalamnya terdapat tiga elemen dasar yang penting yaitu:

1. Variabel keputusan
2. Fungsi objektif atau fungsi tujuan
3. Kendala

Berikut merupakan bentuk umum dari permasalahan LP

Minimum (Min) $$f = \sum_{j=1}^{n} c_j x_j \qquad (2.1)$$

dengan syarat (d.s) $$\sum_{j=1}^{n} a_{ij} x_j \geq b_i \; ; i = 1,2,3, \dots, m \qquad (2.2)$$

$$x_j \geq 0 \; ; j = 1,2,3, .., n \qquad (2.3)$$

Notasi f merupakan nilai dari keseluruhan tujuan pemodelan, biasanya dalam bentuk biaya. Notasi x_j disebut variabel keputusan dimana variabel ini

menunjukkan tingkatan aktivitas ke-j ($j = 1,2, \dots, n$). Notasi c_j menunjukkan variabel biaya pada aktivitas ke-j. Notasi b_i menunjukkan banyaknya sumber ke-i ($i = 1,2, \dots, m$) yang tersedia untuk keseluruhan aktivitas yang dilakukan. Notasi a_{ij} menunjukkan banyaknya sumber ke-i yang dibutuhkan untuk melakukan aktivitas ke-j. Pertidaksamaan (2.2) disebut fungsi kendala fungsional, sedangkan pertidaksamaan (2.3) disebut kendala nonnegatif untuk variabel keputusan x_j (Hillier & Lieberman, 1995).

Masalah LP (2.1) - (2.3) dapat dinyatakan dalam bentuk matriks, yaitu sebagai berikut:

$$\text{Min} \qquad f = \mathbf{c}^{\mathrm{T}}\mathbf{x} \qquad (2.4)$$

$$\text{d.s} \qquad \mathbf{Ax} \geq \mathbf{b} \qquad (2.5)$$

$$\mathbf{x} \geq \mathbf{0} \qquad (2.6)$$

Dimana vektor $\mathbf{c^T}$ adalah vektor berukuran $1 \times n$ dengan entri koefisien biaya pada fungsi tujuan, $\mathbf{x}$ adalah matriks berukuran $n \times 1$, matriks $\mathbf{A}$ adalah matriks berukuran $m \times n$, dan $\mathbf{b}$ berukuran $m \times 1$ (Hillier dan Lieberman, 1995).

Setiap masalah pemrograman linear dapat diubah ke dalam bentuk standar dimana semua kendala dalam bentuk persamaan dan semua variabel bernilai nonnegatif. Berikut merupakan langkah-langkah yang dilakukan untuk mengubah masalah pemrograman linear menjadi bentuk standar:

1. Jika kendala dalam bentuk pertidaksamaan $\leq$, maka tambahkan variabel *slack*, yaitu s_i pada ruas kiri kendala ke-i dengan $s_i \geq 0$. Lalu, tambahkan $0s_i$ ke fungsi tujuan.
2. Jika kendala dalam bentuk pertidaksamaan $\geq$, maka tambahkan variabel *surplus* r_i pada ruas kanan kendala kendala ke-i dengan $r_i \geq 0$. Lalu, tambahkan $0r_i$ ke fungsi tujuan.

Sehingga bentuk standar dari pemrograman linear adalah sebagai berikut:

$$\text{Min} \qquad f = \mathbf{c^T x} \qquad (2.7)$$

$$\text{d.s} \qquad \mathbf{Ax} = \mathbf{b} \qquad (2.8)$$

$$\mathbf{x} \geq \mathbf{0} \qquad (2.9)$$

Himpunan nilai-nilai yang memenuhi kendala (2.7) dan (2.9) disebut sebagai solusi layak, sedangkan solusi layak yang mengoptimalkan fungsi tujuan disebut sebagai solusi optimal (Hillier & Lieberman, 1995).

2.2 Masalah Dual

Setiap masalah program linear (disebut masalah primal) dalam bentuk standar berkaitan dengan masalah program linear lain disebut sebagai masalah dual. Masalah primal dan dual ini saling berhubungan, yaitu setiap solusi layak dari masalah yang satu menghasilkan suatu batas dari nilai optimal masalah yang lainnya. Misalkan bentuk umum dari masalah primal adalah masalah LP (2.4)-(2.6). Sehingga, bentuk masalah dual dari masalah primal (2.4)-(2.6) adalah sebagai berikut (Dantzig & Thapa, 1997):

Maks $g = \mathbf{b}^T\mathbf{y}$ (2.10)

d.s $\mathbf{A}^T\mathbf{y} \leq \mathbf{c}$ (2.11)

$\mathbf{y} \geq \mathbf{0}$ (2.12)

Hubungan antara masalah primal (2.4)-(2.6) dan maslah dual (2.10)-(2.12) adalah sebagai berikut:

1. Sifat dualitas lemah (*weak duality property*): Jika **x** adalah solusi layak untuk masalah primal (2.4)-(2.6) dan **y** adalah solusi layak untuk masalah dual (2.10)-(2.12), maka terdapat hubungan $\mathbf{b}^T\mathbf{y} \leq \mathbf{c}^T\mathbf{x}$.
2. Sifat dualitas kuat (*strong duality property*): Jika $\mathbf{x}^*$ adalah solusi optimal dari masalah primal (2.4)-(2.6) dan $\mathbf{y}^*$ adalah solusi optimal dari masalah dual (2.10)-(2.12), maka berlaku hubungan $\mathbf{b}^T\mathbf{y}^* = \mathbf{c}^T\mathbf{x}^*$.
   (Hillier & Lieberman, 1995)

Teorema Dualitas Masalah primal dan dual memiliki hubungan antara lain sebagai berikut:

1. Jika salah satu masalah mempunyai solusi layak dan fungsi tujuan terbatas sehingga memiliki suatu solusi optimal, maka berlaku juga terhadap masalah lainnya.
2. Jika salah satu masalah mempunyai solusi layak dan fungsi tujuan tidak mempunyai batasan sehingga tidak memiliki solusi optimal maka masalah lainnya tidak mempunyai solusi layak.
3. Jika salah satu masalah tidak mempunyai solusi layak, maka masalah lainnya juga tidak mempunyai solusi layak atau fungsi tujuan tidak terbatas.

(Hillier & Lieberman, 1995)

2.3 Masalah Pemrograman Linear Bilangan Bulat

Masalah pemprograman linear bilangan bulat ada karena nilai variabel keputusan harus terbatas dalam nilai bilangan bulat. Secara matematis, pemrograman linear bilangan bulat dapat dinyatakan sebagai berikut :

$$\text{Min} \quad f = \mathbf{c}^T\mathbf{x} \tag{2.13}$$

$$\text{d.s} \quad \mathbf{Ax} \geq \mathbf{b} \tag{2.14}$$

$$\mathbf{x} \geq \mathbf{0}, \mathbf{x} \in \mathbb{Z}^{\boldsymbol{n}} \tag{2.15}$$

Jika nilai dari variabel x adalah 0 atau 1, masalah ini disebut pemrograman bilangan bulat biner atau *binary integer programming* (BIP). Secara matematis, BIP dapat dinyatakan sebagai berikut :

$$\text{Min} \quad f = \mathbf{c}^T\mathbf{x} \tag{2.16}$$

$$\text{d.s} \quad \mathbf{Ax} \geq \mathbf{b} \tag{2.17}$$

$$\mathbf{x} \geq \mathbf{0}, \mathbf{x} \in \{0,1\}^n \tag{1.18}$$

2.4 Masalah Pemrograman Linear Relaksasi

Suatu masalah pemrograman bilangan bulat adalah bagian dari masalah pemrograman linear dengan penambahan kendala variabel bilangan bulat. Terkadang sulit untuk mendapatkan solusi optimal dari masalah pemrograman bilangan bulat yang telah ada. Karena itu untuk mempermudah penyelesaian masalah pemrograman bilangan bulat diperlukan relaksasi pada kendala variabel bilangan bulat. Secara umum, relaksasi suatu masalah pemrograman linear bilangan bulat didapatkan dengan cara menghapus kendala dimana kendala tersebut merupakan bilangan bulat yang membuat masalah sulit untuk diselesaikan (Hillier & Lieberman, 1995). Masalah pemrograman bilangan bulat akan direlaksasi pada kendala yang terbatas pada bilangan bulat digantikan dengan kendala yang terbatas pada bilangan real.

Definisi 2.1 Untuk pemrograman linear bilangan bulat $min\{\mathbf{c}^T\mathbf{x}: \mathbf{x} \in P \cap \mathbb{Z}^n\}$ dengan formulasi $P = \{\mathbf{x} \in \mathbb{R}^n: \mathbf{Ax} \geq \mathbf{b}\}$, pemrograman linear relaksasi adalah program linear $z^{LP} = min\{\mathbf{cx}: \mathbf{x} \in P\}$ (Wolsey, 1998).

Terdapat hubungan antara program linear relaksasi dengan pemrograman bilangan bulat, yaitu:

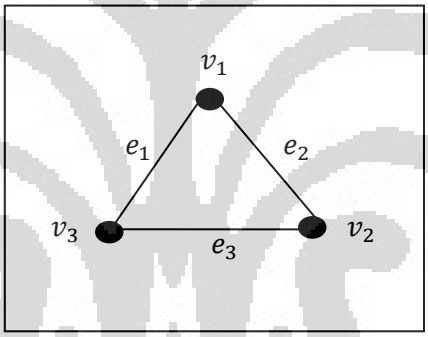
a. Jika solusi program linear relaksasi tidak layak, maka solusi pada pemrograman linear bilangan bulat tidak layak.
b. Misal $\mathbf{x}^*$ adalah solusi optimal dari program linear relaksasi. Jika $\mathbf{x}^* \in P$ dan $f(\mathbf{x}^*) = g(\mathbf{x}^*)$ maka $\mathbf{x}^*$ solusi optimal dari pemrograman linear bilangan bulat dimana f fungsi tujuan pemrograman linear bilangan bulat sedangan g fungsi tujuan program linear relaksasi.

(Wolsey, 1998)

2.5 Teori Graf

Teori graf adalah bagian dari matematika diskrit yang banyak digunakan sebagai alat bantu untuk menggambarkan atau menyatakan suatu masalah agar lebih mudah dimengerti dan diselesaikan.

Definisi 2.2 Suatu graf G terdiri dari pasangan himpunan tak kosong **simpul** dan pasangan tak terurut dari simpul-simpul yang disebut **busur**. Himpunan simpul dari graf G dinotasikan dengan $V(G)$ dan himpunan busur dari graf G dinotasikan dengan $E(G)$. Jika v dan w adalah simpul-simpul dari G, maka suatu busur yang terbentuk dari vw atau wv dikatakan *join* v dan w (Wilson & Watkins, 1990). Contoh penggambaran suatu graf G:

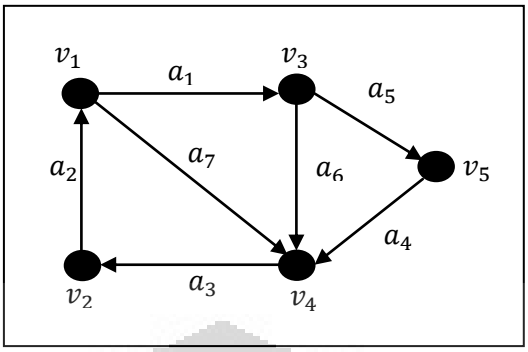


Gambar 2.1 Contoh Graf $G = (V, E)$

Pada Gambar 2.1 menunjukkan penggambaran graf dimana himpunan simpul adalah $V(G) = \{v_1, v_2, v_3\}$ dan himpunan busur adalah $E(G) = \{e_1, e_2, e_3\}$.

Selain graf pada umumnya, tidak jarang suatu masalah menggunakan graf berarah atau digraf untuk memepermudah dalam menyelesaikan masalah tersebut.

Definisi 2.2 Graf berarah atau digraf $G = (V, A)$ adalah himpunan tak kosong simpul V dan himpunan busur berarah A. Busur berarah $a \in A$ dapat direpresentasikan dengan pasangan terurut (u, v) dimana $u, v \in V$. Dengan adanya arah, maka (u, v) tidak sama dengan (v, u). Suatu busur pada graf tidak berarah dipandang sebagai busur berarah dengan dua arah (Hartsfield.,dkk, 2004). Berikut merupakan contoh graf berarah:



Gambar 2.2 Contoh Digraf $G = (V, A)$

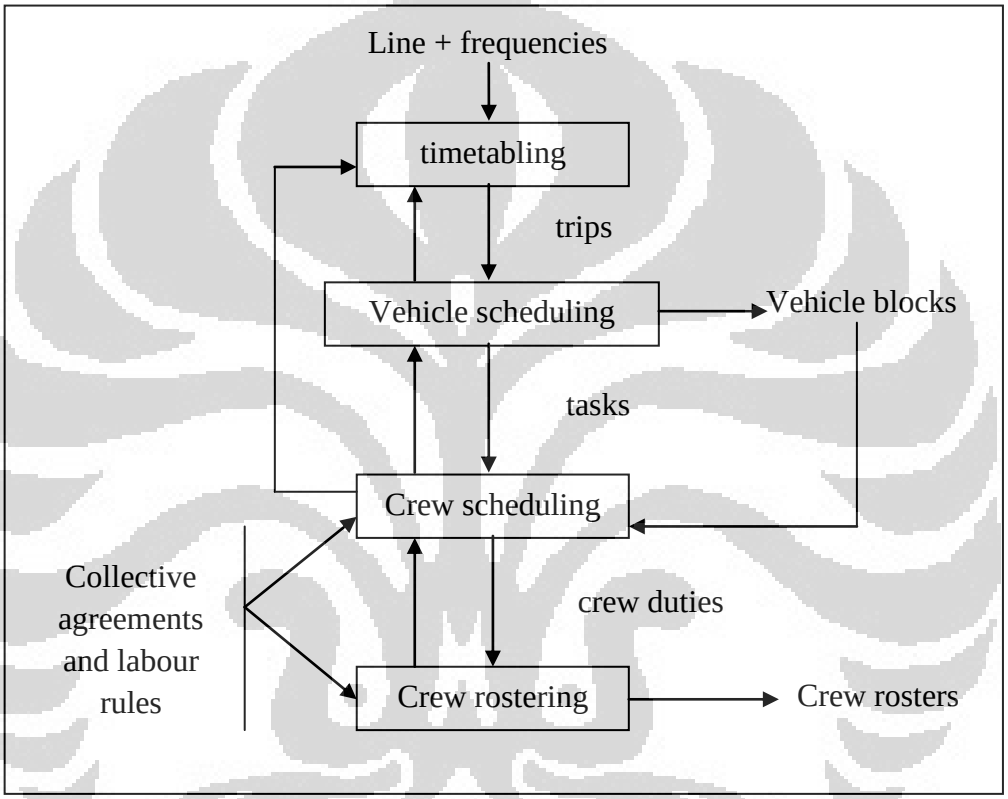
Pada Gambar 2.2 menunjukkan penggambaran digraf dimana himpunan simpul adalah $V(G) = \{v_1, v_2, v_3, v_4, v_5\}$ dan himpunan busur berarah adalah $A(G) = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7\}$. Ketika busur pada digraf G mempunyai nilai atau bobot, maka digraf G disebut sebagai graf berarah dan berbobot.

Graf juga biasa disebut sebagai jaringan (*network*). Menurut Ahuja,.dkk (1993), tidak ada perbedaan antara graf dan jaringan. Istilah jaringan lebih sering digunakan pada masalah penugasan. Penjadwalan kendaraan merupakan bagian dari masalah penugasan, maka pada skripsi ini akan menggunakan istilah jaringan untuk menggambarkan dan membantu dalam penyelesaian masalah penjadwalan kendaraan.

2.6 Dasar-dasar Perencanaan Transportasi untuk Angkutan Umum

Secara umum definisi transportasi adalah pemindahan manusia atau barang dari satu tempat ke tempat lainnya dengan menggunakan sebuah wahana yang digerakkan oleh manusia atau mesin (Nasution, 2004). Transportasi dapat dikatakan sebagai suatu kebutuhan turunan karena transportasi timbul disebabkan adanya maksud atau tujuan yang ingin dicapai melalui transportasi. Misalnya, pengiriman barang, berpergian, bekerja dan lain-lain. Konsep transportasi didasarkan pada adanya perjalanan antara asal dan tujuan. Perjalanan dilakukan melalui suatu lintasan tertentu yang menghubungkan asal dan tujuan, menggunakan alat angkut atau kendaraan.

Angkutan umum merupakan salah satu media transportasi yang digunakan masyarakat secara bersama-sama dengan membayar tarif. Trayek merupakan lintasan angkutan umum untuk pelayanan jasa angkutan orang dengan kendaraan angkutan umum yang mempunyai asal dan tujuan perjalanan tetap, lintasan tetap dan jadwal tetap maupun tidak berjadwal. Adapun proses perencanaan pengoperasian angkutan umum mempunyai beberapa tahap seperti berikut:



Sumber: Huisman, 2004

Gambar 2.3 Proses Perencanaan Pengoperasian Angkutan Umum

Berdasarkan Gambar 2.3, dapat dilihat bahwa proses perencanaan pengoperasian angkutan umum berawal dari penetapan trayek atau rute (*lines*) yang akan dilalui kendaraan dan intensitas kendaraan yang melalui suatu rute (*frequency*) dalam suatu periode waktu. Berdasakan rute dan frekuensi yang sudah ada, maka dapat ditentukan jam keberangkatan kendaraan (*timetable)* pada tahapan selanjutnya. Seandainya terdapat tiga rute, yaitu dari lokasi A ke lokasi B dan dari lokasi A ke lokasi C. Diketahui *timetable* sebagai berikut:

Tabel 2.1 Contoh *Timetable*

Trip	**Lokasi Keberangkatan**	**Lokasi Kedatangan**	**Waktu Keberangkatan**	**Waktu Kedatangan**
1	B	A	07.00	08.00
2	C	A	07.15	08.00
3	A	B	08.05	09.05
4	A	C	08.15	09.00

Setelah diketahui *timetable*, dapat ditentukan himpunan perjalanan (*trips*) dimana *trip* terdiri dari waktu keberangkatan, waktu kedatangan, lokasi awal, dan lokasi akhir. Berdasarkan Tabel 2.1, suatu *trip* dapat dilihat dari baris tabel dan *trip* diurutkan berdasarkan waktu keberangkatan yang meningkat. Contohnya pada *trip* 1 merupakan baris pertama pada Tabel 2.1, dimana baris pertama memuat informasi bahwa suatu kendaraan berjalan dari lokasi B menuju lokasi A dengan waktu awal pukul 07.00 dan waktu akhir 08.00. Jika dilihat Tabel 2.1, terdapat 4 *trip* yang akan dilakukan kendaraan. Penugasan kendaraan dalam menjalani himpunan perjalanan diatur pada tahap penjadwalan kendaraan *(vehicle scheduling).* Jadwal tugas masing-masing kendaraan disebut dengan *vehicle block* digunakan untuk mengatur penugasan pengemudi (*crew scheduling*) berdasarkan aturan-aturan tertentu terkait dengan aturan kerja. Sementara *crew rostering* merupakan penjadwalan awak pengemudi dalam jangka panjang.

2.7 Masalah Penugasan *Quasi* untuk Masalah Penjadwalan Kendaraan

Pada skripsi ini akan dibahas tahap penjadwalan kendaraan pada proses perencanaan pengoperasian angkutan umum. Sebelum membahas lebih lanjut mengenai penjadwalan kendaraan, akan dijelaskan mengenai beberapa istilah yang digunakan pada proses penjadwalan ini, diantaranya *depot*, perjalanan yang bersesuaian atau *compatible trip, time point, deadhead, vehicle block, headway*, dan *interlining*.

Depot merupakan tempat parkir kendaraan ketika sedang tidak digunakan untuk sementara waktu. Waktu maksimal yang diperlukan kendaraan untuk memulai perjalanan dari *depot* ke *trip* awal, dari *trip* ke *trip,* atau dari *trip* akhir ke *depot* disebut ***time point***. Perpindahan kendaraan dari satu *trip* ke *trip* lainnya atau dari *depot* ke *trip* atau dari *trip* ke *depot* tanpa penumpang disebut ***deadhead***. *Trip* i dan *trip* j dikatakan ***compatible trip***, jika *trip* i dan j secara berturut-turut ditugaskan pada kendaraan yang sama, dimana waktu awal *trip* j lebih besar atau sama dengan waktu akhir *trip* i ditambah waktu tempuh dari lokasi akhir *trip* i ke lokasi awal *trip* j. Dengan kata lain, *trip* i dan *trip* j dikatakan *compatible* jika $st_j \geq et_i + trav(i,j)$, dimana et_i adalah waktu kedatangan *trip* i, $trav(i,j)$ adalah waktu melakukan *deadhead* dari lokasi kedatangan *trip* i ke lokasi keberangkatan j, dan st_j adalah waktu keberangkatan *trip* j. Jika dilihat dari Tabel 2.1 pada Subbab 2.6, maka *compatible trip* terjadi pada *trip* 1 dengan *trip* 3 dan *trip* 2 dengan *trip* 4. ***Vehicle block*** merupakan barisan *trip* dan *deadhead* yang dapat dijalankan satu kendaraan dimulai dari *depot* dan berakhir di *depot*. Contohnya pada Tabel 2.1, *vehicle block* pertama adalah *depot* – *trip*1 – *trip* 3 – *depot* dan *vehicle block kedua* adalah *depot – trip* 2 – *trip* 4 – *depot*. ***Headway*** merupakan selang waktu kedatangan dua kendaraan pada suatu lokasi. Jalan yang menghubungkan dari satu lokasi ke lokasi lain disebut rute. Sedangkan pergantian rute yang dilakukan kendaraan angkutan disebut ***interlining***.

2.7.1 Penjadwalan Kendaraan Angkutan Umum Satu *Depot* dengan Waktu Perjalanan Tetap (Statik)

Masalah penjadwalan kendaraan merupakan proses penugasan kendaraan terhadap sejumlah perjalanan (*trips*) yang berasal dari jam keberangkatan (*timetable*) sedemikian sehingga diperoleh total biaya operasional kendaraan yang minimum (Huisman, 2004). Pada umumnya, penjadwalan kendaraan angkutan umum yang dilakukan adalah penjadwalan kendaraan angkutan umum dengan waktu tempuh perjalanan yang tetap (statik). Kondisi ideal dimana penjadwalan kendaraan statik dapat dilakukan adalah kondisi pada jalan yang jarang terjadi

kemacetan lalu lintas dimana keadaan jalan serta intensitas kendaraan yang melalui jalan tersebut menunjukkan angka yang konstan (Huisman, 2004).

Penjadwalan kendaraan jika dilihat dari banyaknya *depot* terbagi menjadi dua macam, yaitu penjadwalan kendaraan dengan satu *depot* (*single depot vehicle scheduling*) dan penjadwalan kendaraan dengan lebih dari satu *depot* (*multiple depot vehicle scheduling*). Dalam skripsi ini penjadwalan kendaraan yang digunakan adalah penjadwalan kendaraan dengan satu *depot*.

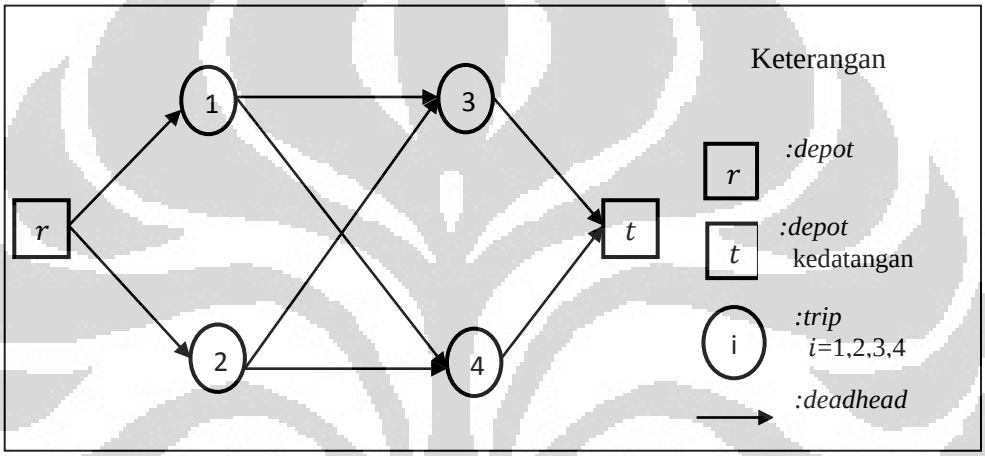
Single depot vehicle scheduling problem (SDVSP) akan mempunyai solusi layak jika setiap *trip* telah ditugaskan ke suatu kendaraan dan setiap kendaraan berangkat dari *depot*, melakukan suatu barisan *trip*, dan berakhir di *depot* yang sama dengan *depot* keberangkatan (Huisman, 2004). Tujuan dari penjadwalan kendaraan adalah menghasilkan suatu jadwal kendaraan yang optimal, yaitu suatu jadwal kendaraan sedemikian sehingga biaya operasional kendaraan minimum. Biaya operasional ini terdiri dari biaya tetap (*fixed cost*) untuk setiap kendaraan dan biaya variabel (*variable cost*) yang berhubungan dengan jarak, waktu tempuh, serta *deadhead*. Selain itu, diasumsikan biaya kendaraan dari *depot* ke setiap *trip* adalah sama.

Pada masalah penjadwalan kendaraan, diasumsikan banyak kendaraan sama dengan banyaknya *trip*. Jika terdapat n *trip*, maka diasumsikan terdapat n kendaraan yang dapat melakukannya. Namun, dalam satu hari suatu kendaraan mungkin melakukan lebih dari satu *trip* sehingga sebanyak n *trip* mungkin dilakuakan oleh m kendaraan, dimana $m < n$. Oleh karena itu, penjadwalan kendaraan juga bertujuan untuk meminimumkan banyaknya kendaraan yang digunakan untuk melakukan seluruh *trip* yang ada sedemikian sehingga meminimumkan biaya operasional.

2.7.2 Representasi Masalah Penjadwalan Kendaraan dalam Jaringan

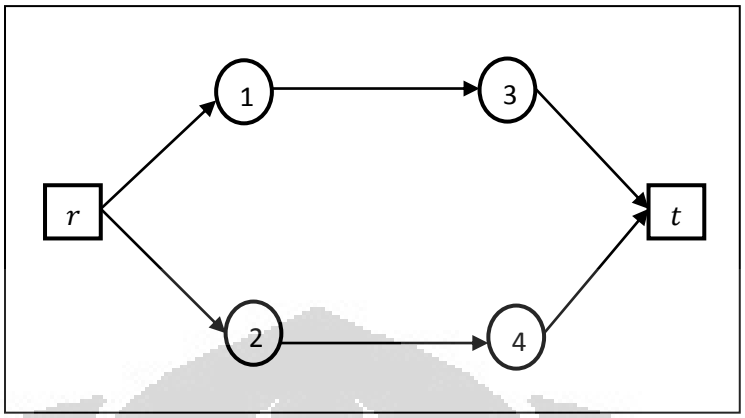
Untuk memudahkan penyelesaian masalah penjadwalan kendaraan, akan dibentuk model matematis dari masalah tersebut. Sebelumnya, representasikan masalah penjadwalan ke dalam suatu jaringan.

Misalkan, N adalah himpunan *trip* dan E adalah himpunan *deadhead* diantara *compatible trip*. Kendaraan harus berangkat dari *depot* awal (*source*) dan kembali ke *depot* akhir (*sink*), dimana *depot* awal dinotasikan dengan r dan *depot* akhir dinotasikan dengan t. Definisikan suatu jaringan $G = (V, A)$ yang merepresentasikan suatu masalah penjadwalan kendaraan, dimana $V = N \cup \{r, t\}$. V adalah himpunan simpul dan $A = E \cup (\{r\} \times N) \cup (N \times \{t\})$ adalah himpunan busur. Berikut contoh masalah penjadwalan kendaraan yang direpresentasikan dalam suatu jaringan:



Gambar 2.4 Contoh Representasi Masalah Penjadwalan Kendaraan dalam Jaringan

Pada Gambar 2.4, terdapat empat *trip* yang akan dibuat penjadwalan kendaraan. Pelabelan masing-masing simpul pada *trip* didasari peningkatan waktu keberangkatan dari *timetable*. Kemudian akan dicari *compatible trip* sehingga didapatkan *vehicle block* yang diinginkan. Misal, *trip* yang *compatible* adalah *trip* 1 dengan *trip* 3 dan *trip* 2 dengan *trip* 4. Maka *vehicle block* yang didapatkan adalah r-1-3-t dan r-2-4-t. Berdasarkan *vehicle blocks* yang didapatkan, penjawalan kendaraan pada Gambar 2.4 dapat dijalankan oleh dua kendaraan. Jika direpresentasikan dalam suatu jaringan, maka hasil penjadwalan kendaraan yang diperoleh sebagai berikut:



Gambar 2.5 Representasi Hasil Penjadwalan Penjadwalan Kendaraan dalam Jaringan

Jika terjadi penukaran rute, sehingga *vehicle blocks* yang didapatkan adalah r-1-4-t dan r-2-3-t, maka peristiwa ini disebut *interlining*.

2.7.3 Model Masalah Penugasan *Quasi*

Model matematis yang digunakan untuk masalah penjadwalan kendaraan pada skripsi ini berbentuk masalah penugasan *quasi* atau *quasi assignment problem* (QAP), yaitu salah satu bentuk khusus dari masalah penugasan atau *assignment problem* (AP).

Masalah penugasan atau *assignment problem* (AP) merupakan salah satu bagian dari masalah LP dimana terdapat sejumlah pekerja yang akan ditugaskan untuk melakukan sejumlah tugas (Hillier dan Lieberman, 1995). Berdasarkan banyaknya pekerja yang ditugaskan dan banyaknya tugas yang dipasangkan, AP dapat dibagi menjadi dua bentuk penugasan, yaitu AP simetrik dan AP asimetrik (Bertsekas, 1992).

Masalah penugasan dikatakan sebagai AP simetrik jika banyaknya pekerja sama dengan banyaknya tugas, misalkan banyaknya pekerja adalah n dan banyaknya tugas adalah n, maka setiap tugas dipasangkan dengan tepat satu pekerja sedemikian sehingga diperoleh biaya minimum. Sedangkan AP asimetrik jika terdapat m pekerja dan n tugas, dengan $m < n$, maka setiap tugas dipasangkan dengan pekerja sedemikian sehingga diperoleh biaya minimum.

Pada umumnya, suatu masalah penugasan dapat direpresentasikan oleh suatu jaringan $G = (V, E)$, dimana V merupakan himpunan simpul dan E merupakan himpunan busur. Sementara busur (i, j) ada di E jika pekerja i dipasangkan dengan tugas j. Misalkan terdapat n pekerja dan n objek, dimana himpunan objek dinotasikan dengan N dan himpunan tugas juga dinotasikan dengan N untuk suatu masalah penugasan *quasi*. Jika jaringan $G = (V, A)$ merepresentasikan masalah penugasan *quasi*, maka terdapat dua simpul tambahan, misal simpul r dan simpul t, dimana r sebagai *source* dan t sebagai *sink*.

Perbedaan masalah penugasan *quasi* dengan masalah penugasan pada umumnya, yaitu selain pekerja dapat dipasangkan dengan tugas, setiap pekerja juga harus dipasangkan dengan *sink* (simpul t pada jaringan $G = (V, A)$) dan setiap tugas harus dipasangkan dengan *source* (simpul r pada jaringan $G = (V, A)$). Oleh sebab itu, terdapat tiga tipe penugasan dalam masalah penugasan *quasi*, yaitu:

a. Penugasan pekerja-tugas, $S = \{(i, j) | (i, j) \in E\}$
b. Penugasan *source*, $D_r = \{j | j \in N, j \text{ berpasangan dengan } source\}$
c. Penugasan *sink*, $D_t = \{i | i \in N, i \text{ berpasangan dengan } sink\}$

Selanjutnya, dirumuskan masalah penugasan *quasi* sebagai suatu masalah meminimumkan total c_{ij} (Freling, R.,dkk, 1999).

Masalah penjadwalan kendaraan merupakan suatu contoh dari masalah penugasan, dimana kendaraan berperan sebagai pekerja dan *trip* berperan sebagai tugas. Namun, keberadaan *depot* juga harus dipertimbangkan karena *depot* akan menjadi tempat keberangkatan awal dan pemberhentian akhir kendaraan, yaitu setiap kendaraan yang akan menjalankan tugas berasal dari *depot* dan kendaraan yang telah selesai menjalankan tugas kembali ke *depot*. Sehingga penjadwalan kendaraan dapat dimodelkan ke dalam bentuk masalah penugasan *quasi* atau QAP.

Langkah awal dalam membentuk model QAP untuk masalah penjadwalan kendaraan satu *depot* atau SDVSP, yaitu menentukan himpunan simpul dan himpunan busur untuk jaringan QAP, dimana himpunan *trip*, $N = \{1,2, \dots, n\}$ akan menjadi bagian dari himpunan simpul dan himpunan *deadhead* antar *trip*, $E = \{(i, j) | i < j;\ i, j\ compatible \in N\}$ akan menjadi bagian dari himpunan

busur. Sementara itu, misalkan simpul r dan t merepresentasikan *depot* yang sama, dimana *r* sebagai *depot* keberangkatan awal (*source*) dan *t* sebagai *depot* pemberhentian akhir (*sink*).

Berikutnya definisikan suatu jaringan *vehicle scheduling*, $G = (V, A)$ dimana himpunan simpul $V = N \cup \{r, t\}$ dan himpunan busur $A = E \cup (\{r\} \times N) \cup (N \times \{t\})$. Suatu lintasan (*path*) dari r ke t dalam jaringan G akan merepresentasikan suatu penjadwalan yang layak untuk satu kendaraan dan suatu penjadwalan kendaraan layak yang lengkap merupakan suatu himpunan *path* yang saling lepas (*disjoint*) dari r ke t sedemikian sehingga setiap simpul dalam N terlewati.

Misalkan c_{ij} adalah biaya kendaraan dari busur$(i, j) \in A$, yaitu biaya pemasangan *depot* dengan *trip* dan biaya pemasangan antara *trip* dengan *trip*, dimana c_{ij} fungsi dari waktu tempuh dan *deadhead* perjalanan. Tujuan dari pemodelan QAP adalah meminimumkan total biaya pengoperasian kendaraan yang digunakan. Variabel keputusan yang digunakan adalah y_{ij}, dimana $y_{ij} = 1$, jika kendaraan menjalankan *trip* j secara langsung setelah menjalankan *trip* i dan $y_{ij} = 0$, untuk lainnya. Bentuk model QAP adalah sebagai berikut:

$$\text{Min} \quad \sum_{(i,j)\epsilon A} c_{ij} y_{ij} \tag{2.19}$$

$$\text{d.s} \quad \sum_{\{i:(i,j)\in A\}} y_{ij} = 1 \qquad \forall j \in N \tag{2.20}$$

$$\sum_{\{j:(i,j)\in A\}} y_{ij} = 1 \qquad \forall i \in N \tag{2.21}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{2.22}$$

Fungsi (2.19) merupakan fungsi tujuan meminimumkan total biaya pengoperasian kendaraan dalam melakukan barisan *trip*. Kendala (2.20) menjamin bahwa suatu *trip* j berasal dari suatu *trip* i atau dari suatu *depot* keberangkatan. Kendala (2.21) menjamin bahwa setiap *trip* i menuju suatu *trip* j atau *depot* akhir. Kendala (2.22) merupakan batasan variabel keputusan y_{ij} dengan nilai 0 atau 1.

2.8 Algoritma *Auction*

Menurut Bertsekas (1992), algoritma *auction* merupakan algoritma primal-dual untuk menyelesaikan masalah penugasan atau *assignment problem* (AP) linear. Prinsip algoritma *auction* berdasarkan pada sistem taruhan (*bid*) pada pelelangan (*auction*). Tujuan yang diinginkan dalam penggunaan algoritma *auction* adalah mendapatkan total keuntungan (*benefit*) semaksimum mungkin dengan memasangkan setiap tugas ke tepat satu pekerja.

Algoritma *auction* memiliki tiga bentuk, yaitu *forward, reverse,* dan kombinasi *forward-reverse*. Jika setiap tugas memiliki harga (*price*) dan setiap pekerja memiliki keuntungan bersih (*profit*), maka dalam *forward auction*, satu pekerja ditugaskan ke suatu tugas dengan total keuntungan (*benefit*) maksimum, dimana *price* tugas akan dinaikkan sebanyak mungkin hingga memenuhi kondisi *complementary slackness*, dimana kondisi ini akan dijelaskan lebih lengkap pada Subbab 2.8.1. Sedangkan pada *reverse auction*, tugas yang belum dipilih oleh pekerja terbaik akan meningkatkan *profit* sebanyak mungkin hingga memenuhi kondisi *complementary slackness*.

2.8.1 Algoritma *Auction* untuk Masalah Penugasan

Misalkan terdapat n tugas dan n pekerja yang harus dipasangkan, dimana setiap tugas harus dikerjakan oleh tepat satu pekerja. Kemudian dimisalkan terdapat suatu *benefit* a_{ij} untuk menugaskan pekerja i terhadap tugas j sehingga tujuan dari pemasangan tugas dan pekerja ini untuk memperoleh total *benefit* yang maksimum. Misalkan, A adalah himpunan semua pasangan pekerja i dan tugas j yang mungkin bisa dipasangkan. Untuk setiap pekerja i, notasikan $P(i)$, yaitu himpunan tugas j yang dapat dipasangkan dengan i:

$$P(i) = \{j|(i,j) \in A\} \tag{2.23}$$

dan untuk setiap tugas j, notasikan $Q(j)$, yaitu himpunan pekerja i yang dapat dipasangkan dengan j:

$$Q(j) = \{i|(i,j) \in A\} \tag{2.24}$$

Misalkan S menyatakan suatu himpunan penugasan dari pasangan pekerja i dan tugas j atau dinyatakan dengan (i, j), sedemikian sehingga setiap pekerja i dan setiap tugas j paling banyak ada satu pasang di S. Jika banyaknya pasangan (i, j) dalam S adalah n, dimana setiap tugas telah ditugaskan ke setiap pekerja yang berbeda, maka S disebut sebagai himpunan tugas yang layak. Himpunan tugas yang layak S dikatakan optimal jika S terdiri dari pasangan pekerja dan tugas $(1, j_1), \dots, (n, j_n)$ dari A sedemikian sehingga tugas $j_1, \dots, j_n$ berbeda dan total *benefit*, $\sum_{i=1}^{n} a_{ij_i}$, bernilai maksimum.

Secara intuitif, algoritma *auction* dapat digambarkan dengan masalah *equilibrium* dalam bidang ekonomi, yakni adanya nilai keseimbangan antara produsen dan konsumen. Pada algoritma *auction*, kondisi *equlibrium* terjadi ketika tidak ada lagi tugas dapat dipasangkan dengan suatu pekerja. Misalkan, tugas j mempunyai harga (*price*), p_j, sehingga keuntungan bersih (*net*) dari tugas j untuk pekerja i adalah $a_{ij} - p_j$ dan setiap pekerja i yang ingin ditugaskan ke suatu tugas j_i dengan nilai taruhan (*bid*) maksimum mempunyai hubungan sebagai berikut:

$$a_{ij_i} - p_{j_i} = \underset{j \in A(i)}{\text{maks}}\{a_{ij} - p_j\} \tag{2.25}$$

Jika kondisi pada persamaan (2.22) dipenuhi, maka kondisi *equilibrium* telah tercapai. Dalam LP, persamaan (2.22) disebut sebagai kondisi *complementary slackness* (CS). Pada awal setiap iterasi, kondisi CS terpenuhi untuk semua pasangan penugasan (i, j_i). Algoritma *auction* akan berhenti dilakukan ketika sudah tidak ada lagi tugas yang dapat dikerjakan oleh pekerja.

Pada umumnya, dalam proses pengerjaan algoritma *auction* terdiri dari dua fase. Misal I merupakan subhimpunan tak kosong dari pekerja yang belum ditugaskan. Terdapat dua fase dalam menjalankan algoritma *auction*, yaitu:

a. Fase taruhan (*bidding*) :

   Setiap $i \in I$ mencari suatu tugas j_i untuk mendapatkan *bid* maksimum, yaitu:

$$j_i = \arg \underset{j \in P(i)}{\text{maks}}\{a_{ij} - p_j\}$$

Hitung nilai kenaikan taruhan (*bidding increment*), yaitu $\mu_i = \beta_i - \gamma_i$, dimana β_i adalah *bid* maksimum dari tugas, yaitu:

$$a_{ij_i} - p_{j_i} = \underset{j \in A(i)}{\text{maks}}\{a_{ij} - p_j\} \tag{2.26}$$

dan γ_i adalah *bid* maksimum kedua dari tugas , yaitu:

$$\gamma_i = \underset{j \in P(i), j \neq j_i}{\text{maks}}\{a_{ij} - p_j\} \tag{2.27}$$

Jika j_i satu-satunya tugas dalam P_i , maka nilai $\gamma_i = -\infty$.

b. Fase penugasan (*assignment*):

Setiap tugas j dipilih sebagai tugas terbaik oleh suatu subhimpunan tak kosong $T(j)$ dari pekerja dalam I, tentukan petaruh (*bidder*) tertinggi, yaitu :

$$i_j = \arg\underset{i \in T(j)}{\text{maks}}\{\beta_i - \gamma_i\} \tag{2.28}$$

dan meningkatkan *price* tugas j sebesar *bidding increment* tertinggi, yaitu $\text{maks}_{i \in T(j)}\{\beta_i - \gamma_i\}$. Kemudian tugas yang terpilih akan dipasangkan dengan petaruh tertinggi i_j. Sehingga terbentuklah himpunan pasangan (i, j) sebagai himpunan *assignment* S.

$\boldsymbol{\varepsilon - Complementary\ Slackness\ (\varepsilon - CS)}$

Misalkan terjadi nilai $\beta_i = \gamma_i$, ini akan mengakibatkan pengulangan iterasi secara terus menerus (*cycling*) pada algoritma *auction* karena sulit untuk memutuskan tugas mana yang dipilih pekerja, dimana *bidding increment* yang bernilai nol. Untuk menghindari *cycling*, digunakan suatu pengganggu mekanisme berdasarkan prinsip pelelangan sebenarnya dimana setiap *bid* untuk suatu tugas harus menaikkan *price* tugas dengan meminimumkan *increment* yang bernilai positif dan *bidder* harus memenangkan tugas yang mereka pilih. Oleh sebab itu, untuk mencegah adanya *cycling* dalam pengerjaan algoritma *auction* pilih suatu bilangan positif ε dan suatu penugasan beserta suatu vektor *price* **p** memenuhi ε *−complementary slackness* $(\varepsilon - CS)$ jika:

$$a_{ij_i} - p_{j_i} \geq \underset{j \in A(i)}{\text{maks}}\{a_{ij} - p_j\} - \varepsilon \tag{2.29}$$

untuk semua pasangan tugas (i, j_i). Selanjutnya, lakukan perubahan pada perhitungan *bidding increment*, yaitu $\mu_i = \beta_i - \gamma_i + \varepsilon$ sehingga nilai minimal *bidding increment* sebesar ε dan $\varepsilon - CS$ terpenuhi.

Teorema 2.4 Suatu penugasan layak yang memenuhi ε *−complementary slackness* dengan suatu vektor *price* lebih kecil dari $n\varepsilon$ merupakan solusi yang optimal.

Teorema 2.5 Misalkan suatu masalah penugasan layak mempunyai *benefit* a_{ij}. Jika $\varepsilon < \frac{1}{n}$, maka algoritma *auction* akan menghasilkan suatu solusi yang optimal dengan sejumlah iterasi yang berhingga.

2.8.2 Algoritma *Auction* untuk Masalah Penugasan *Quasi* dalam Masalah Penjadwalan Kendaraan

Pada masalah penugasan *quasi* atau *quasi assignment problem* (QAP) dalam masalah penjadwalan kendaraan yang berperan sebagai pekerja adalah kendaraan dan yang berperan menjadi tugas adalah *depot* awal, *trip*, dan *depot* akhir. Bentuk algoritma *auction* yang sering digunakan untuk menyelesaikan QAP adalah kombinasi *forward-reverse*. Pada skripsi ini, bentuk *reverse* hanya digunakan untuk menjamin kendaraan yang ditugaskan ke *trip* awal berasal dari *depot* awal. Kemudian, bentuk *forward* digunakan untuk menentukan penugasan kendaraan dari *depot* awal ke *trip* hingga menuju *depot* akhir.

Misal diberikan himpunan *trip*, $N = \{1,2, \dots, n\}$ dan himpunan busur yang saling berhubungan terhadap *deadhead* perjalanan, yaitu $E = \{(i,j) | i < j, i \in N, j \in N\}$. Kemudian, r dan t merepresentasikan *depot* yang sama, dimana r sebagai *depot* keberangkatan awal (*source*) dan *t* sebagai *depot* pemberhentian akhir (*sink*). Masalah penugasan *quasi* dalam penjadwalan kendaraan mempunyai solusi layak jika setiap *trip* ditugaskan *forward* ke *trip* lainnya menuju *sink* (t) dan

setiap *trip* ditugaskan *backward* ke *trip* lainnya menuju *source* (r). Oleh karena itu, terdapat tiga jenis penugasan dalam penjadwalan kendaraan, yaitu:

a. Penugasan antar *trip*, $S = \{(i,j)|(i,j) \in E\}$
b. Penugasan *depot* awal (*source*), $D_r = \{j|j \in N\}$
c. Penugasan *depot* akhir (*sink*),$D_t = \{i|i \in N\}$

Tujuan penyelesaian QAP dengan algoritma *auction* adalah memaksimumkan total *benefit* (a_{ij}) dimana $a_{ij} = -c_{ij}$ dan diasumsikan a_{ij} adalah bilangan bulat, $\forall (i,j) \in A$.

Selanjutnya, dimisalkan π_i adalah *profit* i dan p_j adalah *price* j yang merupakan variabel dual berkaitan dengan kendala (2.17) dan (2.18). Bentuk dual dari QAP adalah sebagai berikut:

$$\text{Min} \quad \sum_{i \in N} \pi_i + \sum_{j \in N} p_j \tag{2.30}$$

$$\text{d.s} \quad \pi_i + p_j \geq a_{ij} \qquad \forall (i,j) \in E \tag{2.31}$$

$$\pi_i \geq a_{it} \qquad \forall i \in N \tag{2.32}$$

$$p_j \geq a_{rj} \qquad \forall i \in N \tag{2.33}$$

Algoritma *auction* akan menyelesaikan bentuk dual dari QAP dengan meminimumkan jumlah *price* dan *profit*.

Definisi 2.6 Suatu penugasan *quasi* Q dan pasangan $(\boldsymbol{\pi}, \mathbf{p})$ memenuhi $\varepsilon - CS$ untuk QAP jika untuk $\varepsilon > 0$:

$$\pi_i + p_j \geq a_{ij} - \varepsilon \qquad \forall (i,j) \in E \tag{2.34}$$

$$\pi_i + p_j = a_{ij} \qquad \forall (i,j) \in S \tag{2.35}$$

$$\pi_i \geq a_{it} \qquad \forall i \in N \tag{2.36}$$

$$\pi_i = a_{it} \qquad \forall i \in D_t \tag{2.37}$$

$$p_j \geq a_{rj} \qquad \forall j \in N \tag{2.38}$$

$$p_j = a_{rj} \qquad \forall j \in D_r \qquad (2.39)$$

2.8.3 Algoritma *Auction Forward/Reverse* untuk Masalah Penjadwalan Kendaraan

Algoritma *auction* mempunyai tiga bentuk, yaitu *forward*, *reverse*, dan kombinasi *forward-reverse*. Pada algoritma *auction forward*, algoritma yang dilakukan menuju *sink*, sedangkan pada algoritma *auction reverse*, algoritma dilakukan menuju *source*. Pada bentuk kombinasi *forward-reverse*, algoritma *auction* dilakukan menuju *sink* kemudian dilanjutkan dengan menuju *source*. Pada masalah penjadwalan kendaraan, tiga bentuk dari algoritma *auction* dapat diterapkan, dimana bentuk kombinasi algoritma *auction forward-reverse* dapat dilakukan secara bergantian.

Dalam menjalankan algoritma *forward* atau *reverse* untuk masalah penjadwalan kendaraan diperlukan beberapa langkah. Sebelumnya, asumsikan nilai *price* j (p_j)dan *profit* i (π_i) adalah nol dan $\varepsilon < \frac{1}{n}$ dimana n adalah banyak *trip*. Misalkan $f_{ij} = a_{ij} - p_j$ merupakan nilai *bid* dari *trip* i untuk *trip* j yang berlaku untuk penugasan *forward*. Sedangkan pada penugasan *reverse*, nilai *bid* yang berlaku dari *trip* j untuk *trip* i, yaitu $f_{ij} = a_{ij} - \pi_i$. Selanjutnya terdapat tiga himpunan kosong yang akan diperbaharui setelah menjalani tugas, yaitu himpunan penugasan antar-*trip* S, himpunan penugasan *depot* awal (*source*) D_r, dan himpunan penugasan *depot* akhir (*sink*) D_t.

Langkah-langkah algoritma *auction forward*

- Langkah 1: Tentukan *trip* $i \in N$ yang belum mempunyai penugasan *forward*.
- Langkah 2: Tentukan *trip* j_i dengan maksimum *bid* $\beta_i = max\{f_{ij}|j: (i,j) \in A\}$.

  Jika *trip* i hanya mempunyai satu busur $(i,j) \in A$ maka $\gamma_i = -\infty$ dan untuk lainnya $\gamma_i = max\{f_{ij}|j: (i,j) \in A, j \neq j_i\}$. Jika $j_i = t$, maka lanjut ke langkah 4.

- Langkah 3: Perbaharui *prices*: $p_{j_i} = p_{j_i} + \beta_i - \gamma_i + \varepsilon = a_{ij_i} - \gamma_i + \varepsilon$ dan

  $\pi_i = a_{ij_i} - p_{j_i}$.

  Perbaharui penugasan: (i, j_i) masukkan ke dalam himpunan S.
- Langkah 4: Perbaharui *prices*: $\pi_i = a_{it}$

  Perbaharui penugasan: masukkan i ke dalam himpunan D_t.

  Kembali ke langkah 1.

Algoritma *auction forward* akan berhenti ketika sudah tidak ada lagi kandidat *trip* atau *sink* yang dapat dijalankan.

Langkah-langkah algoritma *auction reverse*

- Langkah 1: Tentukan *trip* j $\in N$ yang belum mempunyai penugasan *reverse*.
- Langkah 2: Tentukan *trip* j_i dengan maksimum *bid* $\beta_i = max\{f_{ij} | j: (i,j) \in A\}$.

  Jika *trip i* hanya mempunyai satu busur $(i,j) \in A$, maka $\gamma_i = -\infty$ dan untuk lainnya $\gamma_i = max\{f_{ij} | j: (i,j) \in A, j \neq j_i\}$. Jika $j_i = t$, maka lanjut ke langkah 4.
- Langkah 3: Perbaharui *profit*: $\pi_{i_j} = \pi_{i_j} + \beta_i - \gamma_i + \varepsilon = a_{ij_i} - \gamma_i + \varepsilon$ dan

  $p_j = a_{ij_i} - \pi_{i_j}$.

  Perbaharui tugas : (i_j, j) masukkan ke dalam himpunan S.
- Langkah 4: Perbaharui *profit*: $\pi_{i_j} = a_{rj}$ dan

  Perbaharui penugasan: masukkan i ke dalam himpunan D_r.

  Kembali ke langkah 1.

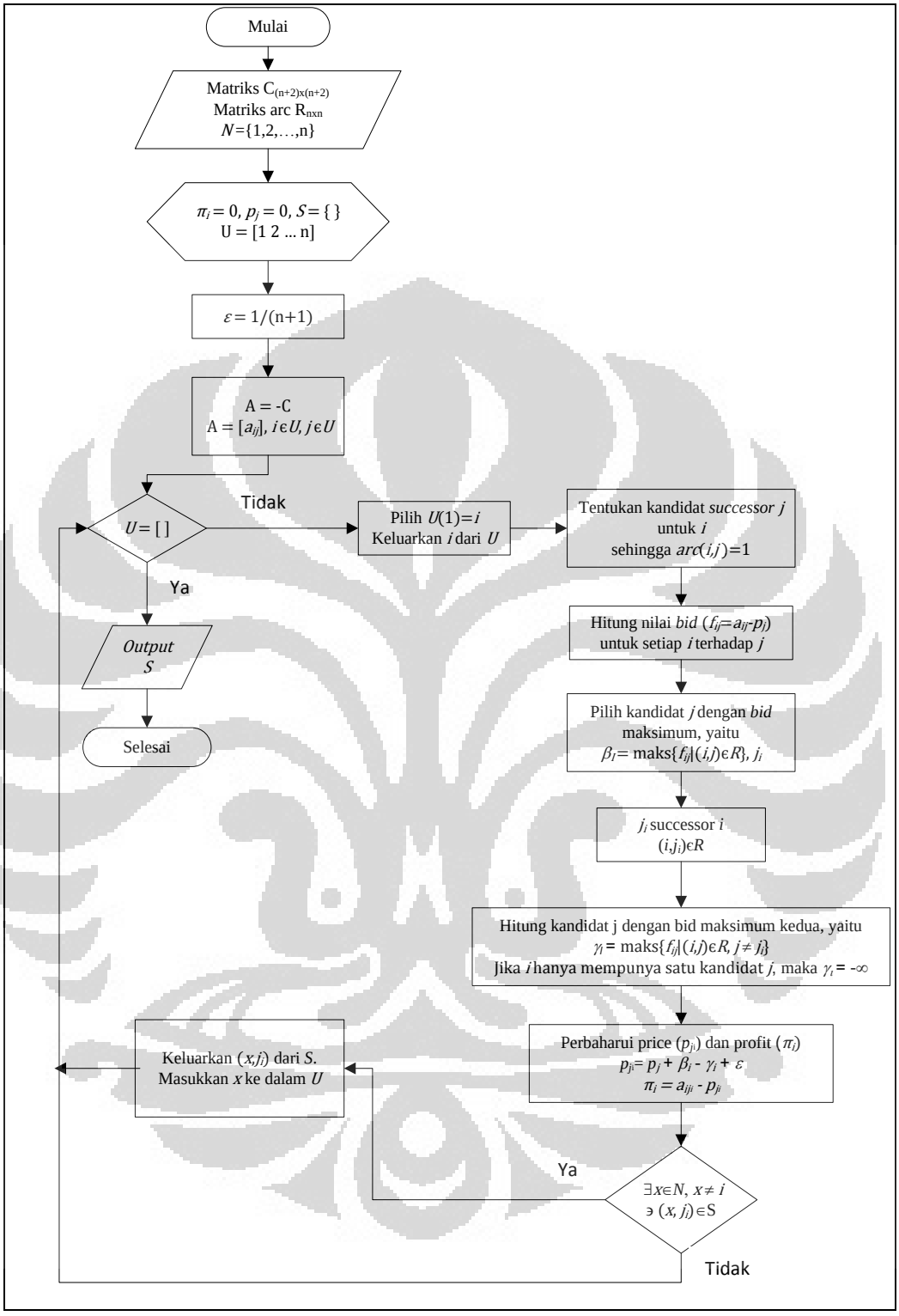
Algoritma *auction reverse* akan berhenti ketika sudah tidak ada lagi kandidat *trip* atau *source* yang dapat dijalankan.

2.8.4 *Flowchart* Algoritma *Auction Forward*

Pada subbab ini akan dibahas mengenai diagram alur (*flowchart*) untuk mempermudah dalam memahami algoritma *auction*. *Flowchart* yang akan dibahas dalam subbab ini adalah *flowchart* algoritma *auction forward*. Sebelum menjalani algoritma *auction*, definisikan dahulu matriks biaya C yang entrinya

adalah c_{ij} dan matriks busur R yang entrinya adalah busur ij yang *compatible*. Kemudian, tentukan kondisi awal dimana $\pi_i = 0$, $p_j = 0$, dan barisan *trip* yang tersedia $U = [1\ 2\ \ldots . n]$, n merupakan banyaknya *trip*.

Pada prosesnya, algoritma *auction* dimulai dengan menghitung nilai ε, dimana $\varepsilon < {}^{1}/_{n}$ dan hitung matriks *benefit* $\mathrm{A} = -\mathrm{C}$ yang entrinya a_{ij}. Selanjutnya, periksa apakah U sudah kosong, jika belum, maka lakukan proses algoritma auction seperti langkah-langkah pada Subbab 2.8.3. Jika U sudah kosong, maka keluarkan himpunan penugasan antar *trip* yang direpresentasikan dengan S. Kemudian proses algoritma *auction forward* selesai. Berikut merupakan penggambaran algoritma *auction forward*:



Gambar 2.6 *Flowchart* Algoritma *Auction Forward* untuk Penjadwalan Kendaraan Angkutan Umum

BAB 3
PENJADWALAN KENDARAAN ANGKUTAN UMUM DINAMIK DAN PENYELESAIAN PENJADWALAN DENGAN PENDEKATAN ALGORITMA *AUCTION*

Pada bab ini dibahas mengenai penjadwalan kendaraan angkutan umum dengan mempertimbangkan waktu perjalanan yang tidak tetap (dinamik) dan kemudian dilanjutkan dengan penerapan algoritma *auction* dalam mendapatkan solusi penjadwalan kendaraan angkutan umum dari model penugasan *quasi*. Pada Subbab 3.1 dijelaskan mengenai penjadwalan kendaraan dinamik atau *dynamic vehicle scheduling* (D-VSP) secara umum. Kemudian, pada Subbab 3.2 dijelaskan mengenai pemodelan D-VSP yang disesuaikan dengan pengoperasian angkutan umum kota. Pada Subbab 3.3 dibahas mengenai penerapan algoritma *auction* dalam menyelesaikan model D-VSP yang telah dijelaskan pada Subbab 3.2 dan ilustrasi masalah sederhana. Pada Subbab 3.4 dijelaskan mengenai perhitungan denda (*penalty*) jika kendaraan mengalami ketelambatan (*delay*) tanpa menambahkan *buffer time* pada saat *peak time*.

3.1 Penjadwalan Kendaraan Angkutan Umum Dinamik

Situasi lalu lintas yang padat terkadang menyebabkan waktu tempuh perjalanan (*travel time*) yang tidak tetap, sehingga dapat mengakibatkan keterlambatan (*delay*) pada waktu keberangkatan kendaraan angkutan umum yang sudah terjadwal sebelumnya. Menurut standar pelayanan transportasi umum, *delay* pada waktu keberangkatan dapat menyebabkan suatu perusahaan angkutan umum terkena denda (*penalty*). Semakin banyak *delay* yang terjadi, semakin banyak *penalty* yang dikeluarkan. *Penalty* yang berlaku didasari oleh waktu tunggu penumpang. Jika kendaraan mengalami *delay* pada waktu keberangkatan, maka penumpang akan menunggu lebih lama dari waktu keberangkatan yang telah ada. Karena itu, *delay* pada waktu keberangkatan dapat mengakibatkan kerugian untuk penumpang maupun perusahaan angkutan umum.

Diperlukan penjadwalan kendaraan angkutan umum yang sesuai untuk mengatasi waktu tempuh perjalanan yang tidak tetap. Penjadwalan ini disebut dengan penjadwalan kendaraan dinamik atau *dynamic vehicle scheduling problem* (D-VSP). Tujuan D-VSP adalah meminimumkan biaya pengoperasian kendaraan angkutan umum dengan mempertimbangkan waktu perjalanan yang tidak tetap (Huisman, 2004). Sebelum membuat D-VSP harus diketahui penyelesaian masalah penjadwalan kendaraan statik atau *static vehicle scheduling problem* (S-VSP) dari suatu masalah penjadwalan angkutan umum karena dengan adanya solusi dari S-VSP dapat diketahui barisan *trip* yang dilakukan kendaraan dan banyaknya kendaraan yang digunakan pada waktu perjalanan tetap (statik). Dengan adanya solusi S-VSP dapat diketahui apakah tambahan kendaraan diperlukan dalam melakukan barisan perjalanan dengan mempertimbangkan keadaan dinamik pada penjadwalan kendaraan. Selain itu, perlu diketahui persentasi kenterlambatan perjalanan kendaraan yang digunakan untuk menghitung biaya pengoperasian kendaraan pada keadaan dinamik.

Keadaan yang dinamik akan menyebabkan *delay* pada waktu keberangkatan kendaraan angkutan umum dimana *delay* biasanya terjadi pada waktu ramai (*peak time*). Oleh sebab itu, pada D-VSP akan dilakukan penambahan waktu penyangga (*buffer time*) yang merupakan waktu toleransi antar *trip* saat *peak time*. Dalam penyelesaian D-VSP diperlukan beberapa hal yang harus dilakukan. Pertama, tentukan *time point* T dimana *time point* T menunjukkan *peak time*. Kemudian, tambahkan *buffer time* l pada pada *time point* T dan dilanjutkan dengan membentuk periode waktu $[T, T + l)$, dimana periode ini menunjukkan periode waktu yang mungkin terjadi *delay*. Kedatangan kendaraan angkutan tidak boleh sama dengan atau lebih dari *time point* $T + l$ karena akan dikenakan *penalty* yang lebih besar dan ditanggung oleh perusahaan.

Selama periode $[T, T + l)$ akan dibuat beberapa alternatif keputusan. Beberapa alternatif keputusan ini disebut dengan skenario. Dalam skripsi ini, dibuat tiga skenario yang mungkin terjadi pada periode $[T, T + l)$. Skenario pertama adalah kendaraan tetap menjalankan barisan *trip* yang sesuai dengan solusi S-VSP. Skenario kedua adalah kendaraan bertukar tugas dalam menjalani

barisan *trip* tanpa ada penambahan kendaraan. Skenario ketiga adalah penambahan kendaraan dalam menjalani barisan *trip*. Berdasarkan data dan pengalaman yang terjadi, waktu tempuh saat *peak time* cenderung berubah-ubah dan kemungkinan kendaraan untuk menjalankan *trip* yang seharusnya sangat kecil, sehingga probabilitas untuk skenario pertama (p^{s_1}) sangat kecil. Probabilitas untuk skenario kedua (p^{s_2}) lebih besar daripada probabilitas skenario pertama karena kemungkinan kendaraan untuk bertukar tugas dengan kendaraan lainnya lebih besar daripada tetap menjalankan barisan *trip* yang seharusnya. Probabilitas untuk skenario ketiga (p^{s_3}) paling besar karena kemungkinan penambahan kendaraan pada saat *peak time* sangat besar.

Setelah menentukan periode *peak time,* D-VSP dapat direpresentasikan dalam bentuk jaringan. Jika pada S-VSP simpul menunjukkan *trip*, maka pada D-VSP simpul menunjukkan lokasi keberangkatan beserta waktu keberangkatan dari suatu *trip* atau lokasi kedatangan beserta waktu kedatangan dari suatu *trip*. Sehingga dari satu simpul yang menunjukkan suatu *trip* dibagi menjadi dua simpul. Perbedaan penggambaran jaringan antara S-VSP dengan D-VSP akan dijelaskan pada Subbab 3.3.2.

3.2 Model Matematis Penjadwalan Kendaraan Angkutan Umum Dinamik

Untuk menyelesaikan masalah penjadwalan yang mempertimbangkan keadaan dinamik diperlukan suatu model yang dapat menyelesaikan D-VSP. Pemodelan masalah yang ingin diselesaikan pada dan beberapa waktu setelah *time point* T, dimana jadwal untuk periode $[T, T + l)$ dan skenario berlaku pada dan setelah *time point* $T + l$. Berikut merupakan pendefinisian variabel yang digunakan dalam pemodelan D-VSP:

N : himpunan *trip* dimana $N = \{1,2, \dots, n\}$.

E : himpunan *deadhead* dimana $E = \{(i,j) | \, i < j, i \in N, j \in N\}$.

r : *depot* awal keberangkatan (*source*).

t : *depot* akhir keberangkaan (*sink*).

S : himpunan skenario.

Jika $|S| = 1$, maka penjadwalan hanya menggunakan skenario tunggal.

Setiap skenario adalah $s \in S$.

A : himpunan busur diantara dua *trip*, dari r menuju *trip*, dan dari *trip* menuju t.

A^* : himpunan busur tanpa busur panjang (kembali ke *depot*).

A_1 : subhimpunan dari A^* pada waktu tidak ramai (*peak off time*).

A_2 : subhimpunan dari A^* waktu ramai (*peak time*).

p^s : probabilitas skenario s yang terjadi.

B^s : banyaknya kendaraan yang digunakan pada skenario s.

c'_{ij} : biaya pemasangan *depot* dengan *trip* dan *trip* dengan *trip* dalam periode *peak off time.*

c^s_{ij} : biaya pemasangan *depot* dengan *trip* dan *trip* dengan *trip* dalam periode *peaktime* pada skenario s.

c : biaya tetap pengoperasian kendaraan.

Variabel keputusan yang digunakan adalah z_{ij} dan y^s_{ij}, dimana z_{ij} menunjukkan *depot-trip* dan *trip-trip* yang dapat dipasangkan pada periode *peak off time* yang merupakan kondisi statik dan y^s_{ij} menunjukkan *depot-trip* dan *trip-trip* yang dapat dipasangkan pada periode *peak time* yang merupakan kondisi dinamik untuk skenario s. Untuk pembatasan penggunaan variabel keputusan, didefinisikan:

$$z_{ij} = \begin{cases} 1, & \text{jika } \textit{trip } j \text{ dilakukan tepat setelah melaukan } \textit{trip } i \text{ pada kondisi statik} \\ 0, & \text{lainnya} \end{cases}$$

$$y^s_{ij} = \begin{cases} 1, & \text{jika } \textit{trip } j \text{ dilakukan tepat setelah melakukan } \textit{trip } i \text{ pada akondisi dinamik untuk skenario } s \\ 0, & \text{lainnya} \end{cases}$$

Tujuan dari pemodelan masalah penjadwalan angkutan umum dinamik adalah meminimumkan biaya pengoperasian kendaraan angkutan umum dengan mempertimbangakan waktu perjalanan yang tidak tetap. Model yang bersesuaian dengan masalah yang ada adalah sebagai berikut:

$$\text{Min} \quad \sum_{(i,j)\in A_1} c'_{ij} z_{ij} + \sum_{(i,j)\in A_2} c^s_{ij} y^s_{ij} \qquad (3.1)$$

d.s $$\sum_{\{i:(i,j)\in A_1\}} z_{ij} + \sum_{\{i:(i,j)\in A_2\}} y_{ij}^s = 1 \qquad \forall s \in S, \forall j \in N \qquad (3.2)$$

$$\sum_{\{j:(i,j)\in A_1\}} z_{ij} + \sum_{\{j:(i,j)\in A_2\}} y_{ij}^s = 1 \qquad \forall s \in S, \forall i \in N \qquad (3.3)$$

$$z_{ij} \in \{0,1\} \qquad \forall (i,j) \in A_1 \qquad (3.4)$$

$$y_{ij}^s \in \{0,1\} \qquad \forall (i,j) \in A_2, \forall s \in S \qquad (3.5)$$

Persamaan (3.1) merupakan fungsi tujuan dari masalah penjadwalan kendaraan angkutan umum dinamik, yaitu meminimumkan biaya pengoperasian kendaraan per hari. Kendala (3.2) menjamin bahwa *trip* i menuju *trip* lainnya, yakni *trip* j atau menuju *depot* akhir. Kendala (3.3) menjamin bahwa setiap *trip* j berasal dari *trip* lainnya, yakni *trip* i atau berasal dari *depot* awal.

Untuk menyelesaikan permasalahan penjadwalan kendaraan dengan algoritma *auction*, dibutuhkan bentuk dual dari model yang telah didapatkan. Dengan memisalkan π_i (*profit*) yaitu variabel dual yang terkait dengan kendala (3.2), p_j (*price*) variabel dual yang terkait dengan kendala (3.3), serta $a_{ij} = -c_{ij}$, maka bentuk dual dari model masalah ini adalah sebagai berikut:

Min $$\sum_{\{i:(i,j)\in A\}} \pi_i + \sum_{\{j:(i,j)\in A\}} p_j \qquad (3.6)$$

d.s $$\pi_i + p_j \geq a_{ij} \qquad \forall i \in E, \forall j \in E \qquad (3.7)$$

$$\pi_i \geq a_{it} \qquad \forall i \in N \qquad (3.8)$$

$$p_j \geq a_{rj} \qquad \forall i \in N \qquad (3.9)$$

Selain biaya pengoperasian kendaraan, diperlukan biaya tambahan pada D-VSP untuk mempersiapkan jika kendaraan mengalami *delay* dalam waktu perjalanan. Tambahan biaya ini melibatkan probabilitas skenario yang telah ditentukan sebelumnya. Rumusan biaya tambahan ini adalah biaya tetap dikali dengan probabilitas skenario s dikali dengan banyaknya kendaraan yang digunakan pada skenario tersebut. Sehingga, persamaan untuk biaya tambahan pada skenario s sebagai berikut:

$$k^s = c.p^s.B^s \qquad (3.10)$$

Secara lengkap rumusan total biaya operasional kendaraan angkutan umum dengan D-VSP, yaitu biaya tetap ditambah dengan biaya tambahan ditambah

dengan biaya pengoperasian kendaraan. Sehingga, persamaannya sebagai berikut:

Total biaya $= c + k^s + \sum_{(i,j)\in A_1} c'_{ij} z_{ij} + \sum_{(i,j)\in A_2} c^s_{ij} y^s_{ij}$ (3.11)

3.3 Penerapan Algoritma *Auction* pada Masalah Penjadwalan Kendaraan Angkutan Umum Dinamik

Pada Subbab 3.1 dan 3.2 sudah dijelaskan mengenai D-VSP dan model yang dapat digunakan untuk D-VSP. Setelah diketahui model penjadwalan kendaraan angkutan yang sesuai dengan keadaan dinamik, maka diperlukan suatu algoritma untuk menyelesaikan model D-VSP. Algoritma yang digunakan untuk menyelesaikan model D-VSP adalah algoritma *auction*. Pengerjaan algoritma *auction* pada model D-VSP sama seperti algoritma *auction* untuk QAP yang telah dijelaskan bada Subbab 2.9.

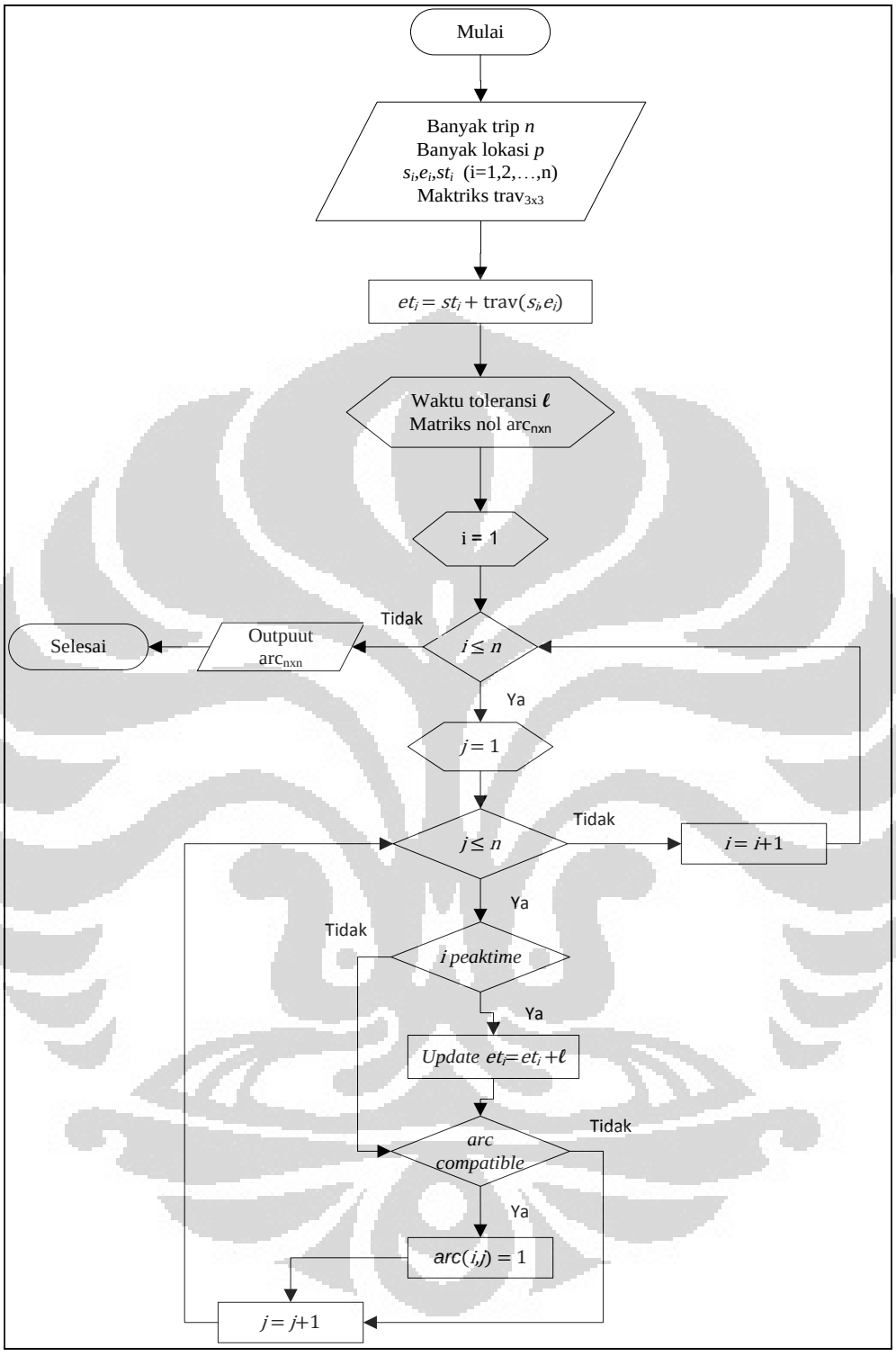
3.3.1 *Flowchart* Pemilihan *Trip* yang *Compatible* pada D-VSP

Perbedaan antara S-VSP dengan D-VSP, yakni pemiliihan *trip* yang *compatible*. Pada S-VSP pemilihan *trip* yang *compatible* berdasarkan kenaikan waktu keberangkatan tanpa mempertimbangkan keadaan apapun. Sedangkan pada D-VSP diperlukan pertimbangan keadaan dinamik dengan menambahkan *buffer time* pada waktu perjalanan kendaraan. Untuk memudahkan dalam memahami pemilihan *trip* yang *compatible* pada D-VSP, akan dibuat *flowchart* seperti pada Gambar 3.1.

Dalam pemilihan *trip* yang *compatible* pada D-VSP harus diketahui terlebih dahulu banyaknya *trip* pada masalah penjadwalan kendaraan, misalnya terdapat n *trip*. Kemudian diketahui juga banyaknya lokasi, yaitu p lokasi disertai dengan informasi lokasi awal (s_i), lokasi akhir akhir (e_i), dan waktu awal *trip* (st_i), dimana $i = 1,2, \dots n$. Selanjutnya, diketahui waktu tempuh dari satu lokasi kelokasi lain yang direpresentasikan dengan matriks trav_{pxp}. Langkah pertama adalah hitung waktu akhir perjalanan (et_i), yaitu $et_i = st_i +$

trav(s_i, e_i). Kemudian masukkan *buffer time* l dan matriks nol sebagai representasi awal *trip* yang *compatible*, yaitu matriks arc_{nxn}. Langkah berikutnya adalah pilih *trip* $i = 1$. Jika $i > n$, maka keluarkan matriks arc_{nxn}. Jika lainnya, maka lanjutkan proses pemilihan *trip* yang *compatible*, yaitu pilih *trip* lain yang akan dipasangkan dengan *trip* i, misal *trip* j. Periksa apakah *trip* i berada pada waktu padat (*peak time*). Jika iya, perbaharui waktu akhir perjalanan, yaitu waktu akhir telah dihitung sebelumnya ditambah dengan *buffer time* l ($et_i = et_i + l$). Kemudian, periksa apakah *trip compatible* ($s_J \geq et_i + trav(i,j)$). Jika tidak, pilih *trip* j lain. Jika ya, maka *trip* i dengan *trip* j *compatible* dan perbaharui matriks arc_{nxn}. Jalankan kembali proses ini hingga *trip* habis dan keluarkan matriks arc_{nxn}.

Agar lebih mudah memahami proses pemilihan *trip* yang *compatible* pada D-VSP, akan digambarkan *flowchart* sebagai berikut:



Gambar 3.1 *Flowchart* Pemilihan *Trip* yang *Compatible* pada Penjadwalan Kendaraan Angkutan Umum Dinamik

3.3.2 Masalah Penjadwalan Kendaraan Angkutan Umum Kota

Pada Subbab ini akan dijelaskan mengenai penggambaran masalah D-VSP. Diberikan data perjalanan kendaraan angkutan umum sebagai berikut:

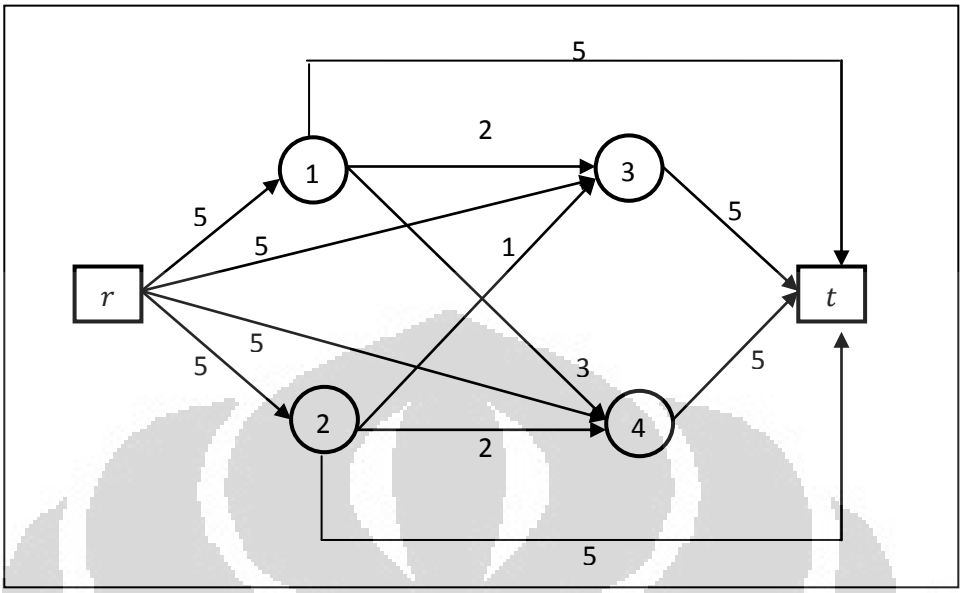
Tabel 3.1 Data Perjalanan

Trip	**Lokasi Keberangkatan**	**Lokasi Kedatangan**	**Waktu Keberangkatan**	**Waktu Kedatangan**
1	B	A	07.00	08.00
2	C	A	07.15	08.00
3	A	B	08.05	09.05
4	A	C	08.15	09.00

Berdasarkan data perjalanan pada Tabel 3.1, diketahui bahwa terdapat empat *trip* dan tiga lokasi tujuan. Diasumsikan hanya terdapat satu *depot* sebagai tempat keberangkatan awal maupun tempat pemberhentian akhir kendaraan. Biaya dari *depot* awal ke setiap *trip* dan dari setiap *trip* ke *depot* akhir diasumsikan sama, yaitu 5 satuan biaya ($c_{r1} = c_{r2} = c_{r3} = c_{r4} = c_{1r} = c_{2r} = c_{3r} = c_{4r} = 5$) . Biaya tetap adalah 100 satuan biaya ($c = 100$). Biaya yang diperlukan dari satu *trip* ke *trip* lainnya sebagai berikut:

- Dari *trip* 1 ke *trip* 3 adalah 2 satuan biaya ($c_{13} = 2$).
- Dari *trip* 1 ke *trip* 4 adalah 3 satuan biaya ($c_{14} = 3$).
- Dari *trip* 2 ke *trip* 3 adalah 1 satuan biaya ($c_{13} = 1$).
- Dari *trip* 2 ke *trip* 4 adalah 2 satuan biaya ($c_{13} = 2$).

Jika direpresentasikan dalam jaringan, maka jaringan masalah penjadwalan kendaraan untuk data perjalanan pada Tabel 3.1 sebagai berikut:



Gambar 3.2 Jaringan Penjadwalan Kendaraan Angkutan Umum

Berdasarkan data perjalanan pada Tabel 3.1 maka akan diperoleh model matematika sebagai berikut:

- Fungsi tujuan

$$\text{Min} \quad (c'_{r1}z_{r1} + c'_{r2}z_{r2} + c'_{13}z_{13} + c'_{14}z_{14} + c'_{23}z_{23} + c'_{24}z_{24} + c'_{3t}z_{3t} + c'_{4t}z_{4t}) + \left(c_{r1}^{s_1}y_{r1}^{s_1} + c_{r2}^{s_1}y_{r2}^{s_1} + c_{13}^{s_1}y_{13}^{s_1} + c_{14}^{s_1}y_{14}^{s_1} + c_{23}^{s_1}y_{23}^{s_1} + c_{24}^{s_1}y_{24}^{s_1} + c_{3t}^{s_1}y_{3t}^{s_1} + c_{4t}^{s_1}y_{4t}^{s_1} + c_{r1}^{s_2}y_{r1}^{s_2} + c_{r2}^{s_2}y_{r2}^{s_2} + c_{13}^{s_2}y_{13}^{s_2} + c_{14}^{s_2}y_{14}^{s_2} + c_{23}^{s_2}y_{23}^{s_2} + c_{24}^{s_2}y_{24}^{s_2} + c_{3t}^{s_2}y_{3t}^{s_2} + c_{4t}^{s_2}y_{4t}^{s_2} + c_{r1}^{s_3}y_{r1}^{s_3} + c_{r2}^{s_3}y_{r2}^{s_3} + c_{13}^{s_3}y_{13}^{s_3} + c_{14}^{s_3}y_{14}^{s_3} + c_{23}^{s_3}y_{23}^{s_3} + c_{24}^{s_3}y_{24}^{s_3} + c_{3t}^{s_3}y_{3t}^{s_3} + c_{4t}^{s_3}y_{4t}^{s_3}\right)$$

- Fungsi kendala

  Penjabaran untuk kendala (3.2) yang menunjukkan bahwa setiap kendaraan pada *trip* i dapat ditugaskan ke tepat satu *trip* j :

  Untuk $j = 1$ dan s_1

$$z_{r1} + z_{11} + \cdots + z_{t1} + y_{r1}^{s_1} + y_{11}^{s_1} + \cdots + y_{t1}^{s_1} = 1$$

  Untuk $j = 2$ dan s_1

$$z_{r2} + z_{12} + \cdots + z_{t2} + y_{r2}^{s_1} + y_{12}^{s_1} + \cdots + y_{t2}^{s_1} = 1$$

  Untuk $j = 3$ dan s_1

$$z_{r3} + z_{13} + \cdots + z_{t3} + y_{r3}^{s_1} + y_{13}^{s_1} + \cdots + y_{t3}^{s_1} = 1$$

Untuk $j = 4$ dan s_1

$$z_{r4} + z_{14} + \cdots + z_{t4} + y_{r4}^{s_1} + y_{14}^{s_1} + \cdots + y_{t4}^{s_1} = 1$$

Untuk $j = 1$ dan s_2

$$z_{r1} + z_{11} + \cdots + z_{t1} + y_{r1}^{s_2} + y_{11}^{s_2} + \cdots + y_{t1}^{s_2} = 1$$

Untuk $j = 2$ dan s_2

$$z_{r2} + z_{12} + \cdots + z_{t2} + y_{r2}^{s_2} + y_{12}^{s_2} + \cdots + y_{t2}^{s_2} = 1$$

Untuk $j = 3$ dan s_2

$$z_{r3} + z_{13} + \cdots + z_{t3} + y_{r3}^{s_2} + y_{13}^{s_2} + \cdots + y_{t3}^{s_2} = 1$$

Untuk $j = 4$ dan s_2

$$z_{r4} + z_{14} + \cdots + z_{t4} + y_{r4}^{s_2} + y_{14}^{s_2} + \cdots + y_{t4}^{s_2} = 1$$

Untuk $j = 1$ dan s_3

$$z_{r1} + z_{11} + \cdots + z_{t1} + y_{r1}^{s_3} + y_{11}^{s_3} + \cdots + y_{t1}^{s_3} = 1$$

Untuk $j = 2$ dan s_3

$$z_{r2} + z_{12} + \cdots + z_{t2} + y_{r2}^{s_3} + y_{12}^{s_3} + \cdots + y_{t2}^{s_3} = 1$$

Untuk $j = 3$ dan s_3

$$z_{r3} + z_{13} + \cdots + z_{t3} + y_{r3}^{s_3} + y_{13}^{s_3} + \cdots + y_{t3}^{s_3} = 1$$

Untuk $j = 4$ dan s_3

$$z_{r4} + z_{14} + \cdots + z_{t4} + y_{r4}^{s_3} + y_{14}^{s_3} + \cdots + y_{t4}^{s_3} = 1$$

Penjabaran untuk kendala (3.3) yang menunjukkan bahwa setiap kendaraan pada *trip* j dijamin berasal dari *trip* i :

Untuk $i = 1$ dan s_1

$$z_{1r} + z_{11} + \cdots + z_{1t} + y_{1r}^{s_1} + y_{11}^{s_1} + \cdots + y_{1t}^{s_1} = 1$$

Untuk $i = 2$ dan s_1

$$z_{2r} + z_{21} + \cdots + z_{2t} + y_{2r}^{s_1} + y_{21}^{s_1} + \cdots + y_{2t}^{s_1} = 1$$

Untuk $i = 3$ dan s_1

$$z_{3r} + z_{31} + \cdots + z_{3t} + y_{3r}^{s_1} + y_{31}^{s_1} + \cdots + y_{3t}^{s_1} = 1$$

Untuk $i = 4$ dan s_1

$$z_{4r} + z_{41} + \cdots + z_{4t} + y_{4r}^{s_1} + y_{41}^{s_1} + \cdots + y_{4t}^{s_1} = 1$$

Untuk $i = 1$ dan s_2

$$z_{1r} + z_{11} + \cdots + z_{1t} + y_{1r}^{s_2} + y_{11}^{s_2} + \cdots + y_{1t}^{s_2} = 1$$

Untuk $i = 2$ dan s_2

$$z_{2r} + z_{21} + \cdots + z_{2t} + y_{2r}^{s_2} + y_{21}^{s_2} + \cdots + y_{2t}^{s_2} = 1$$

Untuk $i = 3$ dan s_2

$$z_{3r} + z_{31} + \cdots + z_{3t} + y_{3r}^{s_2} + y_{31}^{s_2} + \cdots + y_{3t}^{s_2} = 1$$

Untuk $i = 4$ dan s_2

$$z_{4r} + z_{41} + \cdots + z_{4t} + y_{4r}^{s_2} + y_{41}^{s_2} + \cdots + y_{4t}^{s_2} = 1$$

Untuk $i = 1$ dan s_3

$$z_{1r} + z_{11} + \cdots + z_{1t} + y_{1r}^{s_3} + y_{11}^{s_3} + \cdots + y_{1t}^{s_3} = 1$$

Untuk $i = 2$ dan s_3

$$z_{2r} + z_{21} + \cdots + z_{2t} + y_{2r}^{s_3} + y_{21}^{s_3} + \cdots + y_{2t}^{s_3} = 1$$

Untuk $i = 3$ dan s_3

$$z_{3r} + z_{31} + \cdots + z_{3t} + y_{3r}^{s_3} + y_{31}^{s_3} + \cdots + y_{2t}^{s_3} = 1$$

Untuk $i = 4$ dan s_3

$$z_{4r} + z_{41} + \cdots + z_{4t} + y_{4r}^{s_3} + y_{41}^{s_3} + \cdots + y_{4t}^{s_3} = 1$$

Untuk menyelesaikan ilustrasi masalah yang ada dengan algoritma *auction*, ubah bentuk masalah ke dalam bentuk dual seperti pada Subbab 3.2. Bentuk dual dari masalah primal pada ilustrasi di atas adalah sebagai berikut:

Min $\pi_r + \pi_1 + \pi_2 + \pi_3 + \pi_4 + p_1 + p_1 + p_1 + p_1 + p_1$

d.s

$$\pi_1 + p_3 \geq a_{13}$$

$$\pi_1 + p_4 \geq a_{14}$$

$$\pi_2 + p_3 \geq a_{23}$$

$$\pi_2 + p_4 \geq a_{24}$$

$$\pi_3 \geq a_{3t}$$

$$\pi_4 \geq a_{4t}$$

$$p_1 \geq a_{r1}$$

$$p_2 \geq a_{r2}$$

Selanjutnya, selesaikan masalah dual dengan algoritma *auction*.

Masalah penjadwalan kendaraan pada data perjalanan Tabel 2.1 pada awalnya diselesaikan dengan algoritma *auction* dengan model penjadwalan kendaraan statik. Pengerjaan algoritma *auction* pada model S-VSP sama seperti

pengerjaan pada algoritma *auction* untuk QAP yang telah dijelaskan pada Subbab 2.9. Diasumsikan semua nilai *price* awal untuk setiap *trip* adalah nol dan semua *profit* awal untuk setiap kendaraan adalah nol. Kemudian, diasumsikan juga bahwa banyaknya kendaraan sama dengan banyaknya *trip*. Berikut merupakan langkah-langkah algoritma *auction* untuk S-VSP:

- Penugasan *forward*

Iterasi 1 :

Pilih $i = 1$, artinya pilih *trip* 1. Kandidat *trip* untuk *trip* 1 adalah *trip* 3 dan *trip* 4. Kemudian hitung *bid trip* 1 untuk *trip* 3 yang direpresentasikan dengan busur (1,3) yaitu $f_{13} = a_{13} - p_3 = -2 - 0 = -2$ dan *bid trip* 1 untuk *trip* 4 yang direpresentasikan dengan busur (1,4) , yaitu $f_{14} = a_{14} - p_4 = -3 - 0 = -3$.

Untuk menghitung *bidding increment* diperluan nilai β_1 yang merupakan nilai maksimum dari semua *bid* kandidat *trip* yang terpilih, yaitu $\beta_1 = max\{f_{13}, f_{14}\} = -2$ dan γ_1 yang merupakan *bid* maksimum kedua dari *bid* kandidat yang terpilih, yaitu $\gamma_1 = -3$. Berdasarkan perhitungan β_1, dapat diketahui *trip* mana yang dipilih oleh *trip* 1, yaitu *trip* 3 dengan *bidding increment* $\mu_1 = \beta_1 - \gamma_1 + \varepsilon = -2 + 3 + \frac{1}{6} = 1\frac{1}{6}$.

Perbaharui *prices*:

Setelah diketahui bahwa kandidat *trip* 3 yang terpilih, perbaharui nilai *price* dimana *price trip* 1 untuk *trip* 3 terbaru sama dengan *price* awal diambah dengan *bidding increment*, yaitu $p_3 = 1\frac{1}{6}$. Setelah mengetahui *price* yang terbaru, dapat dihitung *profit trip* 3 untuk kendaraan 1, yaitu $\pi_1 = a_{13} - p_3 = -2 - 1\frac{1}{6} = -3\frac{1}{6}$

Perbaharui penugasan:

Masukkan penugasan *trip* 1 dengan *trip* 3 ke dalam himpunan penugasan *trip*, yaitu $S(Q) = \{(1,3)\}$. *Trip* yang belum dijalankan adalah *trip* 2, *trip* 3, dan *trip* 4.

Lakukan penugasan *forward* hingga iterasi terakhir sesuai dengan banyaknya *trip* yang ada. Sehingga iterasi terakhir adalah iterasi ke-4.

- Penugasan *reverse*:

Penugasan *reverse* hanya digunakan untuk memastikan bahwa penugasan *trip* awal berasal dari *depot* awal. Untuk *trip* 1 langkah-langkah penugasan *reverse* adalah langkah awal pilih $i = 1$. Kemudian hitung *bid depot* awal untuk *trip* 1, yaitu $f_{r1} = a_{r1} - \pi_r = -5 - 0 = -5$. Karena kandidat *predecessor trip* 1 hanya satu pilihan, yaitu *depot* awal, maka dapat diketahui secara langsung nilai maksimum *bid depot* awal untuk *trip* 1, yaitu $\beta_1 = -5$. Selanjutnya maksimum kedua dari *bid depot* awal untuk *trip* 1 dibuat sangat kecil, yaitu $\gamma_1 = -\infty$.

Update profits:

Setelah mengetahui kepastian bahwa *trip* 1 berasal dari *depot* awal, maka perbaharui nilai *profit depot* awal untuk kendaraan 1, yaitu $\pi_r = \pi_r + \beta_1 - \gamma_1 + \varepsilon = -5 + \infty + \frac{1}{6} = \infty$ dan hitung *price trip* 1 untuk *depot* awal yang terbaru, yaitu $p_1 = a_{r1} - 0 = -5$.

Update assignments:

Masukkan penugasan *depot* awal dengan *trip* 1 pada himpunan penugasan *depot* awal, yaitu $D_r(Q) = \{1\}$.

Setelah menjalankan semua iterasi, maka himpunan penugasan yang didapatkan adalah sebagai berikut:

- Himpunan perjalanan : $S(Q) = \{(1,3), (2,4)\}$
- Himpunan *source* : $D_r(Q) = \{1,2\}$
- Himpunan *sink* : $D_t(Q) = \{3,4\}$

Berdasarkan himpunan penugasan yang didapatkan, dapat disusun barisan perjalanan, misalnya (1,3) elemen himpunan perjalanan yang artinya kendaran pada *trip* 1 akan melakukan tugas menuju *trip* 3. Kemudian, 1 merupakan elemen dari himpunan *source*, artinya kendaraan yang berada di *trip* 1 berasal dari *depot* awal r. Pada himpunan *sink* terdapat 3 sebagai salah satu elemennya, artinya kendaraan di *trip* 3 akan kembali ke *depot* akhir t. Sehingga barisan *trip* yang dijalankan oleh kendaraan adalah kendaraan I : r-1-3-t dan kendaraan II : r-2-4-t. Solusi untuk semua *trip*, yaitu dengan menugaskan dua kendaraan, dimana:

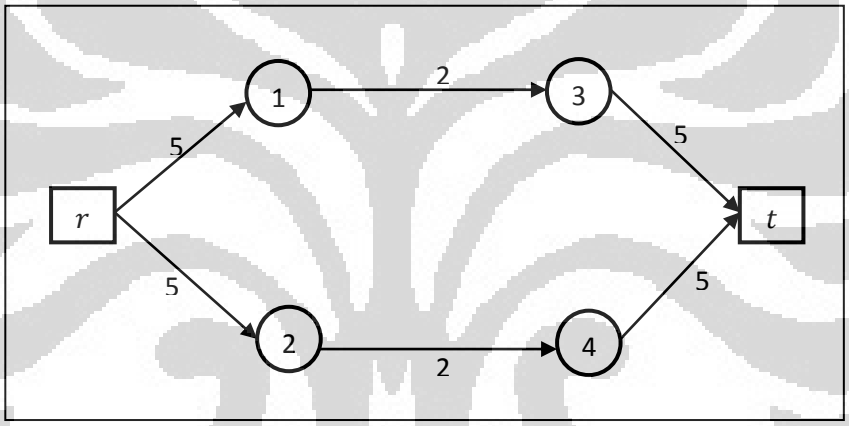
a. Kendaraan I menjalani *trip* 1 dan *trip* 3.
b. Kendaraan II menjalani *trip* 2 dan *trip* 4.

Biaya pengoperasian kendaraan dengan S-VSP adalah:

$$\sum a_{ij} = \sum \pi_i + \sum p_j$$
$$= \pi_1 + \pi_2 + \pi_3 + p_1 + p_3 + p_4$$
$$= -1\tfrac{1}{6} - \tfrac{1}{6} - 5 - 3 - \tfrac{5}{6} - 1\tfrac{5}{6} = -10\tfrac{5}{6} = -10.83 \text{ satuan biaya}$$

Karena $c_{ij} = -a_{ij}$, maka total biaya yang diperlukan untuk masalah penjadwalan kendaraan dengan S-VSP adalah biaya tetap ditambah biaya operasional kendaran, yaitu $c + \sum c_{ij} = 100 + 10.83 = 110.83$ satuan biaya.

Representasi jaringan penjadwalan kendaraan yang layak untuk data perjalanan Tabel 2.1 dengan S-VSP sebagai berikut:



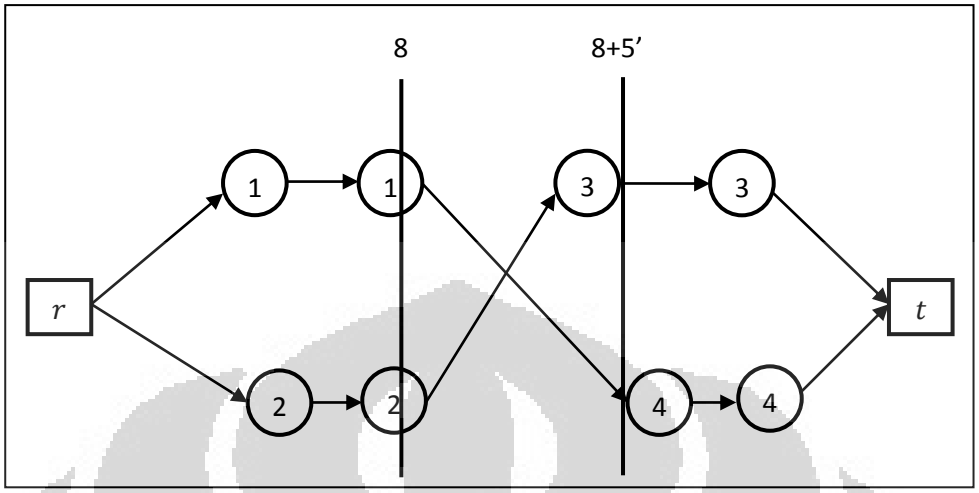
Gambar 3.3 Jaringan Penjadwalan Kendaraan Layak dengan S-VSP

Seandainya terjadi *delay* pada *trip* 1 dengan waktu *delay* lebih dari 5 menit, maka akan berakibat *delay* pada *trip* 3. Oleh sebab itu diperlukan solusi yang tepat untuk meminimumkan kendaraan yang digunakan. Jika ditugaskan suatu penambahan kendaraan, tentu saja ini bukan merupakan solusi yang optimal karena akan berpengaruh pada biaya operasional kendaraan yang bertambah. Dengan pendekatan D-VSP dapat ditemukan solusi yang optimal, yaitu tetap dengan menugaskan dua kendaraan saja. Berikut merupakan solusi dengan D-VSP:

a. Kendaraan I menjalani *trip* 1 dan *trip* 4
b. Kendaraan II menjalani *trip* 2 dan *trip* 3

Adapun *buffer time* yang digunakan adalah 5 menit. Penggambaran jadwal kendaraan angkutan umum dengan menggunakan S-VSP:

Penggambaran jadwal kendaraan angkutan umum dengan menggunakan D-VSP:



Gambar 3.4 Jaringan Penjadwalan Kendaraan dengan D-VSP

Terlihat perbedaan antara penggambaran S-VSP (Gambar 3.1) dan penggambaran D-VSP (Gambar 3.2). Penggambaran D-VSP membagi simpul menjadi dua bagian dan satu simpul tidak lagi menggambarkan suatu *trip* melainkan waktu keberangkatan beserta lokasi keberangkatan atau waktu kedatangan beserta lokasi kedatangan. Pada simpul pertama pada *trip* 1 menggambarkan keberangkatan kendaraan dari lokasi B pada pukul 07.00 dan simpul kedua pada *trip* 1 menggambarkan kedatangan kendaraan ke lokasi A pada pukul 08.00. Begitu juga pada *trip* lainnya, penggambaran simpul serupa dengan *trip* 1.

Pada ilustrasi D-VSP, *time point* T adalah pukul 08.00 dan *time point* $T + l$ adalah pukul 08.05 sehingga periode yang akan dibuat keputusan adalah [8,8+5′). Keputusan yang akan dibuat pada periode [8,8+5′) berdasarkan skenario yaitu, kendaraan tetap menjalankan tugas yang sama (skenario pertama), kendaraan bertukar tugas (skenario kedua), dan menambahkan satu kendaraan untuk menjalani *trip* yang terlambat (skenario ketiga). Pada ilustrasi masalah yang sesuai dengan data perjalanan tabel 3.1, keputusan yang dipilih adalah menukar tugas antara kendaraan I dengan kendaraan II dimana waktu *delay* kendaraan I sama dengan atau lebih dari 5 menit dan kurang dari 15 menit. Jika waktu *delay* pada kendaraan I sama dengan atau lebih dari 15 menit, maka keputusan yang harus dipilih adalah menambah satu kendaraan karena sudah

tidak memungkinkan lagi untuk kendaraan I bertukar tugas dengan kendaraan II.

Untuk menggambarkan penerapan algoritma *auction* untuk D-VSP, digunakan ilustrasi masalah sederhana sesuai dengan data perjalanan Tabel 3.1. Solusi awal yang digunakan adalah penjadwalan kendaraan statik, yaitu kendaraan yang digunakan hanya dua kendaraan dimana kendaraan I menjalani *trip* 1 dan *trip* 3 sedangkan kendaraan II menjalani *trip* 2 dan *trip* 4.

Seandainya terdapat tiga skenario pada data di atas dalam periode setelah $[T, T + l)$, dimana skenario pertama (s_1) menunjukkan kedatangan kendaraan angkutan yang terlambat tetapi tetap dapat menjalankan tugas sesuai dengan barisan *trip* yang seharusnya, skenario kedua (s_2) menunjukkan kedatangan kendaraan angkutan terlambat tetapi dapat ditukar penugasannya, dan skenario ketiga (s_3) menunjukkan kedatangan kendaraan angkutan terlambat dan menambahkan angkutan baru. Adapun probabilitas yang dimisalkan pada saat waktu ramai (*peak time*) sebagai berikut: probabilitas terjadinya skenario pertama (p^{s_1}) adalah 0.2, probabilitas terjadinya skenario kedua (p^{s_2}) adalah 0.3, dan probabilitas terjadinya skenario ketiga (p^{s_3}) adalah 0.5.

Berikut merupakan penyelesaian D-VSP dengan algoritma *auction*:

Skenario pertama

Skenario pertama menunjukkan kedatangan kendaraan angkutan yang terlambat tetapi tetap dapat menjalankan tugas sesuai dengan barisan *trip* yang seharusnya sehingga solusi barisan perjalanan pada skenario pertama sama dengan solusi perjalanan pada S-VSP. Perbedaan terdapat pada perhitungan total biaya operasional, yaitu ada biaya tambahan untuk mengetahui total biaya D-VSP. Berdasarkan hasil S-VSP didapatkan bahwa banyaknya kendaraan yang digunakan adalah 2 kendaran sama dengan kendaraan untuk skenario pertama, sehingga $B^{s_1} = 2$ dengan probabilitas telah diketahui sebelumnya yaitu p^{s_1}=0.2.

Maka, total biaya oerasional untuk skenario pertama adalah:

$$\text{Total Biaya} \quad = c + c(p^{s_1} B^{s_1}) + \sum c_{ij}$$

$= 100 + 100\ (0.2 \times 2) + 10\frac{5}{6}$

$= 150\frac{5}{6} = 150.83$ satuan biaya

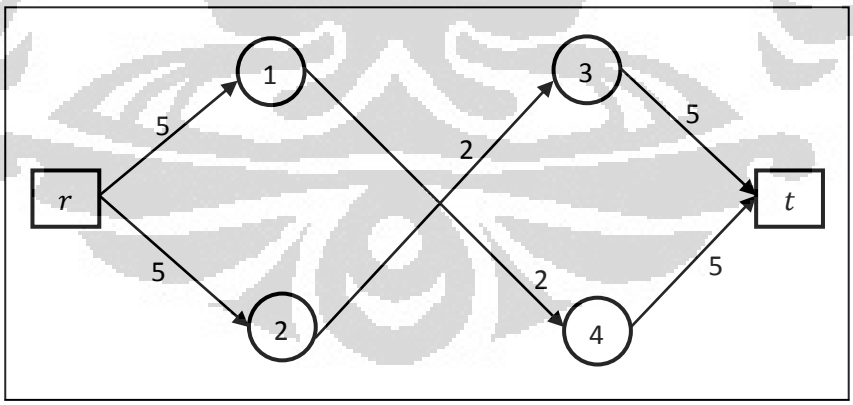
Skenario kedua

Proses algoritma yang dilakukan untuk skenario kedua sama seperti skenario pertama. Hanya saja terjadi penukaran biaya pada busur $(1,3)$ dengan biaya pada busur $(1,4)$ dan biaya antara biaya pada busur $(2,3)$ dengan biaya pada busur $(2,4)$. Sehingga, biaya antar *trip* untuk skenario 2 adalah $c'_{13} = 3$, $c'_{14} = 2$, $c'_{23} = 2$, $c'_{24} = 1$.

Berdasarkan hasil akhir pengerjaan algoritma *auction* pada skenario 2, dihasilkan perjalanan sebagai berikut:

- Himpunan perjalanan : $S(Q) = \{(1,4), (2,3)\}$
- Himpunan *source* : $D_s(Q) = \{1,2\}$
- Himpunan *sink* : $D_t(Q) = \{3,4\}$

Sehingga barisan *trip* yang dijalani oleh kendaraan adalah kendaraan I : r-1-4-t dan kendaraan II : r-2-3-t. Jika direpresentasikan dengan jaringan, maka penjadwalan kendaraan umum untuk skenario kedua sebagai berikut:



Gambar 3.5 Jaringan Penjadwalan Kendaraan untuk Skenario Kedua

Dari hasil di atas didapatkan bahwa banyaknya angkutan yang digunakan dengan keadaan *delay* (skenario dua), yaitu : $B^{s_2} = 2$ dengan probabilitas telah diketahui sebelumnya yaitu p^{s_1}=0.3.

Biaya pengoperasian kendaraan untuk skenario kedua adalah:

$$\sum a_{ij} = \sum \pi_i + \sum p_j$$

$$= \pi_1 + \pi_2 + \pi_3 + p_1 + p_3 + p_4 = -1\frac{1}{6} - \frac{1}{6} - 5 - 3 - \frac{5}{6} - 1\frac{5}{6}$$

$$= -10\frac{5}{6}$$

Karena $c_{ij} = -a_{ij}$, maka total biaya untuk penjadwalan kendaraan angkutan umum untuk skenario 2 adalah:

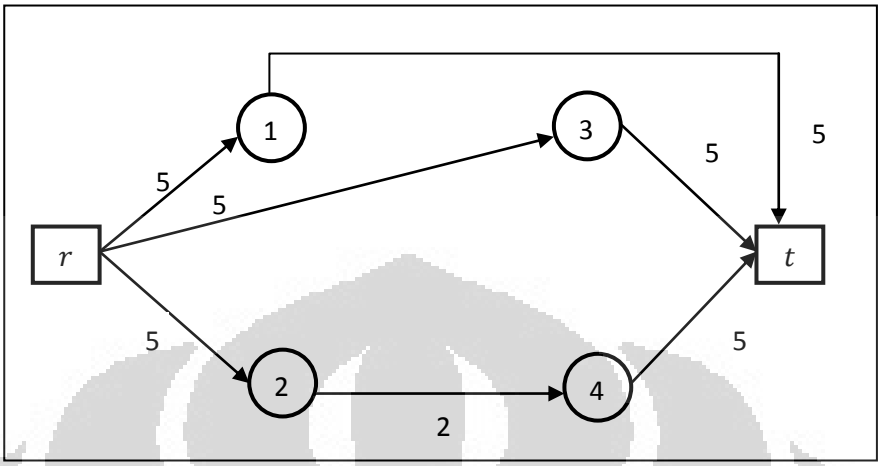
Total Biaya $= c + c(p^{s_2} B^{s_2}) + \sum c_{ij}$

$= 100 + 100\ (0.3 \times 2) + 10\frac{5}{6}$

= 160.83 satuan biaya

Skenario ketiga

Perbedaan antara skenario ketiga dan skenario lainnya terlihat pada biaya yang digunakan. Hal ini dikarenakan adanya penambahan angkutan yang disebabkan oleh *delay* lebih dari waktu toleransi (*buffer time*). Pada ilustrasi masalah sederhana pada Subbab 3.3.2, *delay* lebih dari *buffer time* terjadi pada kendaraan I, sehingga tidak memungkinkan kendaraan untuk menjalani *trip* selanjutnnya. Kendaraan tambahan berasal dari *depot* dan langsung mengerjakan tugasnya untuk menjalankan *trip* yang seharusnya, dalam ilustrasi ini menjalankan *trip* 3. Untuk skenario 3, lakukan langkah-langkah algoritma *auction* yang sama seperti pada skenario pertama dan skenario kedua. Namun, penugasan yang seharusnya dari *trip* 1 ke *trip* 3 diganti menjadi penugasan dari *depot* langsung menuju ke *trip* 3. Jika direpresentasikan dengan jaringan, maka penjadwalan kendaraan angkutan umum untuk skenario ketiga sebagai berikut:



Gambar 3.6 Jaringan Penjadwalan Kendaraan untuk Skenario Ketiga

Biaya pengoperasian kendaraan untuk skenario ketiga sebagai berikut:

$$\sum a_{ij} = \sum \pi_i + \sum p_j$$

$$= \pi_1 + \pi_2 + \pi_3 + p_1 + p_3 + p_4 = -1\frac{1}{6} - \frac{1}{6} - 5 - 3 - \frac{5}{6} - 1\frac{5}{6}$$

$$= -10\frac{5}{6}$$

Karena $c_{ij} = -a_{ij}$, maka total biaya penjadwalan kendaraan angutan umum untuk skenario 3 adalah:

Total Biaya $= c + c(p^{s_3} B^{s_3}) + \sum c_{ij}$

$= 100 + 100\ (0.5 \times 2) + 10\frac{5}{6}$

= 180.83 satuan biaya

Berdasarkan ilustrasi sederhana pada Subbab 3.3.2 ini, dapat diketahui bahwa terdapat kemungkinan untuk menggunakan skenario kedua. Jika *delay* tidak melebihi dari *buffer time*, maka penukaran tugas dapat dilakukan untuk menyelesaikan *trip* yang ada pada jadwal keberangkatan. Jika *delay* melebihi *buffer time*, maka diperlukan penambahan kendaraan untuk menyelesaikan *trip* yang ada pada jadwal keberangkatan. Hal ini akan berdampak pada penambahan biaya. Penjadwalan dengan penukaran tugas kendaraan akan lebih murah

daripada penjadwalan dengan penambahan kendaraan. Hal ini dapat berlaku jika keterlambatan kurang dari atau sama dengan *buffer time* yang telah ditentukan.

Ilustrasi pada Subbab 3.3.2 ini menggambarkan bagaimana keadaan dinamik terjadi dalam penjadwalan kendaraan. Keadaan tersebut akan terjadi ketika penjadwalan kendaraan angkutan umum mempertimbangkan waktu keberangkatan yang tidak tetap sehingga dapat mengakibatkan *delay*.

3.4 Perhitungan Biaya Denda (*Penalty*) pada Kendaraan yang Mengalami Keterlambatan Tanpa Menggunakan *Buffer Time*

Pengadaan angkutan umum adalah bagian dari pelayanan umum terhadap masyarakat. Selayaknya pengadaan angkutan umum disediakan oleh pemerintah daerah. Namun, tidak menutup kemungkinan bagi pihak swasta untuk ikut serta dalam pengadaan angkutan umum di suatu daerah. Sebagai pihak swasta yang bergerak dalam pengadaan angkutan umum melalui perusahaan yang dibangun mempunyai tujuan untuk memperoleh keuntungan yang optimal dalam pengoperasian kendaraan angkutan umum. Pihak perusahaan angkutan umum menginginkan keuntungan sebesar-besarnya untuk perusahaan yang dimiliki, tetapi sebagai pihak penumpang (konsumen) tentu tidak ingin dirugikan dengan pelayanan yang buruk, terutama keterlambatan (*delay*) pada jam keberangkatan kendaraan. Oleh sebab itu, dalam skripsi ini akan dibahas mengenai perhitungan biaya *delay* dalam pengoperasian kendaraan angkutan umum. Biaya *delay* yang berlaku bertujuan agar tidak ada pihak yang dirugikan, baik dari sisi perusahaan maupun sisi penumpang.

Untuk mengetahui biaya keterlambatan, pertama-tama tentukan banyaknya *trip* yang ada di waktu padat (*peak time*). Selanjutnya, asumsikan bahwa setiap *trip* yang ada pada waktu padat padat dijalankan oleh kendaraan berkapasitas penuh penumpang. Setiap penumpang harus membayar penuh tarif yang telah ditentukan perusahaan angkutan umum. Jika terjadi *delay* pada waktu keberangkatan, maka kerugian yang akan ditanggung oleh perusahaan adalah banyaknya *trip* yang berada di jam-jam padat dikali dengan jumlah tarif yang harus dibayar oleh seluruh penumpang dalam suatu kendaraan angkutan umum

yang berkapasitas penuh (maksimum). Apabila dibuat rumusan perhitungan biaya *delay* adalah sebagai berikut:

Misal:

c_D : total biaya *delay*.

n_T : banyaknya *trip* yang berada di waktu padat.

c_P : tarif yang dikenakan untuk setiap penumpang.

k_V : kapaitas maksimum penumpang suatu kendaraan angkutan umum.

Sehingga formulasi perhitungan total biaya *delay* adalah

$$c_D = n_T . c_P . k_V \tag{3.12}$$

BAB 4
IMPLEMENTASI ALGORITMA *AUCTION* DALAM PENJADWALAN KENDARAAN ANGKUTAN UMUM KOTA JAKARTA

Pada bab ini dibahas mengenai implementasi algoritma *auction* dalam penjadwalan angkutan umum di kota Jakarta. Rute bus yang digunakan, yaitu Manggarai-Blok M dan Manggarai-Pasar Minggu. Penjadwalan ini akan diselesaikan dengan algoritma *auction* dengan bantuan *software* Matlab R2009a.

Sebelum membuat penjadwalan kendaraan angkutan umum dengan waktu perjalanan dinamik diperlukan penjadwalan kendaraan umum statik sehingga dapat diketahui keputusan yang harus dilakukan pada saat waktu padat (*peak time*). Kemungkinan keputusan yang ada disebut dengan skenario, dimana skenario pertama kendaraan tetap menjalankan barisan *trip* pada penjadwalan statik, skenario kedua menunjukkan adanya penukaran tugas kendaraan dalam menjalani barisan *trip*, dan skenario ketiga menunjukkan adanya penambahan kendaraan dalam menjalani barisan *trip*.

Pada masalah ini terdapat beberapa asumsi yang dilakukan, yaitu:

1. Kendaraan yang digunakan berjenis bus sedang, biasa dikenal dengan Metromini atau Kopaja, dengan kapasitas maksimum 40 orang penumpang.
2. Penjadwalan dilakukan pada hari-hari kerja, yaitu dari hari Senin sampai Jumat.
3. Kendaraan tidak bernomor atau tidak berlabel sehingga memungkin untuk suatu kendaraan angkutan menjalani rute sebagai berikut:
    - Manggarai-Blok M atau sebaliknya
    - Manggarai-Pasar Minggu atau sebaliknya
    - Dapat terjadi *interlining* di Manggarai.
4. Kecepatan kendaraan adalah sama, yaitu 40km/jam
5. Bahan bakar yang digunakan adalah solar dengan kapasitas maksimum 80 liter.

6. *Timetable* diketahui berdasarkan *headway* yang telah ditetapkan. Hal ini berlaku untu setiap rute yang digunakkan dalam masalah ini. Berikut *headway* untuk antar angkutan, yaitu:
    - Pukul 05.00-06.00 *headway* angkutan adalah setiap 5 menit
    - Pukul 06.00-08.00 *headway* angkutan adalah setiap 3 menit.
    - Pukul 08.00-10.00 *headway* angkutan adalah setiap 5 menit.
    - Pukul 10.00-11.03 *headway* angkutan adalah setiap 7 menit.
    - Pukul 11.03-12.01 *headway* angkutan adalah setiap 5 menit.
    - Pukul 12.01-13.01 *headway* angkutan adalah setiap 3 menit.
    - Pukul 13.01-15.01 *headway* angkutan adalah setiap 5 menit.
    - Pukul 15.01-16.04 *headway* angkutan adalah setiap 7 menit.
    - Pukul 16.04-17.04 *headway* angkutan adalah setiap 5 menit.
    - Pukul 17.04-19.01 *headway* angkutan adalah setiap 3 menit.
    - Pukul 19.01-20.01 *headway* angkutan adalah setiap 7 menit.
    - Pukul 20.01-21.54 *headway* angkutan adalah setiap 10 menit.
7. Diketahui waktu tempuh dari masing-masing rute, yaitu:
    - Manggarai - Blok M atau sebaliknya adalah 19 menit.
    - Manggarai - Pasar Minggu atau sebaliknya adalah 15 menit.
    - Blok M - Pasar Minggu atau sebaliknya adalah 21 menit.
8. Waktu padat (*peak time*) diasumsikan pada waktu berikut:
    - Pagi : pukul 06.00-08.00
    - Siang : pukul 12.01-13.01
    - Sore : pukul17.04-19.01
9. Waktu toleransi (*buffer time*) yang digunakan adalah: $l = 5$ menit.

Berdasarkan *headway* antar angkutan dan waktu tempuh dari masing-masing *trip* dapat dibentuk *timetable* dari masalah penjadwalan ini. Daftar *trip* yang dilakukan dapat dilihat dalam lampiran.

4.1 Penjadwalan Angkutan Umum Perkotaan kota Jakarta

Menurut proses perencanaan angkutan umum, setelah didapatkan *timetable* dapat ditentukan *trip* yang digunakan untuk penjadwalan ini. Setelah didapatkan

trip, jadwal angkutan umum dapat dibuat. Sebelumnya, akan didefinisikan variabel-variabel yang digunakan dalam penjadwalan angkutan umum.

- $N = \{1,2, \ldots., 912\}$ merupakan himpunan *trip* yang mungkin.
- $E = \{(1,5), (2,5) \ldots\}$ merupakan himpunan *deadhead*.
- $A = \{(1,913), (2,913), \ldots, (912,913)\}$ merupakan himpunan *arc* dari *depot* ke *trip*.
- A^* =himpunan *arc* tanpa *long arc*
- A_1=subset dari A^* pada *peak time*
- A_2=subset dari A^* pada *offpeak time*
- Probabilitas skenario 1 terjadi $(p^{s_1}) = 0.2$
- Probabilitas skenario 2 terjadi $(p^{s_2}) = 0.3$
- Probabilitas skenario 3 terjadi $(p^{s_3}) = 0.5$
- Pendefinisian biaya untuk setiap kendaraan:
  Biaya tetap operasional (c)= Rp.100.000,00 per hari.
  Biaya tak tetapnya (c_{ij}) =Rp.650,00 per kilometer jarak tempuh.

Tujuan dari penjadwalan ini adalah meminimumkan biaya operasional kendaraan angkutan umum per hari. Masalah ini akan diselesaikan dengan algoritma *auction* dan penggunaa *software* Matlab R2009a untuk menyelesaikan masalah penjadwalan ini. Untuk menjalankan algoritma *auction* didefinisikan beberpa kondisi awal:

- $\pi_i = 0, i = 1,2, ..,912$
- $p_j = 0, i = 1,2, \ldots, 912$
- $p_t = \infty$
- $\varepsilon = \frac{1}{913}$

4.2 Solusi dari Penjadwalan Angkutan Umum Perkotaan Kota Jakarta

Solusi dinyatakan dengan urutan *trip* sesuai *timetable* yang telah dibentuk dalam lampiran. Berikut merupakan barisan *trip* yang dihasilkan dari penjadwalan kendaraan angkutan umum kota Jakarta:

- **Jadwal pertama**, yaitu dengan keadaan statik.

Hasil program akan direpresentasikan dengan tabel, dimana kolom sebelah kiri merupakan urutan rute dan kolom sebelah kanan merupakan barisan *trip* yang terjadi untuk suatu kendaraan.

Tabel 4.1 Hasil Pemrograman Penjadwalan Kendaraan Angkutan Umum Statik pada Jalur Manggarai-Blok M dan Manggarai-Pasar Minggu

Kendaraan ke-1	913 - 1 - 19 - 86 - 108 - 158 - 180 - 422 - 444 - 473 - 515 - 533 - 551 - 573 - 587 - 614 - 628 - 649 - 679 - 722 - 744 - 785 - 815 - 837 - 872 - 911 – 913
Kendaraan ke-2	913 - 2 - 16 - 30 - 44 - 62 - 84 - 122 - 144 - 165 - 195 - 218 - 232 - 245 - 275 - 294 - 308 - 321 - 351 - 386 - 408 - 429 - 459 - 502 - 516 - 541 - 559 - 602 - 616 - 633 - 651 - 694 - 716 - 753 - 783 - 826 - 850 - 874 - 906 – 913
Kendaraan ke-3	913 - 3 - 22 - 36 - 106 - 128 - 149 - 179 - 209 - 239 - 258 - 272 - 285 - 315 - 334 - 348 - 365 - 395 - 442 - 464 - 606 - 620 - 641 - 663 - 690 - 712 - 749 - 779 - 818 - 842 - 865 - 894 – 913
Kendaraan ke-4	913 - 4 - 14 - 28 - 74 - 96 - 121 - 151 - 190 - 212 - 229 - 259 - 274 - 288 - 301 - 331 - 350 - 368 - 385 - 415 - 482 - 496 - 513 - 531 - 545 - 563 - 577 - 591 - 617 - 635 - 658 - 680 - 705 - 735 - 770 - 792 - 824 - 855 - 884 – 913
Kendaraan ke-5	913 - 5 - 23 - 38 - 52 - 82 - 104 - 146 - 168 - 189 - 219 - 234 - 248 - 261 - 291 - 342 - 356 - 393 - 423 - 458 - 480 - 501 - 519 - 554 - 568 - 605 - 623 - 637 - 659 - 697 - 727 - 758 - 780 - 833 - 868 - 895 - 913
Kendaraan ke-6	913 - 6 - 20 - 34 - 48 - 70 - 92 - 113 - 143 - 174 - 196 - 217 - 247 - 270 - 284 - 297 - 327 - 337 - 367 - 402 - 424 - 457 - 479 - 514 - 528 - 553 - 571 - 594 - 608 - 625 - 643 - 666 - 688 - 778 - 800 - 841 - 878 - 910 – 913

Kendaraan ke-7	913 - 7 - 26 - 40 - 78 - 100 - 126 - 148 - 414 - 436 - 465 - 491 - 538 - 552 - 706 - 728 - 773 - 803 - 829 - 862 - 898 – 913
Kendaraan ke-8	913 - 8 - 18 - 32 - 54 - 76 - 114 - 136 - 169 - 199 - 222 - 236 - 249 - 279 - 302 - 316 - 329 - 359 - 398 - 420 - 441 - 471 - 510 - 524 - 565 - 579 - 646 - 668 - 713 - 743 - 782 - 804 - 853 - 888 – 913
Kendaraan ke-9	913 - 9 - 27 - 42 - 60 - 90 - 112 - 133 - 163 - 202 - 404 - 425 - 455 - 506 - 520 - 537 - 555 - 590 - 604 - 645 - 671 - 718 - 740 - 781 - 811 - 846 - 869 - 902 – 913
Kendaraan ke-10	913 - 10 - 24 - 58 - 80 - 118 - 140 - 173 - 203 - 238 - 252 - 289 - 319 - 338 - 352 - 373 - 403 - 445 - 655 - 686 - 708 - 729 - 759 - 786 - 808 - 840 - 889 – 913
Kendaraan ke-11	913 - 11 - 66 - 88 - 109 - 139 - 178 - 200 - 362 - 376 - 389 - 419 - 462 - 664 - 685 - 715 - 742 - 764 - 797 - 823 - 845 - 876 – 913
Kendaraan ke-12	913 - 12 - 46 - 68 - 94 - 116 - 137 - 167 - 194 - 216 - 237 - 267 - 314 - 328 - 341 - 371 - 406 - 428 - 461 - 483 - 534 - 548 - 674 - 696 - 745 - 775 - 810 - 844 - 867 - 900 – 913
Kendaraan ke-13	913 - 13 - 31 - 45 - 71 - 101 - 131 - 186 - 208 - 225 - 255 - 278 - 292 - 305 - 335 - 361 - 391 - 426 - 448 - 477 - 495 - 550 - 564 - 585 - 599 - 622 - 636 - 653 - 683 - 726 - 748 - 814 - 838 - 880 - 912 – 913
Kendaraan ke-14	913 - 15 - 50 - 72 - 125 - 155 - 198 - 220 - 418 - 440 - 469 - 487 - 530 - 544 - 569 - 583 - 597 - 615 - 662 - 684 - 709 - 739 - 774 - 796 - 836 - 882 – 913
Kendaraan ke-15	913 - 17 - 35 - 130 - 152 - 366 - 380 - 397 - 427 - 474 - 656 - 693 - 723 - 754 - 776 - 813 - 859 - 892 – 913
Kendaraan ke-16	913 - 21 - 39 - 57 - 87 - 117 - 147 - 182 - 204 - 221 - 251 - 286 - 300 - 470 - 488 - 509 - 527 - 570 - 584 - 698 - 720 - 761 - 791 - 830 - 857 - 890 – 913

Kendaraan ke-17	913 - 25 - 43 - 61 - 91 - 129 - 159 - 201 - 231 - 262 - 276 - 293 - 323 - 354 - 372 - 401 - 431 - 466 - 484 - 517 - 535 - 566 - 580 - 601 - 619 - 638 - 652 - 681 - 711 - 746 - 768 - 809 - 843 - 885 – 913
Kendaraan ke-18	913 - 29 - 47 - 150 - 172 - 193 - 223 - 242 - 256 - 269 - 299 - 326 - 340 - 353 - 383 - 405 - 435 - 494 - 508 - 642 - 660 - 689 - 719 - 762 - 784 - 825 - 871 - 904 – 913
Kendaraan ke-19	913 - 33 - 51 - 102 - 124 - 177 - 207 - 254 - 268 - 277 - 307 - 322 - 336 - 349 - 379 - 434 - 456 - 481 - 499 - 521 - 539 - 558 - 572 - 593 - 607 - 626 - 640 - 661 - 691 - 730 - 752 - 777 - 807 - 821 - 847 - 877 – 913
Kendaraan ke-20	913 - 37 - 59 - 89 - 119 - 153 - 183 - 298 - 312 - 325 - 355 - 382 - 400 - 433 - 463 - 522 - 536 - 561 - 611 - 654 - 676 - 717 - 747 - 790 - 812 - 839 - 864 - 891 – 913
Kendaraan ke-21	913 - 41 - 63 - 93 - 123 - 161 - 191 - 226 - 240 - 253 - 283 - 310 - 324 - 377 - 407 - 450 - 472 - 489 - 507 - 586 - 600 - 621 - 639 - 678 - 700 - 733 - 763 - 789 - 819 - 849 - 873 - 905 – 913
Kendaraan ke-22	913 - 49 - 79 - 110 - 132 - 157 - 187 - 214 - 360 - 369 - 399 - 454 - 476 - 493 - 511 - 562 - 576 - 589 - 603 - 634 - 648 - 677 - 707 - 741 - 771 - 802 - 832 - 863 - 896 – 913
Kendaraan ke-23	913 - 53 - 83 - 142 - 164 - 185 - 215 - 250 - 264 - 281 - 311 - 330 - 344 - 357 - 387 - 410 - 432 - 453 - 675 - 710 - 732 - 769 - 799 - 817 - 854 - 881 - 909 – 913
Kendaraan ke-24	913 - 55 - 85 - 115 - 170 - 192 - 213 - 243 - 266 - 280 - 446 - 468 - 497 - 667 - 702 - 724 - 757 - 787 - 822 - 856 - 883 - 907 – 913
Kendaraan ke-25	913 - 56 - 98 - 120 - 358 - 396 - 417 - 447 - 498 - 512 - 525 - 543 - 582 - 596 - 613 - 631 - 670 - 692 - 725 - 755 - 794 - 816 - 851 - 875 - 908 – 913

Kendaraan ke-26	913 - 64 - 81 - 111 - 145 - 175 - 206 - 228 - 241 - 271 - 290 - 304 - 317 - 347 - 378 - 392 - 421 - 451 - 478 - 492 - 505 - 523 - 546 - 560 - 581 - 595 - 618 - 632 - 657 - 687 - 721 - 751 - 798 - 828 - 860 - 901 – 913
Kendaraan ke-27	913 - 65 - 95 - 134 - 156 - 181 - 211 - 230 - 244 - 265 - 295 - 313 - 343 - 370 - 384 - 413 - 443 - 490 - 504 - 529 - 547 - 574 - 588 - 609 - 627 - 650 - 672 - 701 - 731 - 766 - 788 - 820 - 852 - 887 – 913
Kendaraan ke-28	913 - 67 - 97 - 127 - 154 - 176 - 197 - 227 - 246 - 260 - 273 - 303 - 318 - 332 - 345 - 375 - 394 - 416 - 449 - 475 - 526 - 540 - 557 - 575 - 610 - 624 - 665 - 695 - 738 - 760 - 801 - 835 - 870 - 903 – 913
Kendaraan ke-29	913 - 69 - 99 - 162 - 184 - 346 - 364 - 381 - 411 - 438 - 460 - 598 - 612 - 629 - 647 - 669 - 699 - 734 - 756 - 793 - 831 - 861 - 893 - 913
Kendaraan ke-30	913 - 73 - 103 - 138 - 160 - 430 - 452 - 485 - 503 - 542 - 556 - 682 - 704 - 737 - 767 - 806 - 848 - 879 - 899 – 913
Kendaraan ke-31	913 - 75 - 105 - 135 - 166 - 188 - 205 - 235 - 257 - 287 - 306 - 320 - 333 - 363 - 390 - 412 - 437 - 467 - 518 - 532 - 549 - 567 - 578 - 592 - 714 - 736 - 765 - 795 - 834 - 866 - 897 – 913
Kendaraan ke-32	913 - 77 - 107 - 141 - 171 - 210 - 224 - 233 - 263 - 282 - 296 - 309 - 339 - 374 - 388 - 409 - 439 - 486 - 500 - 630 - 644 - 673 - 703 - 750 - 772 - 805- 827 - 858 -886 – 913

Berdasarkan penjadwalan statik, banyaknya kendaraan yang digunakan sebanyak 32 kendaraan. Untuk mengetahui jalur mana yang dilalui dapat dilihat dari Tabel 4.1. Angka-angka pada barisan *trip* menunjukkan urutan *trip* dari *timetable* pada Lampiran 1. Ilustrasinya, untuk kendaraan ke-1, barisan trip yang terjadi adalah 913 - 1 - 19 - 86 - 108 - 158 - 180 - 422 - 444 - 473 - 515 - 533 - 551 - 573 - 587 - 614 - 628 - 649 - 679 - 722 - 744 - 785 - 815 - 837 - 872 - 911 –

913. Angka 913 menunjukkan *depot* awal maupun *depot* akhir. Angka 1 menunjukkan *trip* 1. Jika dilihat dari *timetable*, maka *trip* 1 dijalani oleh kendaraan dari lokasi awal Manggarai dengan waktu keberangkatan pukul 05.00 ke lokasi akhir Blok-M dengan waktu kedatangan pukul 05.19. Setelah menjalani *trip* 1, kendaraan yang sama akan menjalani *trip* 19, yaitu lokasi awal adalah Blok M dengan waktu keberangkatan pukul 05.20 dan lokasi akhir adalah Manggarai dengan waktu kedatangan pukul 05.39. Lakukan hal yang sama untuk mengetahui lokasi awal, lokasi akhir, waktu keberangkatan, dan waktu kedatangan pada *trip* selanjutnya.

Biaya operasional yang diperlukan untuk 32 kendaraan per hari adalah total biaya tetap untuk kendaraan per hari ditambah dengan total biaya pengoperasian kendaraan per hari, yaitu Rp3.200.000,00 + Rp4.136.500,00 = Rp7.336.500,00. Jika terjadi keterlambatan pada jam-jam padat, maka akan diberlakukan biaya keterlambatan (*penalty*) pada penjadwalan statik yang telah didapatkan. Seperti yang telah diasumsikan sebelumnya, bahwa jam-jam padat berlaku pada tiga periode. Periode pertama berlaku dipagi hari, yaitu pukul 06.00-08.00. Periode kedua berlaku di siang hari, yaitu pukul 12.01-13.01. Periode ketiga berlaku di sore hari, yaitu pukul 17.04-19.01. Untuk menghitung biaya keterlambatan, tentukan dahulu trip yang berada di jam-jam padat. Melalui hasil dari Tabel 4.1 dapat diketahui terdapat 399 trip yang berada pada jam-jam padat. Selanjutnya, asumsikan 399 trip tersebut dijalani oleh kendaraan angkutan umum yang berkapasitaas penuh penumpang (dalam kasus ini kapasitas penuh suatu kendaraan adalah 40 penumpang). Berlakukan tarif yang telah ditentukan perusahaan angkutan umum kepada setiap penumpang. Untuk Metromini atau Kopaja tarif yang berlaku adalah Rp2.000,00 per penumpang. Berarti, biaya keterlambatan yang harus dikeluarkan oleh perusahaan angkutan umum sebesar banyaknya *trip* yang berada di jam-jam padat dikali dengan jumlah tarif yang harus dibayar oleh seluruh penumpang dalam suatu kendaraan angkutan umum yang berkapasitas penuh (maksimum), yaitu 399×40×Rp2000,00 = Rp31.920.000,00.

- **Jadwal kedua**,yaitu dengan mempertimbangkan keadaan dinamik.

Tabel 4.2 Hasil Pemrograman Penjadwalan Kendaraan Angkutan Umum Dinamik pada Jalur Manggarai-Blok M dan Manggarai-Pasar Minggu

Kendaraan ke-1	913 - 1 - 19 - 62 - 92 - 138 - 168 - 197 - 231 - 370 - 384 - 401 - 435 - 470 - 676 - 713 - 747 - 794 - 840 - 876 – 913
Kendaraan ke-2	913 - 2 - 16 - 42 - 60 - 89 - 123 - 214 - 228 - 386 - 408 - 437 - 471 - 490 - 504 - 658 - 688 - 729 - 763 - 810 - 843 - 872 - 907 - 913
Kendaraan ke-3	913 - 3 - 26 - 40 - 106 - 136 - 161 - 195 - 353 - 387 - 430 - 460 - 529 - 547 - 586 - 600 - 637 - 659 - 698 - 728 - 802 - 834 - 861 - 897 – 913
Kendaraan ke-4	913 - 4 - 14 - 28 - 58 - 88 - 113 - 147 - 177 - 211 - 266 - 280 - 297 - 331 - 350 - 368 - 381 - 415 - 454 - 480 - 525 - 551 - 590 - 604 - 625 - 643 - 674 - 704 - 749 - 783 - 826 - 866 - 905 – 913
Kendaraan ke-5	913 - 5 - 23 - 54 - 84 - 109 - 143 - 178 - 208 - 233 - 267 - 298 - 312 - 329 - 363 - 402 - 432 - 465 - 495 - 546 - 560 - 581 - 595 - 605 - 623 - 673 - 707 - 754 - 784 - 828 - 868 - 909 – 913
Kendaraan ke-6	913 - 6 - 20 - 46 - 68 - 118 - 148 - 230 - 244 - 257 - 291 - 318 - 332 - 357 - 391 - 434 - 464 - 493 - 511 - 550 - 564 - 585 - 599 - 662 - 692 - 733 - 767 - 814 - 837 - 881 – 913
Kendaraan ke-7	913 - 7 - 38 - 52 - 77 - 111 - 142 - 172 - 205 - 239 - 254 - 268 - 285 - 319 - 346 - 364 - 377 - 411 - 450 - 656 - 697 - 731 - 770 - 800 - 839 - 867 - 903 – 913
Kendaraan ke-8	913 - 8 - 18 - 32 - 66 - 96 - 130 - 160 - 189 - 223 - 258 - 272 - 289 - 323 - 338 - 352 - 413 - 447 - 482 - 496 - 509 - 527 - 578 - 592 - 617 - 635 - 666 - 696 - 741 - 775 - 805 - 823 - 852 - 883 – 913

Kendaraan ke-9	913 - 9 - 27 - 78 - 108 - 170 - 200 - 225 - 259 - 274 - 288 - 305 - 339 - 373 - 407 - 466 - 672 - 705 - 739 - 774 - 804 - 844 - 890 - 913
Kendaraan ke-10	913 - 10 - 24 - 50 - 80 - 114 - 144 - 169 - 203 - 242 - 256 - 273 - 307 - 322 - 336 - 369 - 403 - 458 - 664 - 701 - 735 - 778 - 808 - 835 - 878 – 913
Kendaraan ke-11	913 - 11 - 34 - 48 - 65 - 99 - 129 - 163 - 238 - 252 - 281 - 315 - 330 - 344 - 405 - 439 - 494 - 508 - 569 - 583 - 650 - 680 - 717 - 751 - 790 - 836 - 882 – 913
Kendaraan ke-12	913 - 12 - 22 - 36 - 70 - 100 - 154 - 184 - 241 - 275 - 294 - 308 - 325 - 359 - 390 - 420 - 449 - 563 - 602 - 616 - 653 - 687 - 718 - 748 - 785 - 815 - 845 - 899 – 913
Kendaraan ke-13	913 - 13 - 31 - 74 - 104 - 186 - 220 - 229 - 263 - 286 - 300 - 317 - 351 - 394 - 424 - 453 - 483 - 538 - 552 - 573 - 587 - 630 - 644 - 689 - 723 - 762 - 792 - 832 - 871 - 906 – 913
Kendaraan ke-14	913 - 15 - 30 - 44 - 57 - 91 - 126 - 156 - 185 - 219 - 382 - 400 - 429 - 463 - 510 - 524 - 549 - 567 - 610 - 624 - 641 - 663 - 709 - 743 - 773 - 807 - 833 - 873 - 901 – 913
Kendaraan ke-15	913 - 17 - 35 - 94 - 124 - 153 - 187 - 250 - 264 - 277 - 311 - 334 - 348 - 365 - 399 - 446 - 476 - 501 - 519 - 558 - 572 - 593 - 607 - 626 - 640 - 657 - 691 - 738 - 768 - 817 - 850 - 896 – 913
Kendaraan ke-16	913 - 21 - 39 - 158 - 188 - 217 - 251 - 270 - 284 - 301 - 335 - 362 - 376 - 393 - 427 - 462 - 484 - 517 - 535 - 598 - 612 - 645 - 671 - 710 - 740 - 769 - 803 - 854 - 894 – 913
Kendaraan ke-17	913 - 25 - 43 - 61 - 95 - 125 - 159 - 198 - 224 - 410 - 440 - 473 - 491 - 554 - 568 - 589 - 603 - 621 - 639 - 669 - 703 - 746 - 776 - 825 - 862 - 912 – 913

Kendaraan ke-18	913 - 29 - 47 - 69 - 103 - 150 - 180 - 209 - 243 - 262 - 276 - 293 - 327 - 361 - 395 - 442 - 472 - 497 - 515 - 566 - 580 - 609 - 627 - 670 - 700 - 782 - 816 - 856 - 911 – 913
Kendaraan ke-19	913 - 33 - 51 - 82 - 112 - 166 - 196 - 245 - 279 - 302 - 316 - 333 - 367 - 389 - 423 - 486 - 500 - 537 - 555 - 622 - 636 - 661 - 695 - 742 - 772 - 820 - 865 - 904 – 913
Kendaraan ke-20	913 - 37 - 59 - 90 - 120 - 145 - 179 - 210 - 416 - 445 - 479 - 514 - 528 - 541 - 559 - 618 - 632 - 649 - 683 - 734 - 764 - 813 - 851 - 880 – 913
Kendaraan ke-21	913 - 41 - 63 - 93 - 127 - 165 - 199 - 337 - 371 - 422 - 452 - 481 - 499 - 562 - 576 - 629 - 647 - 681 - 715 - 761 - 795 - 822 - 855 - 888 – 913
Kendaraan ke-22	913 - 45 - 71 - 110 - 140 - 173 - 207 - 345 - 379 - 426 - 456 - 642 - 660 - 693 - 727 - 766 - 796 - 858 - 900 – 913
Kendaraan ke-23	913 - 49 - 83 - 146 - 176 - 221 - 255 - 358 - 396 - 425 - 459 - 502 - 516 - 682 - 712 - 737 - 771 - 809 - 848 - 893 – 913
Kendaraan ke-24	913 - 53 - 87 - 117 - 151 - 181 - 215 - 282 - 296 - 313 - 347 - 378 - 392 - 421 - 455 - 530 - 544 - 561 - 575 - 634 - 648 - 677 - 711 - 758 - 788 - 827 - 860 - 898 – 913
Kendaraan ke-25	913 - 55 - 85 - 119 - 202 - 412 - 457 - 675 - 706 - 736 - 781 - 819 - 853 - 885 – 913
Kendaraan ke-26	913 - 56 - 86 - 116 - 141 - 175 - 222 - 236 - 438 - 468 - 489 - 507 - 570 - 584 - 597 - 615 - 678 - 708 - 753 - 787 - 821 - 864 - 895 - 913
Kendaraan ke-27	913 - 64 - 102 - 132 - 157 - 191 - 354 - 372 - 385 - 419 - 474 - 488 - 513 - 531 - 574 - 588 - 613 - 631 - 654 - 684 - 725 - 759 - 806 - 830 - 863 - 891 – 913

Kendaraan ke-28	913 - 67 - 97 - 131 - 174 - 204 - 398 - 428 - 461 - 487 - 542 - 556 - 577 - 591 - 614 - 628 - 730 - 760 - 801 - 838 - 875 - 910 - 913
Kendaraan ke-29	913 - 72 - 98 - 128 - 190 - 216 - 237 - 271 - 290 - 304 - 321 - 355 - 406 - 436 - 469 - 679 - 726 - 756 - 797 - 859 - 887 – 913
Kendaraan ke-30	913 - 73 - 107 - 162 - 192 - 213 - 247 - 278 - 292 - 309 - 343 - 374 - 388 - 409 - 443 - 478 - 492 - 505 - 523 - 606 - 620 - 665 - 699 - 750 - 780 - 824 - 874 – 913
Kendaraan ke-31	913 - 75 - 122 - 152 - 234 - 248 - 261 - 295 - 310 - 324 - 341 - 375 - 414 - 444 - 477 - 543 - 582 - 596 - 722 - 752 - 793 - 831 - 870 - 913
Kendaraan ke-32	913 - 76 - 101 - 135 - 218 - 360 - 441 - 475 - 498 - 512 - 521 - 539 - 565 - 579 - 601 - 619 - 646 - 668 - 721 - 755 - 798 - 829 - 869 - 902 – 913
Kendaraan ke-33	913 - 79 - 134 - 164 - 193 - 227 - 246 - 260 - 269 - 303 - 314 - 328 - 349 - 383 - 418 - 448 - 485 - 503 - 533 - 571 - 594 - 608 - 633 - 651 - 714 - 744 - 789 - 847 - 884 – 913
Kendaraan ke-34	913 - 81 - 115 - 149 - 183 - 226 - 240 - 265 - 299 - 342 - 356 - 417 - 451 - 506 - 520 - 545 - 667 - 702 - 732 - 777 - 811 - 841 - 877 - 913
Kendaraan ke-35	913 - 105 - 139 - 182 - 212 - 253 - 287 - 366 - 380 - 397 - 431 - 526 - 540 - 694 - 724 - 765 - 799 - 846 - 886 – 913
Kendaraan ke-36	913 - 121 - 155 - 194 - 404 - 433 - 467 - 522 - 536 - 553 - 611 - 638 - 652 - 685 - 719 - 786 - 812 - 857 - 892 – 913
Kendaraan ke-37	913 - 133 - 167 - 206 - 232 - 249 - 283 - 306 - 320 - 518 - 532 - 686 - 716 - 745 - 779 - 818 - 849 - 889 – 913
Kendaraan ke-38	913 - 137 - 171 - 201 - 235 - 326 - 340 - 534 - 548 - 557 - 655 - 690 - 720 - 757 - 791 - 842 - 879 - 908 – 913

Berdasarkan penjadwalan dinamik, banyaknya kendaraan yang digunakan sebanyak 38 kendaraan. Terlihat bahwa terjadi penambahan kendaraan dalam menjalani *trip* disaat waktu ramai (*peak time*). Karena adanya penambahan kendaraan, maka skenario yang digunakan adalah skenario 3, dimana probabilitas terjadinya adalah 0.5 (seperti asumsi pada Subbab 4.1). Maka, total biaya operasional yang diperlukan untuk 38 kendaraan untuk skenario 3 adalah total biaya tetap kendaraan per hari ditambah dengan biaya *delay* yang dikenakan dengan menggunakan skenario 3 ditambah dengan total biaya pengoperasian kendaraan per hari, yaitu Rp3.200.000,00 + Rp100.000,00 (0.5 × 38) + Rp4.119.200,00 = Rp9.219.200,00.

4.3 Analisa Solusi

Berdasarkan Subbab 4.3 dapat diketahui banyaknya kendaraan yang harus digunakan dari setiap penjadwalan kendaraan (statik atau dinamik) dan biaya operasional yang diperlukan untuk pengoperasian kendaraan per hari. Terjadi penambahan kendaraan sebanyak 6 kendaraan pada penjadwalan dinamik. Karena adanya penambahan kendaraan, tentu saja ada penambahan biaya pengoperasian kendaraan pada penjadwalan dinamik. Namun, jika penjadwalan statik tetap diberlakukan, maka perusahaan harus membayar biaya *delay* sebesar Rp31.920.000,00. Hai ini dapat merugikan pihak perusahaan. Sehingga, lebih baik perusahaan menambah kendaraan daripada membayar biaya *delay* yang melebihi biaya operasional pada penjadwalan dinamik.

BAB 5
KESIMPULAN

Berdasarkan pembahasan pada bab-bab sebelumnya dapat disimpulkan bahwa penjadwalan kendaran angkutan umum kota yang merupakan masalah penjadwalan kendaraan dinamik atau *dynamic vehicle scheduling problem* (DVSP) dapat dimodelkan sebagai masalah penugasan *quasi* dan diselesaikan dengan pendekatan algoritma *auction*. Algoritma *auction* bertujuan untuk mendapatkan keuntungan maksimum yang didasari metode pelelangan, dimana pemberi taruhan terbesar akan dipilih dalam menentukan barisan *trip* yang dilalui kendaraan.

Jika diimplementasikan pada masalah penjadwalan kendaraan angkutan umum Jakarta dengan beberapa asumsi yang dilakukan, maka hasil yang diperoleh adalah menggunakan 38 kendaraan dengan total biaya operasional sebesar Rp9.219.200,00. Apabila dibandingkan dengan penjadwalan kendaran statik, maka solusi pada penjadwalan kendaraan dinamik memungkinkan terjadi penambahan kendaran dalam menjalankan *trip* yang ada. Walaupun terjadi penambahan kendaraan, solusi tetap optimal dikarenakan kerugian pada waktu tunggu penumpang tidak lama. Hal ini dapat menguntungkan pihak perusahaan maupun penumpang.

DAFTAR PUSTAKA

Ahuja, Ravindra. K., Magnanti, Thomas. L., Orlin, B. James.(1993). *Network Flows: Theory, Algorithms, and Applications.* United State of America: Prentice-Hall.

Bertsekas, D. P. (1992). *Auction Algorithms for Network Flow Problems: A Tutorial Introduction*. Cambridge: Massachusetts Institute of Technology.

Ceder, A. (2007). *Public Transit Planning and Operation : Theory, Modelling, and Practice*. United Kingdom: Elsevier.

Dantzig, G. B. dan Thapa, M. N. (1997). *Linear Programming* 1 : *Introduction*. New York: Springer-Verlag.

Freling, R., Wagelmans, A. P. M., dan Paixão, J. M. P. (1999). *Model and Algorithms for Vehicle Scheduling*. United Kingdom: Elsevier.

Freling, R., Wagelmans, A. P. M., dan Paixão, J. M. P. (2001, Mei). *Model and Algorithms for Single Depot Vehicle Scheduling*. United Kingdom: Elsevier.

Hartsfield, Nora dan Ringel, Gerhard. (2004). *Pearls in Graph Theory: A Comprehensive Introduction*. Doer Publication, Inc.

Hillier, Fredrick S dan Lieberman,Gerald J. (1995). *Intoduction to Operations Research*. Singapur: McGraw-Hill,Inc.

Huisman, D. (2004). *Integrated and Dynamic Vehicle and Crew Scheduling* (Vol. 325). Netherlands: Tinbergen Institute Research Series.

Nasution, M. Nur. (2004). Menejemen Transportasi. Bogor: Ghalia Indonesia.

Taha, Hamdy A. (1997). *Operational Research an Introduction*. United State of America: Prentice-Hall.

Wilson, Robin J dan Watkins, John J. (1990). *Graphs: An Introductary Approach*. New Jersey: John Wiley & Sons.

Wolsey, L. A. (1998). *Integer Programming*. New Jersey: John Wiley & Sons.

LAMPIRAN

Lampiran 1 *Timetable* Rute Manggarai (1)-Blok M(2) dan Rute Manggarai(1)-Pasar Minggu(3)

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
1	1	2	05.00
2	1	3	05.00
3	2	1	05.00
4	3	1	05.00
5	1	2	05.05
6	1	3	05.05
7	2	1	05.05
8	3	1	05.05
9	1	2	05.10
10	1	3	05.10
11	2	1	05.10
12	3	1	05.10
13	1	2	05.15
14	1	3	05.15
15	2	1	05.15
16	3	1	05.15
17	1	2	05.20
18	1	3	05.20
19	2	1	05.20
20	3	1	05.20
21	1	2	05.25
22	1	3	05.25
23	2	1	05.25
24	3	1	05.25
25	1	2	05.30
26	1	3	05.30
27	2	1	05.30
28	3	1	05.30
29	1	2	05.35
30	1	3	05.35
31	2	1	05.35
32	3	1	05.35
33	1	2	05.40
34	1	3	05.40
35	2	1	05.40
36	3	1	05.40
37	1	2	05.45
38	1	3	05.45
39	2	1	05.45
40	3	1	05.45
41	1	2	05.50
42	1	3	05.50
43	2	1	05.50
44	3	1	05.50
45	1	2	05.55
46	1	3	05.55
47	2	1	05.55
48	3	1	05.55
49	1	2	06.00
50	1	3	06.00
51	2	1	06.00
52	3	1	06.00
53	1	2	06.03
54	1	3	06.03
55	2	1	06.03
56	3	1	06.03
57	1	2	06.06
58	1	3	06.06
59	2	1	06.06
60	3	1	06.06
61	1	2	06.09
62	1	3	06.09
63	2	1	06.09
64	3	1	06.09
65	1	2	06.12
66	1	3	06.12
67	2	1	06.12
68	3	1	06.12
69	1	2	06.15
70	1	3	06.15
71	2	1	06.15
72	3	1	06.15
73	1	2	06.18
74	1	3	06.18
75	2	1	06.18
76	3	1	06.18
77	1	2	06.21
78	1	3	06.21
79	2	1	06.21
80	3	1	06.21
81	1	2	06.24
82	1	3	06.24
83	2	1	06.24
84	3	1	06.24
85	1	2	06.27
86	1	3	06.27
87	2	1	06.27
88	3	1	06.27
89	1	2	06.30
90	1	3	06.30

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
91	2	1	06.30
92	3	1	06.30
93	1	2	06.33
94	1	3	06.33
95	2	1	06.33
96	3	1	06.33
97	1	2	06.36
98	1	3	06.36
99	2	1	06.36
100	3	1	06.36
101	1	2	06.39
102	1	3	06.39
103	2	1	06.39
104	3	1	06.39
105	1	2	06.42
106	1	3	06.42
107	2	1	06.42
108	3	1	06.42
109	1	2	06.45
110	1	3	06.45
111	2	1	06.45
112	3	1	06.45
113	1	2	06.48
114	1	3	06.48
115	2	1	06.48
116	3	1	06.48
117	1	2	06.51
118	1	3	06.51
119	2	1	06.51
120	3	1	06.51
121	1	2	06.54
122	1	3	06.54
123	2	1	06.54
124	3	1	06.54
125	1	2	06.57
126	1	3	06.57
127	2	1	06.57
128	3	1	06.57
129	1	2	07.00
130	1	3	07.00
131	2	1	07.00
132	3	1	07.00
133	1	2	07.03
134	1	3	07.03
135	2	1	07.03

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
136	3	1	07.03
137	1	2	07.06
138	1	3	07.06
139	2	1	07.06
140	3	1	07.06
141	1	2	07.09
142	1	3	07.09
143	2	1	07.09
144	3	1	07.09
145	1	2	07.12
146	1	3	07.12
147	2	1	07.12
148	3	1	07.12
149	1	2	07.15
150	1	3	07.15
151	2	1	07.15
152	3	1	07.15
153	1	2	07.18
154	1	3	07.18
155	2	1	07.18
156	3	1	07.18
157	1	2	07.21
158	1	3	07.21
159	2	1	07.21
160	3	1	07.21
161	1	2	07.24
162	1	3	07.24
163	2	1	07.24
164	3	1	07.24
165	1	2	07.27
166	1	3	07.27
167	2	1	07.27
168	3	1	07.27
169	1	2	07.30
170	1	3	07.30
171	2	1	07.30
172	3	1	07.30
173	1	2	07.33
174	1	3	07.33
175	2	1	07.33
176	3	1	07.33
177	1	2	07.36
178	1	3	07.36
179	2	1	07.36
180	3	1	07.36

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
181	1	2	07.39
182	1	3	07.39
183	2	1	07.39
184	3	1	07.39
185	1	2	07.42
186	1	3	07.42
187	2	1	07.42
188	3	1	07.42
189	1	2	07.45
190	1	3	07.45
191	2	1	07.45
192	3	1	07.45
193	1	2	07.48
194	1	3	07.48
195	2	1	07.48
196	3	1	07.48
197	1	2	07.51
198	1	3	07.51
199	2	1	07.51
200	3	1	07.51
201	1	2	07.54
202	1	3	07.54
203	2	1	07.54
204	3	1	07.54
205	1	2	07.57
206	1	3	07.57
207	2	1	07.57
208	3	1	07.57
209	1	2	08.00
210	1	3	08.00
211	2	1	08.00
212	3	1	08.00
213	1	2	08.05
214	1	3	08.05
215	2	1	08.05
216	3	1	08.05
217	1	2	08.10
218	1	3	08.10
219	2	1	08.10
220	3	1	08.10
221	1	2	08.15
222	1	3	08.15
223	2	1	08.15
224	3	1	08.15
225	1	2	08.20

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
226	1	3	08.20
227	2	1	08.20
228	3	1	08.20
229	1	2	08.25
230	1	3	08.25
231	2	1	08.25
232	3	1	08.25
233	1	2	08.30
234	1	3	08.30
235	2	1	08.30
236	3	1	08.30
237	1	2	08.35
238	1	3	08.35
239	2	1	08.35
240	3	1	08.35
241	1	2	08.40
242	1	3	08.40
243	2	1	08.40
244	3	1	08.40
245	1	2	08.45
246	1	3	08.45
247	2	1	08.45
248	3	1	08.45
249	1	2	08.50
250	1	3	08.50
251	2	1	08.50
252	3	1	08.50
253	1	2	08.55
254	1	3	08.55
255	2	1	08.55
256	3	1	08.55
257	1	2	09.00
258	1	3	09.00
259	2	1	09.00
260	3	1	09.00
261	1	2	09.05
262	1	3	09.05
263	2	1	09.05
264	3	1	09.05
265	1	2	09.10
266	1	3	09.10
267	2	1	09.10
268	3	1	09.10
269	1	2	09.15
270	1	3	09.15

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
271	2	1	09.15
272	3	1	09.15
273	1	2	09.20
274	1	3	09.20
275	2	1	09.20
276	3	1	09.20
277	1	2	09.25
278	1	3	09.25
279	2	1	09.25
280	3	1	09.25
281	1	2	09.30
282	1	3	09.30
283	2	1	09.30
284	3	1	09.30
285	1	2	09.35
286	1	3	09.35
287	2	1	09.35
288	3	1	09.35
289	1	2	09.40
290	1	3	09.40
291	2	1	09.40
292	3	1	09.40
293	1	2	09.45
294	1	3	09.45
295	2	1	09.45
296	3	1	09.45
297	1	2	09.50
298	1	3	09.50
299	2	1	09.50
300	3	1	09.50
301	1	2	09.55
302	1	3	09.55
303	2	1	09.55
304	3	1	09.55
305	1	2	10.00
306	1	3	10.00
307	2	1	10.00
308	3	1	10.00
309	1	2	10.07
310	1	3	10.07
311	2	1	10.07
312	3	1	10.07
313	1	2	10.14
314	1	3	10.14
315	2	1	10.14

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
316	3	1	10.14
317	1	2	10.21
318	1	3	10.21
319	2	1	10.21
320	3	1	10.21
321	1	2	10.28
322	1	3	10.28
323	2	1	10.28
324	3	1	10.28
325	1	2	10.35
326	1	3	10.35
327	2	1	10.35
328	3	1	10.35
329	1	2	10.42
330	1	3	10.42
331	2	1	10.42
332	3	1	10.42
333	1	2	10.49
334	1	3	10.49
335	2	1	10.49
336	3	1	10.49
337	1	2	10.56
338	1	3	10.56
339	2	1	10.56
340	3	1	10.56
341	1	2	11.03
342	1	3	11.03
343	2	1	11.03
344	3	1	11.03
345	1	2	11.08
346	1	3	11.08
347	2	1	11.08
348	3	1	11.08
349	1	2	11.13
350	1	3	11.13
351	2	1	11.13
352	3	1	11.13
353	1	2	11.18
354	1	3	11.18
355	2	1	11.18
356	3	1	11.18
357	1	2	11.21
358	1	3	11.21
359	2	1	11.21
360	3	1	11.21

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
361	1	2	11.26
362	1	3	11.26
363	2	1	11.26
364	3	1	11.26
365	1	2	11.31
366	1	3	11.31
367	2	1	11.31
368	3	1	11.31
369	1	2	11.36
370	1	3	11.36
371	2	1	11.36
372	3	1	11.36
373	1	2	11.41
374	1	3	11.41
375	2	1	11.41
376	3	1	11.41
377	1	2	11.46
378	1	3	11.46
379	2	1	11.46
380	3	1	11.46
381	1	2	11.51
382	1	3	11.51
383	2	1	11.51
384	3	1	11.51
385	1	2	11.56
386	1	3	11.56
387	2	1	11.56
388	3	1	11.56
389	1	2	12.01
390	1	3	12.01
391	2	1	12.01
392	3	1	12.01
393	1	2	12.04
394	1	3	12.04
395	2	1	12.04
396	3	1	12.04
397	1	2	12.07
398	1	3	12.07
399	2	1	12.07
400	3	1	12.07
401	1	2	12.10
402	1	3	12.10
403	2	1	12.10
404	3	1	12.10
405	1	2	12.13
406	1	3	12.13
407	2	1	12.13
408	3	1	12.13
409	1	2	12.16
410	1	3	12.16
411	2	1	12.16
412	3	1	12.16
413	1	2	12.19
414	1	3	12.19
415	2	1	12.19
416	3	1	12.19
417	1	2	12.22
418	1	3	12.22
419	2	1	12.22
420	3	1	12.22
421	1	2	12.25
422	1	3	12.25
423	2	1	12.25
424	3	1	12.25
425	1	2	12.28
426	1	3	12.28
427	2	1	12.28
428	3	1	12.28
429	1	2	12.31
430	1	3	12.31
431	2	1	12.31
432	3	1	12.31
433	1	2	12.34
434	1	3	12.34
435	2	1	12.34
436	3	1	12.34
437	1	2	12.37
438	1	3	12.37
439	2	1	12.37
440	3	1	12.37
441	1	2	12.40
442	1	3	12.40
443	2	1	12.40
444	3	1	12.40
445	1	2	12.43
446	1	3	12.43
447	2	1	12.43
448	3	1	12.43
449	1	2	12.46
450	1	3	12.46

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
451	2	1	12.46
452	3	1	12.46
453	1	2	12.49
454	1	3	12.49
455	2	1	12.49
456	3	1	12.49
457	1	2	12.52
458	1	3	12.52
459	2	1	12.52
460	3	1	12.52
461	1	2	12.55
462	1	3	12.55
463	2	1	12.55
464	3	1	12.55
465	1	2	12.58
466	1	3	12.58
467	2	1	12.58
468	3	1	12.58
469	1	2	13.01
470	1	3	13.01
471	2	1	13.01
472	3	1	13.01
473	1	2	13.06
474	1	3	13.06
475	2	1	13.06
476	3	1	13.06
477	1	2	13.11
478	1	3	13.11
479	2	1	13.11
480	3	1	13.11
481	1	2	13.16
482	1	3	13.16
483	2	1	13.16
484	3	1	13.16
485	1	2	13.21
486	1	3	13.21
487	2	1	13.21
488	3	1	13.21
489	1	2	13.26
490	1	3	13.26
491	2	1	13.26
492	3	1	13.26
493	1	2	13.31
494	1	3	13.31
495	2	1	13.31

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
496	3	1	13.31
497	1	2	13.36
498	1	3	13.36
499	2	1	13.36
500	3	1	13.36
501	1	2	13.41
502	1	3	13.41
503	2	1	13.41
504	3	1	13.41
505	1	2	13.46
506	1	3	13.46
507	2	1	13.46
508	3	1	13.46
509	1	2	13.51
510	1	3	13.51
511	2	1	13.51
512	3	1	13.51
513	1	2	13.56
514	1	3	13.56
515	2	1	13.56
516	3	1	13.56
517	1	2	14.01
518	1	3	14.01
519	2	1	14.01
520	3	1	14.01
521	1	2	14.06
522	1	3	14.06
523	2	1	14.06
524	3	1	14.06
525	1	2	14.11
526	1	3	14.11
527	2	1	14.11
528	3	1	14.11
529	1	2	14.16
530	1	3	14.16
531	2	1	14.16
532	3	1	14.16
533	1	2	14.21
534	1	3	14.21
535	2	1	14.21
536	3	1	14.21
537	1	2	14.26
538	1	3	14.26
539	2	1	14.26
540	3	1	14.26

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
541	1	2	14.31
542	1	3	14.31
543	2	1	14.31
544	3	1	14.31
545	1	2	14.36
546	1	3	14.36
547	2	1	14.36
548	3	1	14.36
549	1	2	14.41
550	1	3	14.41
551	2	1	14.41
552	3	1	14.41
553	1	2	14.46
554	1	3	14.46
555	2	1	14.46
556	3	1	14.46
557	1	2	14.51
558	1	3	14.51
559	2	1	14.51
560	3	1	14.51
561	1	2	14.56
562	1	3	14.56
563	2	1	14.56
564	3	1	14.56
565	1	2	15.01
566	1	3	15.01
567	2	1	15.01
568	3	1	15.01
569	1	2	15.08
570	1	3	15.08
571	2	1	15.08
572	3	1	15.08
573	1	2	15.15
574	1	3	15.15
575	2	1	15.15
576	3	1	15.15
577	1	2	15.22
578	1	3	15.22
579	2	1	15.22
580	3	1	15.22
581	1	2	15.29
582	1	3	15.29
583	2	1	15.29
584	3	1	15.29
585	1	2	15.36

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
586	1	3	15.36
587	2	1	15.36
588	3	1	15.36
589	1	2	15.43
590	1	3	15.43
591	2	1	15.43
592	3	1	15.43
593	1	2	15.50
594	1	3	15.50
595	2	1	15.50
596	3	1	15.50
597	1	2	15.57
598	1	3	15.57
599	2	1	15.57
600	3	1	15.57
601	1	2	16.04
602	1	3	16.04
603	2	1	16.04
604	3	1	16.04
605	1	2	16.09
606	1	3	16.09
607	2	1	16.09
608	3	1	16.09
609	1	2	16.14
610	1	3	16.14
611	2	1	16.14
612	3	1	16.14
613	1	2	16.19
614	1	3	16.19
615	2	1	16.19
616	3	1	16.19
617	1	2	16.24
618	1	3	16.24
619	2	1	16.24
620	3	1	16.24
621	1	2	16.29
622	1	3	16.29
623	2	1	16.29
624	3	1	16.29
625	1	2	16.34
626	1	3	16.34
627	2	1	16.34
628	3	1	16.34
629	1	2	16.39
630	1	3	16.39

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
631	2	1	16.39
632	3	1	16.39
633	1	2	16.44
634	1	3	16.44
635	2	1	16.44
636	3	1	16.44
637	1	2	16.49
638	1	3	16.49
639	2	1	16.49
640	3	1	16.49
641	1	2	16.54
642	1	3	16.54
643	2	1	16.54
644	3	1	16.54
645	1	2	16.59
646	1	3	16.59
647	2	1	16.59
648	3	1	16.59
649	1	2	17.04
650	1	3	17.04
651	2	1	17.04
652	3	1	17.04
653	1	2	17.07
654	1	3	17.07
655	2	1	17.07
656	3	1	17.07
657	1	2	17.10
658	1	3	17.10
659	2	1	17.10
660	3	1	17.10
661	1	2	17.13
662	1	3	17.13
663	2	1	17.13
664	3	1	17.13
665	1	2	17.16
666	1	3	17.16
667	2	1	17.16
668	3	1	17.16
669	1	2	17.19
670	1	3	17.19
671	2	1	17.19
672	3	1	17.19
673	1	2	17.22
674	1	3	17.22
675	2	1	17.22

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
676	3	1	17.22
677	1	2	17.22
678	1	3	17.25
679	2	1	17.25
680	3	1	17.25
681	1	2	17.28
682	1	3	17.28
683	2	1	17.28
684	3	1	17.28
685	1	2	17.31
686	1	3	17.31
687	2	1	17.31
688	3	1	17.31
689	1	2	17.34
690	1	3	17.34
691	2	1	17.34
692	3	1	17.34
693	1	2	17.37
694	1	3	17.37
695	2	1	17.37
696	3	1	17.37
697	1	2	17.40
698	1	3	17.40
699	2	1	17.40
700	3	1	17.40
701	1	2	17.43
702	1	3	17.43
703	2	1	17.43
704	3	1	17.43
705	1	2	17.46
706	1	3	17.46
707	2	1	17.46
708	3	1	17.46
709	1	2	17.49
710	1	3	17.49
711	2	1	17.49
712	3	1	17.49
713	1	2	17.52
714	1	3	17.52
715	2	1	17.52
716	3	1	17.52
717	1	2	17.55
718	1	3	17.55
719	2	1	17.55
720	3	1	17.55

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
721	1	2	17.58
722	1	3	17.58
723	2	1	17.58
724	3	1	17.58
725	1	2	18.01
726	1	3	18.01
727	2	1	18.01
728	3	1	18.01
729	1	2	18.04
730	1	3	18.04
731	2	1	18.04
732	3	1	18.04
733	1	2	18.07
734	1	3	18.07
735	2	1	18.07
736	3	1	18.07
737	1	2	18.10
738	1	3	18.10
739	2	1	18.10
740	3	1	18.10
741	1	2	18.13
742	1	3	18.13
743	2	1	18.13
744	3	1	18.13
745	1	2	18.16
746	1	3	18.16
747	2	1	18.16
748	3	1	18.16
749	1	2	18.19
750	1	3	18.19
751	2	1	18.19
752	3	1	18.19
753	1	2	18.22
754	1	3	18.22
755	2	1	18.22
756	3	1	18.22
757	1	2	18.25
758	1	3	18.25
759	2	1	18.25
760	3	1	18.25
761	1	2	18.28
762	1	3	18.28
763	2	1	18.28
764	3	1	18.28
765	1	2	18.31
766	1	3	18.31
767	2	1	18.31
768	3	1	18.31
769	1	2	18.34
770	1	3	18.34
771	2	1	18.34
772	3	1	18.34
773	1	2	18.37
774	1	3	18.37
775	2	1	18.37
776	3	1	18.37
777	1	2	18.40
778	1	3	18.40
779	2	1	18.40
780	3	1	18.40
781	1	2	18.43
782	1	3	18.43
783	2	1	18.43
784	3	1	18.43
785	1	2	18.46
786	1	3	18.46
787	2	1	18.46
788	3	1	18.46
789	1	2	18.49
790	1	3	18.49
791	2	1	18.49
792	3	1	18.49
793	1	2	18.52
794	1	3	18.52
795	2	1	18.52
796	3	1	18.52
797	1	2	18.55
798	1	3	18.55
799	2	1	18.55
800	3	1	18.55
801	1	2	18.58
802	1	3	18.58
803	2	1	18.58
804	3	1	18.58
805	1	2	19.01
806	1	3	19.01
807	2	1	19.01
808	3	1	19.01
809	1	2	19.06
810	1	3	19.06

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
811	2	1	19.06
812	3	1	19.06
813	1	2	19.11
814	1	3	19.11
815	2	1	19.11
816	3	1	19.11
817	1	2	19.16
818	1	3	19.16
819	2	1	19.16
820	3	1	19.16
821	1	2	19.21
822	1	3	19.21
823	2	1	19.21
824	3	1	19.21
825	1	2	19.26
826	1	3	19.26
827	2	1	19.26
828	3	1	19.26
829	1	2	19.31
830	1	3	19.31
831	2	1	19.31
832	3	1	19.31
833	1	2	19.36
834	1	3	19.36
835	2	1	19.36
836	3	1	19.36
837	1	2	19.41
838	1	3	19.41
839	2	1	19.41
840	3	1	19.41
841	1	2	19.46
842	1	3	19.46
843	2	1	19.46
844	3	1	19.46
845	1	2	19.51
846	1	3	19.51
847	2	1	19.51
848	3	1	19.51
849	1	2	19.56
850	1	3	19.56
851	2	1	19.56
852	3	1	19.56
853	1	2	20.01
854	1	3	20.01
855	2	1	20.01

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
856	3	1	20.01
857	1	2	20.08
858	1	3	20.08
859	2	1	20.08
860	3	1	20.08
861	1	2	20.15
862	1	3	20.15
863	2	1	20.15
864	3	1	20.15
865	1	2	20.22
866	1	3	20.22
867	2	1	20.22
868	3	1	20.22
869	1	2	20.29
870	1	3	20.29
871	2	1	20.29
872	3	1	20.29
873	1	2	20.36
874	1	3	20.36
875	2	1	20.36
876	3	1	20.36
877	1	2	20.43
878	1	3	20.43
879	2	1	20.43
880	3	1	20.43
881	1	2	20.50
882	1	3	20.50
883	2	1	20.50
884	3	1	20.50
885	1	2	20.57
886	1	3	20.57
887	2	1	20.57
888	3	1	20.57
889	1	2	21.04
890	1	3	21.04
891	2	1	21.04
892	3	1	21.04
893	1	2	21.14
894	1	3	21.14
895	2	1	21.14
896	3	1	21.14
897	1	2	21.24
898	1	3	21.24
899	2	1	21.24
900	3	1	21.24

No	Lokasi Awal	Lokasi Akhir	Waktu Keberangkatan
901	1	2	21.34
902	1	3	21.34
903	2	1	21.34
904	3	1	21.34
905	1	2	21.44
906	1	3	21.44
907	2	1	21.44
908	3	1	21.44
909	1	2	21.54
910	1	3	21.54
911	2	1	21.54
912	3	1	21.54

Lampiran 2 *Source Code* Matlab

Implementasi Masalah Penjadwalan Angkutan Umum di Jakarta dengan Matlab R2009a

Penjadwalan dengan Keadaan Statik

```
clc;
clear;
%input berupa timetable%
tmtbl=xlsread('timetable2.xlsx');
startloc=tmtbl(:,1);
endloc=tmtbl(:,2);
traveltime=[0 19 15;19 0 21;15 21 0];
starttime=tmtbl(:,8);
n=numel(startloc);

%menentukan waktu tiba kendaraan%
for i=1:n
    endtime(i)=starttime(i)+traveltime(startloc(i),endloc(i));
end

%pendefinisian biaya untuk suatu trip%
for i=1:n
    jar(i) = 0.4*traveltime(startloc(i),endloc(i));
    c(i) = 650*jar(i);
end

%mencari arc yang mungkin terjadi%
for i = 1:n
    for j = i+1:n
        if endtime(i) + traveltime(endloc(i),startloc(j)) <=
starttime(j)
            arc(i,j) = 1;
        end
    end
end

arc(:,n+1) = 1;
arc(n+1,:) = 1;
arca = arc;
%%
%pendefinisian biaya antar trip%
for i=1:n
    for j=1:n
        if arc(i,j)==1
            cost(i,j) = 650*0.4*traveltime(endloc(i),startloc(j))
+ c(j);
        end
    end
end
%%
%pendefinisian biaya dari trip ke depot atau dari depot ke trip%
cost(:,n+1) = 10.000;
cost(n+1,:) = 10.000;
```

```
%initial condition%
S = {};
profit = zeros(1,n);
price = zeros(1,n);
epsilon = 1/(n+1);
a = -cost;
S_ujg = []; S_awal = [];

%memastikan setiap trip memiliki successor%
dpn = zeros(1,n); i=1;
while sum(dpn) < numel(dpn)
    if i == n+1
       next = find(dpn == 0);
       i = next(1);
    end

    %menugaskan kendaraan ke trip selanjutnya jika lokasi akhir
bukan depot
    k = find(S_ujg == i);
    if isempty(k)
        k = length(S)+1;
        S{k} = i;
    end

    %proses algoritma auction%
    dpn(i) = 1;
    disp(['k = ' num2str(k) '/' num2str(length(S)) '; i = '
num2str(i) '; dpn = ' num2str(dpn(i)) '; sum = '
num2str(sum(dpn))]);
    j = find(arc(i,1:n)==1);
    if ~isempty(j)
        f(i,:) = -inf*ones(1,n);
        f(i,j) = a(i,j) - price(j);
        [beta(i) j] = max(f(i,:));
        f(i,j) = -inf;
        gamma(i) = max(f(i,:));

        if gamma(i) == -inf
            j = n+1;
            profit(i)=a(i,j);
        else
            arc(i,j) = 0;
            price(j) = price(j) + beta(i) - gamma(i) + epsilon;
            profit(i) = a(i,j) - price(j);
        end

        i = j;
        disp(['terpilih ' num2str(i)]);
        for l = 1:length(S)
            jl = find(S{l} == j);
            if ~isempty(jl) && j ~= n+1
                disp([num2str(j) ' terdapat di S' num2str(l)]);
                break;
            end
        end

        if isempty(jl) || j == n+1
```

```
            S{k} = [S{k} j];
            S_ujg(k) = j;
        else
            S{k} = [S{k} S{l}(jl:end)];
            S_ujg(k) = S_ujg(l);
            i = S_ujg(k);
            S{l} = S{l}(1:jl-1);
            if isempty (S{l})
                disp(['panjang S' num2str(l) ' = '
num2str(length(S{l}))]);
                k = k-1;
                S = del(S,l);
                S_ujg = del(S_ujg,l);
            else
                S_ujg(l) = S{l}(end);
                dpn(S_ujg(l)) = 0;
                disp(['l = ' num2str(l) '; dpn ' num2str(S_ujg(l))
' = 0']);
                disp(S{l});
            end
        end
    else
       S{k} = [S{k} n+1];
       S_ujg(k) = n+1;
       next = find(dpn == 0);
       i = next(1);
    end

    if k ~= 0
        disp(S{k});
    end
    disp(['sum = ' num2str(sum(dpn))]);
end

%menghitung total biaya%
fprintf('Total cost %d\n', sum(price)+sum(profit));
fprintf('Jumlah kendaraan = %d\n',length(S));
for i=1:length(S)
    disp(['rute ' num2str(i)])
    fprintf('913');
    for j=1:length(S{i})
        fprintf(' - %d',S{i}(j));
    end
    fprintf('\n');
end
```

Penjadwalan dengan Keadaan Dinamik

```
clc;
clear;
%input berupa timetable dalam bentuk excel%
tmtbl=xlsread('timetable2.xlsx');
startloc=tmtbl(:,1);
endloc=tmtbl(:,2);
traveltime=[0 19 15;19 0 21;15 21 0];
starttime=tmtbl(:,8);
n=numel(startloc);

%menentukan waktu tiba kendaraan%
for i=1:n
    endtime(i)=starttime(i)+traveltime(startloc(i),endloc(i));
end

%penambahan buffer time sebanyak 5 menit pada saat peaktime%
%pendefinisian biaya suatu trip%
l = 5;
for i=1:n
    if (starttime(i) >= 360 && starttime(i) <= 480) ||
(starttime(i) >= 721 && starttime(i) <= 778) || (starttime(i) >=
1024 && starttime(i) <= 1138)
        endtime(i)=endtime(i)+l;
    end
    jar(i) = 0.4*traveltime(startloc(i),endloc(i));
    c(i) = 650*jar(i);
end

%mencari arc yang mungkin terjadi%
for i = 1:n
    for j = i+1:n
        if endtime(i) + traveltime(endloc(i),startloc(j)) <=
starttime(j)
            arc(i,j) = 1;
        end
    end
end

arc(:,n+1) = 1;
arc(n+1,:) = 1;
arca = arc;

%pendefinisian biaya antar trip%
for i=1:n
    for j=1:n
        if arc(i,j)==1
            cost(i,j) = 650*0.4*traveltime(endloc(i),startloc(j))
+ c(j);
        end
    end
end
%%
%biaya yang digunakan dari depot ke trip atau dari trip ke depot%
cost(:,n+1) = 10.000;
cost(n+1,:) = 10.000;
```

```
%initial condition%
S = {};
profit = zeros(1,n);
price = zeros(1,n);
epsilon = 1/(n+1);
a = -cost;
S_ujg = []; S_awal = [];

%memastikan semua trip memiliki successor%
dpn = zeros(1,n); i=1;
while sum(dpn) < numel(dpn)
    if i == n+1
       next = find(dpn == 0);
       i = next(1);
    end

   %menugaskan kendaraan ke trip selanjutnya jika lokasi akhir
bukan depot
    k = find(S_ujg == i);
    if isempty(k)
        k = length(S)+1;
        S{k} = i;
    end

    %proses algoritma auction
    dpn(i) = 1;
    disp(['k = ' num2str(k) '/' num2str(length(S)) '; i = '
num2str(i) '; dpn = ' num2str(dpn(i)) '; sum = '
num2str(sum(dpn))]);
    j = find(arc(i,1:n)==1);
    if ~isempty(j)
        f(i,:) = -inf*ones(1,n);
        f(i,j) = a(i,j) - price(j);
        [beta(i) j] = max(f(i,:));
        f(i,j) = -inf;
        gamma(i) = max(f(i,:));

        if gamma(i) == -inf
            j = n+1;
            profit(i)=a(i,j);
        else
            arc(i,j) = 0;
            price(j) = price(j) + beta(i) - gamma(i) + epsilon;
            profit(i) = a(i,j) - price(j);
        end

        i = j;
        disp(['terpilih ' num2str(i)]);
        for l = 1:length(S)
            jl = find(S{l} == j);
            if ~isempty(jl) && j ~= n+1
                disp([num2str(j) ' terdapat di S' num2str(l)]);
                break;
            end
        end
```

```
        if isempty(jl) || j == n+1
            S{k} = [S{k} j];
            S_ujg(k) = j;
        else
            S{k} = [S{k} S{l}(jl:end)];
            S_ujg(k) = S_ujg(l);
            i = S_ujg(k);
            S{l} = S{l}(1:jl-1);
            if isempty (S{l})
                disp(['panjang S' num2str(l) ' = '
num2str(length(S{l}))]);
                k = k-1;
                S = del(S,l);
                S_ujg = del(S_ujg,l);
            else
                S_ujg(l) = S{l}(end);
                dpn(S_ujg(l)) = 0;
                disp(['l = ' num2str(l) '; dpn ' num2str(S_ujg(l))
' = 0']);
                disp(S{l});
            end
        end
    else
       S{k} = [S{k} n+1];
       S_ujg(k) = n+1;
       next = find(dpn == 0);
       i = next(1);
    end

    if k ~= 0
        disp(S{k});
    end
    disp(['sum = ' num2str(sum(dpn))]);
end

%menghitung total biaya
fprintf('Total cost %d\n', sum(price)+sum(profit));
fprintf('Jumlah kendaraan = %d\n',length(S));
for i=1:length(S)
    disp(['rute ' num2str(i)])
    fprintf('913');
    for j=1:length(S{i})
        fprintf(' - %d',S{i}(j));
    end
    fprintf('\n');
end
```