UNIVERSITAS INDONESIA

OPTIMASI RUTE DAN JADWAL KAPAL DALAM MENDISTRIBUSIKAN PREMIUM, KEROSIN DAN SOLAR MENGGUNAKAN ALGORITMA *TABU SEARCH*

SKRIPSI

MARIANA R SIANIPAR
0806337775

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK INDUSTRI
DEPOK
JUNI 2012

UNIVERSITAS INDONESIA

OPTIMASI RUTE DAN JADWAL KAPAL DALAM MENDISTRIBUSIKAN PREMIUM, KEROSIN DAN SOLAR MENGGUNAKAN ALGORITMA *TABU SEARCH*

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar sarjana

MARIANA R SIANIPAR
0806337775

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK INDUSTRI
DEPOK
JUNI 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	**: Mariana R Sianipar**
NPM	**: 0806337775**
Tanda Tangan	**:** 
Tanggal	**: 28 Juni 2012**

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh	:
Nama	: Mariana R Sianipar
NPM	: 0806337775
Program Studi	: Teknik Industri
Judul Skripsi	: Optimasi Rute dan Jadwal Kapal dalam Mendistribusikan Premium, Kerosin dan Solar Menggunakan Algoritma *Tabu Search*

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik pada Program Studi Teknik Industri, Fakultas Teknik, Universitas Indonesia.

DEWAN PENGUJI

Pembimbing	: Ir. Amar Rachman, MEIM	(.....)
Penguji	: Ir. Isti Surjandari, Ph.D	(.....)
Penguji	: Ir. Fauzia Dianawati, M.Si	(.....)
Penguji	: Maya Arlini P., S.T., M.T., MBA	(.....)

Ditetapkan di : Depok

Tanggal : Juni 2012

UCAPAN TERIMA KASIH

Penulis mengucapkan puji dan syukur kepada Tuhan Yang Maha Pengasih atas berkat dan kasihNya, Penulis dapat menyelesaikan skripsi ini dengan baik. Penulis tidak lupa untuk mengucapkan terima kasih kepada yang terhormat:

1. Bapak Ir. Amar Rachman, MEIM selaku dosen pembimbing I yang telah bersedia meluangkan waktu untuk memberi pengarahan, diskusi, dan bimbingan dalam pengerjaan skripsi ini.
2. Dosen Pembimbing II Penulis, yaitu Bapak Sumarsono, ST, MT untuk pengarahan, bimbingan dan bantuan kepada penulis dalam menyelesaikan skripsi ini.
3. Rekan-rekan bimbingan Penulis yaitu Gabriela, Kristina, Andrew, Rini, Eltina atas bantuan, dukungan dan kebersamaan yang sangat menyenangkan.
4. Samuel, Noni, Friska, Roberton, Jessica, Stefani, Paulus atas segala masukan yang sangat berharga dan bantuannya dalam penyusunan program.
5. Teman-teman Program Studi Industri Departemen Teknik Industri Universitas Indonesia angkatan 2008 yang telah memberikan semangat.
6. Orang tua dan saudariku tersayang yang selalu memberikan doa, motivasi dan materi.
7. Semua pihak yang telah banyak membantu menyelesaikan skripsi ini yang tidak dapat saya sebutkan satu per satu.

Penulis menyadari pembuatan skripsi ini masih perlu masukan untuk perbaikan ke depannya. Penulis juga meminta maaf untuk kesalahan dalam pembuatan skripsi ini. Semoga Tuhan memberikan berkat bagi setiap orang yang telah membantu penyelesaian skripsi ini baik secara langsung maupun tidak langsung. Semoga skripsi ini membawa manfaat bagi pengembangan ilmu pengetahuan.

Depok, Juni 2012

Penulis

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

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

Nama : Mariana R Sianipar
NPM : 0806337775
Program Studi : Teknik Industri
Departemen : Teknik Industri
Fakultas : Fakultas Teknik Universitas Indonesia
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:

"Optimasi Rute dan Jadwal Kapal dalam Mendistribusikan Premium, Kerosin, dan Solar Menggunakan Algoritma *Tabu Search*"

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mangalihmedia/ formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan memublikasikan tugas akhir saya tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis/ pencipta sebagai pemilik Hak Cipta. Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok
Pada Tanggal : 28 Juni 2012

Yang menyatakan

(Mariana R Sianipar)

ABSTRAK

Nama : Mariana R Sianipar
Program Studi : Teknik Industri
Judul : Optimasi Rute dan Jadwal Kapal Dalam Mendistribusikan Premium, Kerosin, dan Solar Menggunakan Algoritma *Tabu Search*

Penelitian ini membahas mengenai kasus distribusi Premium, Kerosin dan Solar di perusahaan Migas yang mengalami masalah deviasi jumlah pendistribusian produk di beberapa depot utama (pelabuhan bongkar) yang disebabkan oleh keterlambatan kapal, keterbatasan draft pelabuhan dan fluktuasi permintaan. Pengoptimalan rute dan jadwal pendistribusian bahan bakar ini menggunakan Algoritma *Tabu Search* dengan mengintegrasikan 2 kapal yang berbeda jenis *Medium Range* (MR) dan *General Purpose* (GP), konsumsi harian di tiap pelabuhan bongkar berbeda akan tiap produk, stok pelabuhan muat tidak terbatas, keterbatasan draft pelabuhan sehingga menghasilkan sebuah solusi yang menjaga keberadaan persediaan pengaman dengan biaya transportasi yang minimum. Rute usulan dari penelitian ini dirancang dengan penjadwalan 30 hari menggunakan perangkat lunak Matlab versi 7 (R2000b). Hasil yang diperoleh memberikan performansi yang baik karena rute dan jadwal yang dihasilkan dapat menjaga keberadaan persediaan pengaman dengan total biaya sebesar Rp. 6,265,337,216 dengan pertimbangan Kapal MR digunakan *dedicated* untuk pelabuhan bongkar TTM.

Kata kunci: Algoritma *Tabu Search*, Persediaan Pengaman, Rute, Jadwal, Bahan Bakar (Premium, Kerosin, dan Solar)

ABSTRACT

Name : Mariana R Sianipar
Study Program : Industrial Engineering
Title :Ship Routing and Scheduling Optimization in distributing Premium, Kerosene and Solar using Tabu Search Algorithm.

This research discusses about case of fuel distribution (Premium, Kerosene and Solar) in an oil company which involved in quantity of product distribution problem in some main depot (unloading port) which are caused by ship lateness, limitation of draft and demand fluctuation. It is solved by using Tabu Search Algorithm which have model that integrate two different ships consist of Medium Range (MR) and General Purpose (GP), different daily consumption product of every unloading port, unlimited inventory loading port, and limited port draft for give a solution to maintain safety stock with low transportation cost. This routes were designed by planning horizon 30 days using Matlab 7th version (R2000b). The result prove that safety stock in each unloading port can be maintained with total cost is Rp. 6,265,337,216. MR (Medium Range) ship is dedicated to fulfill TTM port demand.

Keywords: Tabu Search Algorithm, Safety Stock, Routing, Scheduling, Fuel (Premium, Kerosene, and Solar)

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Logistik (*Supply Chain*) terdiri dari berbagai aktivitas fungsional seperti transportasi, *inventory control*, dll. Biaya untuk transportasi merupakan salah satu elemen biaya distribusi yang kontribusinya paling besar, sekitar 1/3 hingga 2/3 dari total biaya aktivitas distribusi (Ballou, 2004). Oleh karena besarnya kontribusi biaya ini, dibutuhkan suatu usaha yang dapat meningkatkan efisiensi penggunaan biaya transportasi yaitu menentukan rute dan jadwal distribusi produk dari depot sampai ke konsumen.

Ronen (1993) mendefenisikan rute sebagai penentuan urutan kunjungan kapal pada serangkaian pelabuhan dan penjadwalan sebagai penetapan urutan dan durasi setiap kegiatan yang berbeda pada rute kapal. Penjadwalan rute yang optimal harus mampu menentukan seberapa banyak masing-masing produk diangkut dari pelabuhan yang satu ke yang lainnya, pada waktu kapan, menggunakan kapal yang mana sesuai dengan kondisi dimana semua pelabuhan memiliki produk yang cukup untuk dikonsumsi, dan level persediaan tidak melebihi kapasitas *inventory* dari pelabuhan (Al-Khayyal, 2005).

Siswanto (2009) dalam artikelnya membahas tentang masalah rute dan penjadwalan inventori kapal yang menjadi beberapa sub masalah seperti pemilihan rute, pemilihan kapal, prosedur aktivitas *loading* dan *unloading* menyelesaikan dengan menggunakan metode *One Step Greedy Heuristic*. Penelitian Korsvik (2010) mengajukan solusi untuk masalah pembuatan jadwal dan rute kapal dengan membandingkan 2 metode heuristik yaitu *Tabu search* dan *Multi-start Local* dari Bronmo (2007) dengan menguji 13 kasus berdasarkan data yang diperoleh dari perusahaan *Shipping*. *Tabu Search* memberikan solusi optimal atau mendekati optimal dalam waktu yang sesuai.

Penelitian ini membahas mengenai rute dan jadwal kapal di perusahaan nasional migas Indonesia yaitu PERTAMINA yang berperan penting untuk memenuhi target *lifting* minyak nasional dan menjadi bagian dari pilar utama

ketahanan energi nasional. PERTAMINA memiliki pangsa pasar yang luas di Indonesia dan selalu berupaya menjamin ketersediaan BBM bagi masyarakat.

Kegiatan pendistribusian BBM (Bahan Bakar Minyak) dan sejumlah produk lain PERTAMINA didukung oleh 81,6% transportasi laut, dan sisanya 18,4% menggunakan transportasi non-laut. Pendistribusian bahan bakar menggunakan jalur laut mempunyai tingkat ketidakpastian yang tinggi sehingga deviasi atau perubahan yang terjadi baik pada rute, tempat *loading*, *unloading*, jenis muatan, jumlah muatan yang diangkut, dan dibongkar merupakan suatu hal yang sering ditemui di dunia perkapalan.

Contoh permasalahan yang terjadi pada pendistribusian premium, Kerosin dan solar dari bulan Januari hingga Desember 2010 adalah

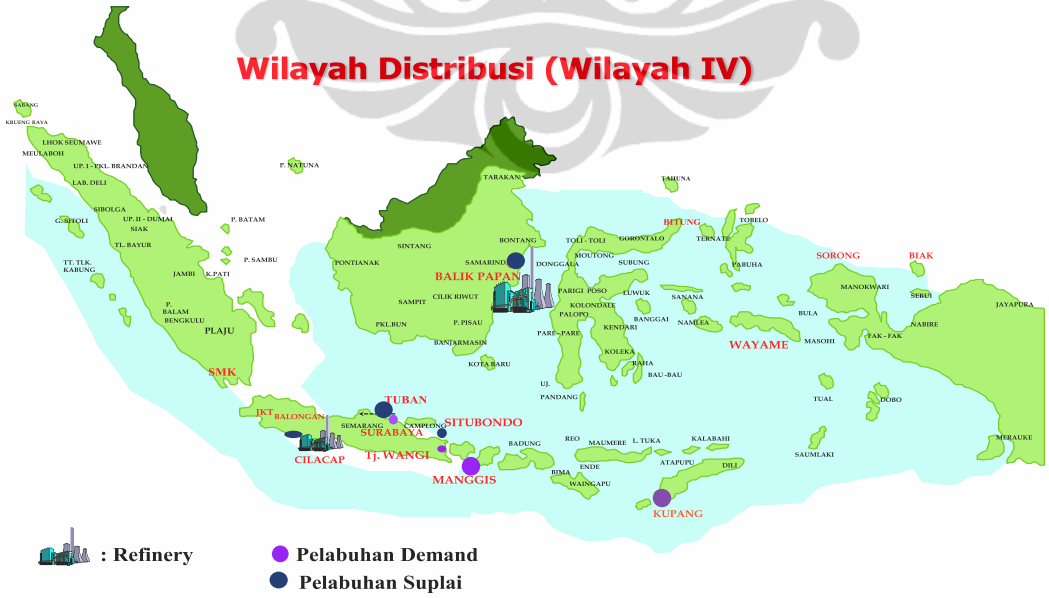
1. Keterlambatan kapal, sebagai contoh, beberapa kapal import terlambat berdampak pada antrian Jetty ex Import, kapal menunggu dermaga kosong di Balikpapan dan Surabaya karena perencanaan dermaga belum diperhitungkan, cuaca buruk yang mengakibatkan kapal Gandini tidak bisa bersandar di Tuban juga mengakibatkan terhambatnya distribusi BBM, dermaga di Pelabuhan Manggis tidak aman dijadikan sandaran kapal.
2. Tangki Pelabuhan *loading* (muat) dan *unloading* (bongkar) memiliki muatan yang terbatas, sehingga perlu diatur pendistribusiannya dengan efektif. Contohnya adalah tangki Kerosin Balikpapan penuh sementara kapal yang mengangkut ke pelabuhan *unloading* tidak ada.
3. *Consumption rate* tinggi seperti kenaikan solar di beberapa pelabuhan *unloading,* kenaikan permintaan premium dan solar di Surabaya dan Tanjung Wangi.

Gambaran umum dari permasalahan yang dibahas harus menjaga kerahasiaan data sehingga dilakukan penyingkatan nama pelabuhan dan kapal, seperti dijelaskan sebagai berikut:

1. PERTAMINA Wilayah IV memiliki empat pelabuhan muat (suplai) BBM (Bahan Bakar Minyak), yaitu BPP, CLC, TTB, XPN dan empat pelabuhan bongkar (*demand*) yaitu SBY, TWI, TTM, dan KUP. Masing- masing pelabuhan muat memiliki tangki untuk ke produk kecuali TTB (2 produk), XPN (1 produk). Masing – masing pelabuhan bongkar memiliki batas stok

minimum (persediaan pengaman) yang harus dijaga keberadaannya dan batas stok maksimal yang dapat dipenuhi. Dalam hal ini, stok pelabuhan suplai dianggap tidak terbatas. Setiap pelabuhan suplai dan kapal memiliki kemampuan berbeda dalam membongkar muatan tergantung dari kapasitas pompa yang dimiliki.

2. Kapal yang digunakan untuk wilayah ini adalah jenis MR (*Medium Range*) dan GP (*General Purpose*) yang mempunyai ruang muat serta draft lebih kecil dibanding MR. Setiap kapal mempunyai karakteristik yang berbeda baik kecepatan, draft, dan kapasitas angkut. Kapal yang digunakan dapat mengangkut tiga produk (Premium, Kerosin dan Solar) yang berbeda dan tidak tercampur antara satu dengan yang lainnya. Pelabuhan TWI dan KUP, mempunyai alur/draft dangkal sehingga kebutuhan bahan bakarnya tidak dapat disuplai oleh kapal besar. Gambar 1.1 menunjukkan gambar wilayah distribusi Premium, Kerosin dan Solar.
3. Biaya transportasi terdiri dari biaya tetap (biaya kunjungan pelabuhan), biaya variabel (bahan bakar untuk kapal berlayar) dan biaya penalti jika kapal terlambat sampai dipelabuhan tujuan yang berakibat pada stok pelabuhan *demand* (bongkar) kritis.



Gambar 1. 1 Gambar wilayah Distribusi BBM Pertamina

Sumber: Laporan Keuangan PERTAMINA, 2010

Dari permasalahan dan kondisi yang dihadapi di wilayah IV PERTAMINA diatas, diperlukan sebuah solusi agar kebutuhan masyarakat akan produk seperti Premium, Kerosin dan Solar tetap terpenuhi dengan membuat sebuah rute dan jadwal kapal.

1.2 Diagram Keterkaitan Masalah

Diagram ini berfungsi untuk memberikan gambaran secara sistemik dan visual yang menyeluruh mengenai permasalahan diatas seperti pada gambar 1.2 .

1.3 Perumusan Permasalahan

Permasalahan utama yang akan dibahas dalam penelitian ini adalah

- Bagaimana rute dan jadwal kapal dalam mendistribusikan Premium, Kerosin dan Solar yang optimal dengan menjaga persediaan pengaman?

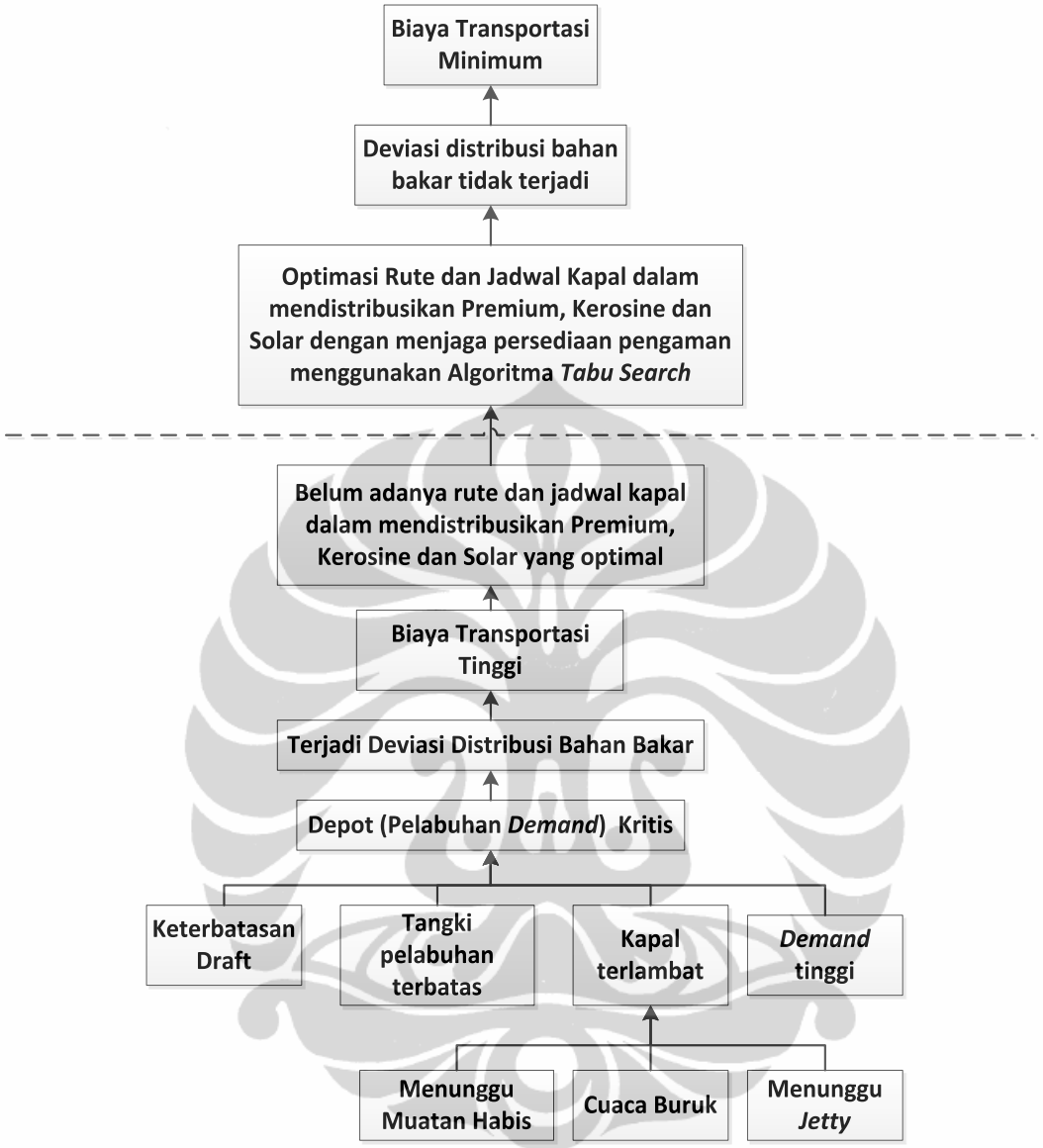
1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah memperoleh sebuah usulan mengenai rute dan jadwal kapal dalam mendistribusikan Premium, Kerosin dan Solar dengan menjaga persediaan pengaman yang optimal menggunakan Algoritma *Tabu Search*.

1.5 Ruang Lingkup Permasalahan

Ruang lingkup permasalahan dibutuhkan untuk menjaga agar penelitian ini tetap fokus pada tujuan utama. Ruang lingkup penelitian ini adalah sebagai berikut:

a. Hanya distribusi BBM wilayah IV dari titik suplai (pelabuhan suplai) ke depot utama (pelabuhan demand).
b. Stok di titik suplai dianggap tak terbatas
c. Konsumsi di tiap depot berbeda dan diasumsikan konstan selama periode tersebut
d. Biaya persediaan pengaman tidak dimasukkan karena bukan termasuk biaya operasional kapal



Gambar 1.2 Diagram Keterkaitan Masalah

1.6 Metodologi Penelitian

Metodologi atau langkah-langkah yang dilakukan dalam penelitian digambarkan dan gambar 1.3.

1.7 Sistematika Penulisan

Sistematika penulisan terdiri dari :

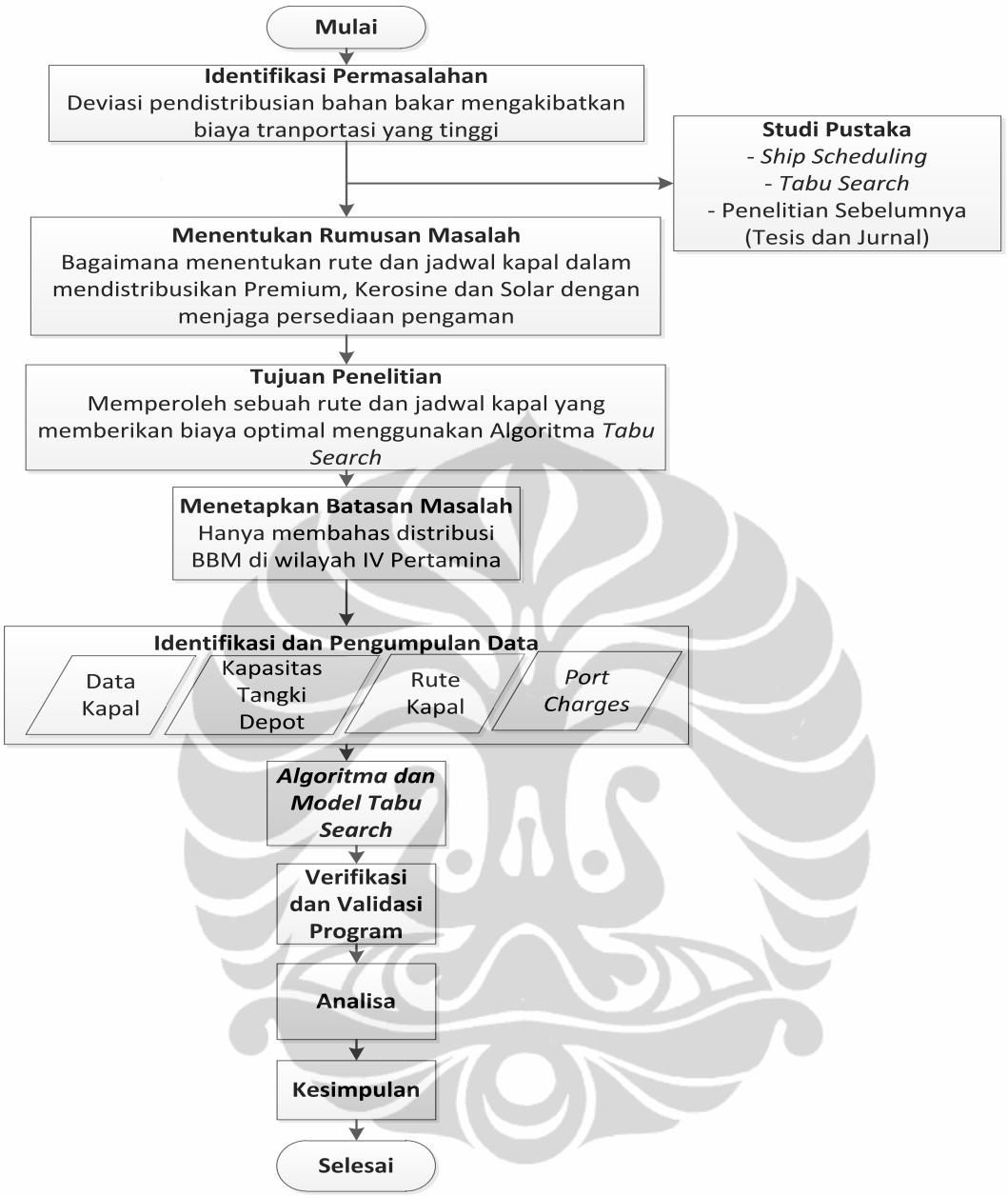
BBBab 1 Pendahuluan yang menjelaskan mengenai latar belakang permasalahan, perumusan masalah, tujuan penelitian yang ingin dicapai, ruang lingkup penelitian yang dilakukan, metodologi penelitian, serta sistematika penulisan.

Bab 2 Dasar Teori berisi landasan teori yang digunakan untuk mendukung pengerjaan topik ini tentang *ship schedulling* dan algoritma *Tabu Search*.

Bab 3 Pengumpulan dan Pengolahan Data yang menjabarkan data yang diperlukan, model matematis dan.

Bab 4 Analisis menjabarkan analisis dari hasil aktivitas pengolahan data menggunakan perangkat lunak MATLAB menggunakan Algoritma *Tabu Search*, kemudian dilakukan analisis terhadap rancangan rute distribusi tersebut. Dan yang terakhir

Bab 5 Penutup berisi kesimpulan yang didapat setelah hasil rancangan dianalisis dan saran bagi pihak yang berniat mengembangkan skripsi dengan topik yang sama.



Gambar 1.3 Metodologi Penelitian

BAB 2
DASAR TEORI

Bab ini membahas tentang landasan teori yang digunakan dalam penelitian ini. Landasan teori berupa pembahasan VRP (*Vehicle Routing Problem*), Jenis-jenis VRP, Programa Integer Campuran, serta langkah pencarian solusi permasalahan VRP menggunakan algoritma *Tabu Search*.

2.1 VRP (*Vehicle Routing Problem*)

2.1.1 Pengertian VRP

VRP (*Vehicle Routing Problem*) pertama kali dikenalkan oleh Dantzig dan Ramser pada tahun 1959. VRP merupakan manajemen distribusi barang yang memperhatikan pelayanan, periode waktu tertentu, sekelompok konsumen dengan sejumlah kendaraan yang berlokasi pada satu atau lebih depot yang dijalankan menggunakan jalur jalan yang sesuai. VRP juga dapat didefenisikan sebagai suatu pencarian solusi yang meliputi penentuan sejumlah rute (Toth dan Vigo, 2002).

Menurut Toth dan Vigo (2002), karakteristik utama VRP berdasarkan komponen-komponennya adalah sebagai berikut :

1. Jaringan jalan, biasanya direpresentasikan dalam sebuah *graph* (diagram) yang terdiri dari *arc* (lengkung atau bagian-bagian jalan) dan *vertex* (titik lokasi konsumen) dan depot. Tiap lengkung diasosiasikan dengan biaya (jarak) dan waktu perjalanan (tergantung jenis kendaraan), kondisi/karakteristik jalan, dan periode pelintasan
2. Konsumen, ditandai dengan *vertex* (titik) dan biasanya memiliki hal-hal seperti berikut.
    - Jumlah permintaan barang (untuk dikirim ataupun diambil), jenis barang dapat berbeda-beda
    - Periode pelayanan tertentu (*time windows*), dimana di luar rentang waktu tersebut konsumen tidak dapat menerima pengiriman maupun pengambilan

- Waktu yang dibutuhkan untuk menurunkan atau memuat barang (*loading/unloading time*) pada lokasi konsumen, biasanya tergantung dari jenis kendaraan
- Pengelompokan (*subset*) kendaraan yang tersedia untuk melayani konsumen (sehubungan dengan keterbatasan akses atau persyaratan pemuatan dan penurunan barang)
- Prioritas atau penalti sehubungan dengan kemampuan kendaraan untuk melayani permintaan.

3. Depot, ditandai dengan suatu titik, merupakan ujung awal dan akhir dari suatu rute kendaraan. Tiap depot memiliki sejumlah kendaraan dengan jenis dan kapasitas tertentu yang dapat digunakan mendistribusikan produk.
4. Kendaraan/armada angkut, memiliki
    a. Depot asal, dan kemungkinan untuk mengakhiri rutenya di depot lain.
    b. Kapasitas (berat, volume atau jumlah palet yang dapat diangkut)
    c. Kemungkinan untuk dipisah menjadi beberapa kompartemen untuk mengangkut barang dengan jenis yang berbeda-beda.
    d. Alat yang tersedia untuk operasi (pemuatan atau penurunan barang).
    e. Pengelompokan (*subset*) lintasan/lengkung dari diagram jaringan jalan
    f. Biaya yang berhubungan dengan penggunaan kendaraan tersebut (unit per jarak, unit per waktu, unit per rute, dan lainnya).

Dalam praktiknya, ada empat tujuan umum dalam VRP yaitu :

- Meminimumkan biaya transportasi global, terkait dengan jarak dan biaya tetap yang berhubungan dengan kendaraan.
- Meminimumkan jumlah kendaraan yang dibutuhkan untuk melayani semua konsumen
- Menyeimbangkan rute-rute dalam hal waktu perjalanan dan muatan kendaraan

- Meminimumkan penalti akibat pelayanan yang kurang memuaskan terhadap konsumen, seperti keterbatasan melayani konsumen secara penuh ataupun keterlambatan pengiriman.

2.1.2 Jenis- jenis VRP

Bruce Golden dalam bukunya menuliskan beberapa kelas atau variasi VRP adalah:

- VRP *with Time Windows* (VRPTW) yaitu kasus VRP dimana setiap konsumen memiliki batasan rentang waktu pelayanan.
- VRP *with Backhauls* (VRPB) yaitu kasus VRP dimana permintaan di masing-masing titik i bersesuaian dengan *delivery* atau *pick-up (backhaul)* yang kemudian dibawa kembali ke depot. Selama barang-barang diambil atau diantar, kuantitas dalam kendaraan tidak pernah melebihi kapasitas kendaraan.
- VRP *with Pick up and Deliveries* (VRPPD) yaitu kasus VRP dimana permintaan transportasi i dihubungkan dengan 2 titik o_i dan d_i dan permintaan q_i harus diambil dari o_i dan diantar ke d_i. Solusi yang mungkin terjadi, o_i dan d_i harus berada pada rute yang sama.
- VRP *with Multiple Use of Vehicles* yaitu kasus VRP dimana kendaraan dirancang untuk beberapa rute karena kapasitas kendaraan relatif kecil. Kendaraan sering ke depot untuk *load* atau *unload* muatan yang ada di kendaraan.
- VRP *with Multiple Depots and Periodic VRP*

Dalam VRP *Multiple Depots* (MDVRP), terdapat banyak depot pada lokasi yang berbeda. Sedangkan *Periodic* VRP (PVRP) adalah perluasan dari VRP dimana konsumen harus dikunjungi satu kali atau lebih selama *planning horizon* dari beberapa periode dengan rute ditunjukkan oleh kendaraan di masing-masing periode.

2.1.3 Metode VRP

Menurut Toth dan Vigo (2002), masalah VRP dapat diselesaikan dengan dua jenis pendekatan yaitu eksak dan heuristik. Penyelesaian heuristik dalam VRP dibagi menjadi dua, yaitu heuristik klasik dan heuristik modern (metaheuristik).

Heuristik klasik dikembangkan tahun 1960 hingga 1990, dilanjutkan dengan pengembangan algoritma metaheuristik hingga kini.

1. Pendekatan Eksak

   Pada solusi eksak, dilakukan pendekatan dengan menghitung setiap solusi yang mungkin hingga diperoleh satu yang terbaik. Beberapa jenis algoritma eksak utama penyelesaian VRP,yaitu :

   - *Branch and Bound*
   - *Branch and Cut*
   - *Set Covering Based*

   Secara umum penggunaan metode eksak untuk penyelesaian VRP akan menghabiskan waktu yang lama. Hal tersebut dikarenakan VRP termasuk dalam permasalahan NP-hard (*Non Polynominal-hard*) dimana kompleksitas penyelesaian permasalahan dan meningkat secara ekponensial dengan semakin besarnya permasalahan.

2. Heuristik Klasik

   Prosedur konstruksi dan perbaikan solusi masalah VRP yang umum digunakan saat ini berasal dari kelas heuristik klasik. Metode-metode tersebut tidak terlalu mengeksplorasi ruang pencarian solusi dan biasanya menghasilkan solusi dengan kualitas yang cukup baik dengan waktu perhitungan yang singkat. Penggunaan metode heuristik klasik dapat diperluas dengan menambahkan kendala-kendala yang mungkin timbul dalam kehidupan sehari-hari sehingga metode tersebut masih dikomersialkan.

3. Heuristik modern (Metaheuristik)

   Heuristik modern atau klasik adalah prosedur pencarian solusi umum untuk melakukan eksplorasi yang lebih dalam pada daerah yang terdapat dalam ruang solusi yang ada (Dreo, Petrowsky dan Taillard, 2006). Perbedaannya dengan heuristik klasik adalah diperbolehkannya perbaikan solusi atau penurunan fungsi tujuan. Kualitas solusi yang dihasilkan dari metode ini jauh lebih baik daripada heuristik klasik. Contoh metaheuristik adalah *simulated annealing, deterministic annealing, genetic algorithm, neural network, ant colony system,* dan *Tabu Search*. Prinsip dasarnya adalah pencarian lokal dan pencarian populasi. Dalam pencarian lokal, eksplorasi

yang intensif dilakukan terhadap ruang solusi yang berpindah dari solusi tetangga yang potensial dari yang satu ke yang lainnya. Metaheuristik menggunakan informasi yang membantu dalam mencari solusi global yang optimum.

2.2 *Mix Integer Programming* dan Metode Penyelesaian

Penelitian ini menggunakan *mix-integer programming* dalam pembuatan model matematis untuk kendala dan fungsi tujuan. *Mix Integer Linier Programming* adalah salah satu jenis program linier dengan beberapa atau semua variabel dibatasi dalam bentuk nilai integer (*dicrete*) dan kontinu.

Model Programa Linier :

$$\text{Maks. } z = c_1x_1 + c_2x_2 + \ldots\ldots\ldots\ldots + c_nx_n$$

$$\text{d. k. } \quad a_{11}x_1 + a_{12}x_2 + \ldots\ldots\ldots\ldots a_{1n}x_n \leq b_1$$

$$a_{m1}x_1 + a_{m2}x_2 + \ldots\ldots\ldots\ldots a_{mn}x_n \leq b_m$$

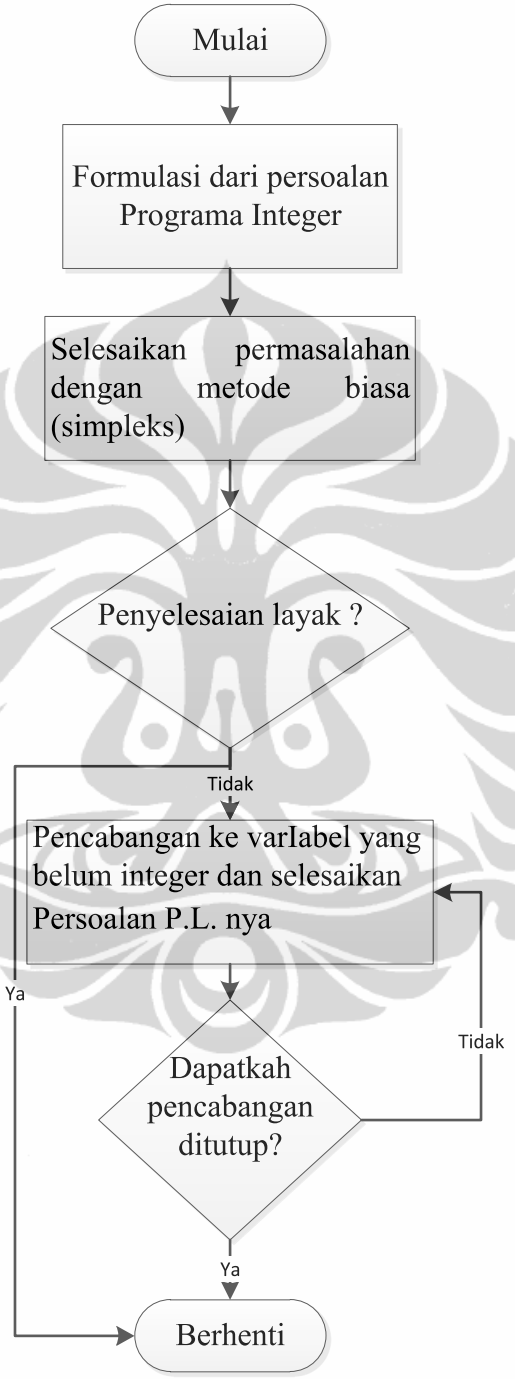
$$x_1; x_2; \ldots\ldots\ldots\ldots x_n \geq 0$$

Persoalan programa integer campuran dapat diselesaikan dengan metode *Branch and Bound*, dimana daerah penyelesaiannya harus dicari secara sistematis sampai sejumlah kecil penyelesaian integer yang mungkin diperoleh hingga diperiksa penyelesaian mana yang optimal. Pada awalnya, permasalahan diselesaikan tanpa memperhatikan kendala integernya. Jika hasilnya telah integer berarti telah diperoleh hasil yang optimal, sedangkan jika belum diperoleh hasil integer, akan dilakukan *branch and bound* yaitu membuat cabang dan membatasinya ke arah variabel yang belum integer, yaitu dengan menambah kendala baru.

1. Cabang kiri model awal ditambahkan kendala baru $x_j \leq a$, dimana a adalah bilangan integer terdekat dari x_j yang lebih kecil dari x_j.
2. Cabang kanan model awal ditambahkan kendala baru $x_j \geq b$, dimana b adalah bilangan integer terdekat dari x_j yang lebih besar dari x_j.

Kedua model yang baru terbentuk dapat diselesaikan dan diperoleh hasil yang optimal, dengan variabel keputusannya sebagian/semuanya merupakan bilangan integer. Apabila kriteria integer yang dinginkan belum terpenuhi semuanya, maka dilakukan pencabangan ke arah variabel keputusan yang belum

integer. Demikian seterusnya sampai semua kriteria integer yang dikehendaki terpenuhi. Diagram alir metode B&B digambarkan pada gambar 2.1 .



Gambar 2.1 Diagram alir tahapan *Branch & Bound*

Programa Linier Integer adalah model Programa Linier biasa yang diselesaikan dengan menggunakan metode simplex yang penyelesaiannya menggunakan perangkat lunak Lingo. Keterbatasan Lingo Industrial yang dimiliki harus menggunakan Algoritma *Tabu Search*.

2.3 Algoritma *Tabu Search*

Tabu Search (TS) adalah metaheuristik yang pertama kali diperkenalkan oleh Fred Glover pada tahun 1986. *Tabu Search* (TS) bertindak seperti algoritma *Local Search* yang tajam, tetapi dapat menerima solusi tanpa perbaikan untuk menjaring dari solusi lokal ketika semua tetangga merupakan solusi yang tanpa perbaikan. Dalam pencarian lokal, ketika solusi tetangga lebih baik, solusi ini menggantikan solusi terakhir. Saat lokal optimal diperoleh, pencarian dilanjutkan dengan pemilihan kandidat yang lebih baik dari solusi saat ini. Solusi terbaik dalam *neighborhood* dipilih sebagai solusi saat ini yang terbaru sekalipun itu tidak meningkatkan solusi saat ini. Lalu pencarian dilanjutkan hingga memungkinkan terjaringnya solusi yang global. Hal ini dapat dilihat pada ilustrasi Gambar 2.2.

Tabu Search dipandang sebagai transformasi dinamis dari *neighborhood*. Hal ini dapat menghasilkan perputaran, dan itu berarti solusi yang sebelumnya dikunjungi dapat dipilih kembali. Untuk mencegah perputaran (*cycling*), TS membuang tetangga yang sebelumnya dikunjungi. *Tabu Search* mengatur sebuah memori dari solusi atau perpindahan yang telah digunakan, biasa disebut dengan *tabu list*. *Tabu list* ini bersifat memori jangka pendek yang diperbaiki (*update*) pada setiap iterasi. Bahwasanya, kita harus memeriksa pada setiap iterasi jika solusi yang dihasilkan tidak masuk dalam daftar semua solusi yang dikunjungi. *Tabu list* biasanya terdiri dari nomor yang konstan dari perpindahan tabu.

Menurut Pham.D, terdapat tiga strategi utama dalam penggunaan *tabu list* pada *Tabu Search*, yaitu :

- Strategi pelarangan (*forbidding strategy*) untuk mengontrol apa saja yang boleh masuk ke dalam *tabu list*.
- Strategi pembebasan (*freeing strategy*) untuk memutuskan apa saja yang boleh dibebaskan dari *tabu list* dan kapan pengeluaran dilakukan.

- Strategi jangka pendek (*short-term strategy*) yang mengatur interaksi antara strategi pelarangan dan strategi pembebasan untuk membangkitkan dan menyeleksi solusi-solusi percobaan.

Solusi yang termasuk dalam status tabu dapat dilepas sesuai dengan skema *tabu tenure* yang digunakan. Skema *tabu tenure* adalah penentuan durasi status tabu. Beberapa skema yang dapat digunakan untuk melakukan kontrol periode tabu, adalah

1. *Fixed Tabu* (F-Tabu)

   Skema ini merupakan metode yang pertama kali dikenal dan digunakan. Periode tabu merupakan nilai yang tetap (tt = jumlah iterasi tertentu) selama proses *Tabu Search*.

2. *Robust Tabu* (Rb-Tabu)

   Dalam skema ini, nilai periode tabu tt diambil secara acak pada kisaran tertentu. Selama proses pencarian, nilai tt berubah setelah x iterasi. Nilai x ini juga diambil secara acak.

3. *Periodic Tabu* (P-Tabu)

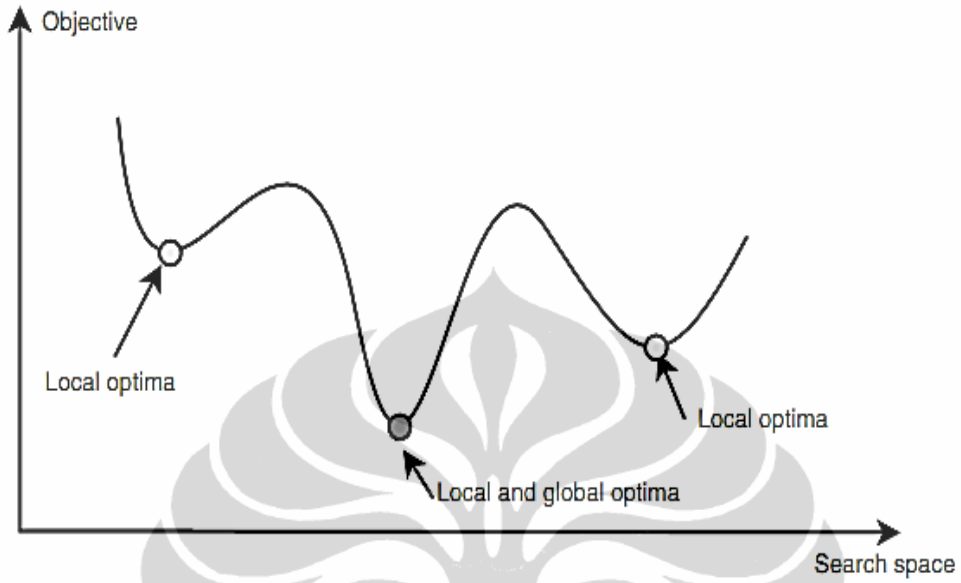
   Nilai tt yang digunakan dalam skema ini berubah secara periodik dari nilai yang terkecil, sedang, hingga besar. Perubahan nilai tt terjadi setelah x iterasi.

4. *Reversed Deterministic Tabu* (Rd-Tabu)

   Skema ini merupakan metode baru dengan mekanisme mengubah dan mengembalikan (*reverse*) nilai tt selama proses pencarian. Skema diawali dengan penggunaan beberapa nilai tt yang telah ditentukan (tt = n/p, dengan n adalah jumlah konsumen dan p diambil dari kisaran 1-9). Skema ini secara dinamis mengubah nilai tt selama proses pencarian. Nilai tt tersebut dihitung setelah x iterasi, dimana x = x-max / n x p. Nilai p pada awal pencarian adalah 9, lalu diturunkan satu setelah x iterasi. Setelah mencapai 1, nilai p diulang kembali dari 9 dan seterusnya hingga pencarian dihentikan.

Tabu Search memiliki parameter yang harus ditentukan dengan hati-hati yaitu prosedur *local search*, struktur ketetanggaan, kondisi aspirasi. Bentuk *tabu moves*, tambahan *tabu move*, maksimum ukuran *tabu list* dan kriteria

pemberhentian. Parameter ini nantinya dapat digunakan sebagai kriteria pemberhentian pencarian solusi yang optimal (Hillier and Lieberman).



Gambar 2.2 Proses Pencarian solusi optima pada *Tabu Search*

Sumber : El Ghazali Talbi, 2009

2.3.1 Penggunaan Memori

Struktur memori dalam *Tabu Search* dijalankan dengan dasar empat pada empat dimensi prinsip, yang terdiri dari atas *recency, frequency, quality* dan *influence* (Glover dan Laguna,1997). Dimensi kualitas mengacu pada kemampuan untuk mengetahui kelebihan dari solusi-solusi yang dikunjungi selama pencarian. Pada konteks tersebut, memori dapat digunakan untuk mengidentifikasi elemen-elemen umum tentang solusi yang baik dan untuk mengidentifikasi tentang jalan yang membawa pada solusi tersebut. Pada praktiknya, kualitas menjadi landasan untuk pembelajaran berbasis intensif (*intencive-based learning*) dimana penghargaan diberikan untuk mendukung tindakan-tindakan yang menghasilkan solusi yang baik dan penalti diberikan untuk menghindari solusi yang buruk. Fleksibilitas struktur memori seperti ini memungkinkan pencarian untuk diarahkan pada lingkungan *multi objective*, dimana kebaikan dari suatu arah pencarian tertentu dapat ditentukan oleh lebih dari satu fungsi. Konsep kualitas dari *Tabu Search* lebih luas dibandingkan dengan metode optimasi standar.

Memori yang digunakan dalam *Tabu Search* dapat bersifat eksplisit dan juga bersifat atributif. Memori eksplisit merekam seluruh solusi secara lengkap, biasanya terdiri dari solusi-solusi penting yang dikunjungi selama pencarian. Perluasan dari dapat memori ini merekam solusi penting yang sangat atraktif yang merupakan solusi tetangga yang belum tereksplorasi.

Sebagai alternatif, *Tabu Search* menggunakan memori atributif untuk tujuan sebagai panduan atau pengarahan. Jenis memori ini merekam informasi tentang atribut-atribut solusi yang mengalami perubahan dalam proses perpindahan dari satu solusi ke solusi yang lain. Sebagai contoh, dalam permasalahan grafik dan jaringan (*graph and network setting*), atribut dapat berupa titik/*node* atau arah/*arc* yang ditambahkan, dihilangkan atau direposisi dengan mekanisme perpindahan. Dalam penjadwalan produksi, daftar (*index of jobs*) dapat digunakan sebagai atribut untuk mencegah atau mendorong ke arah pencarian tertentu.

Peran memori pada *Tabu Search* adalah untuk membatasi pilihan hanya pada bagian (*subset*) (Ni) tertentu dengan melarang pergerakan ke beberapa solusi tetangga. Dengan kata lain, struktur tetangga N(i) dari solusi i akan berbeda-beda tiap iterasi (*dynamic neighborhood*).

2.3.2 Mekanisme Algoritma *Tabu Search*

Tahapan penggunaan Algoritma *Tabu Search* adalah :

1. Pembentukan solusi awal

   El-Ghazali (2009) dalam bukunya menuliskan, ada beberapa strategi yang digunakan untuk menghasilkan solusi awal yaitu pendekatan *random* (acak), eksak, heuristik klasik. Pemilihan penggunaan pendekatan ini berdasarkan kualitas solusi dan waktu komputasi. Hal ini tergantung pada efisiensi dan efektivitas dari pendekatan ini. Menghasilkan solusi awal *random* merupakan operasi yang cepat, tetapi metaheuristik berbeda, metode ini membutuhkan jumlah iterasi yang lebih banyak untuk menemukan titik optimal. Untuk mempercepat pencarian, pendekatan eksak atau heuristik klasik digunakan. Meskipun demikian, hal ini tidak berarti menggunakan solusi yang lebih baik sebagai solusi awal akan selalu menghasilkan lokal

optimum yang lebih baik. Pendekatan eksak dan heuristik klasik adalah seperti : *nearest neighborhood*, *insertion Heuristic*, *sweep method*, *saving method*, dll. Solusi awal yang diperoleh akan digunakan sebagai acuan awal sebagai pembanding ketika proses *Tabu Search* dimulai. Pada tahap ini ditentukan Sbest = S, C_itr = 0 (*current iteration counter*).

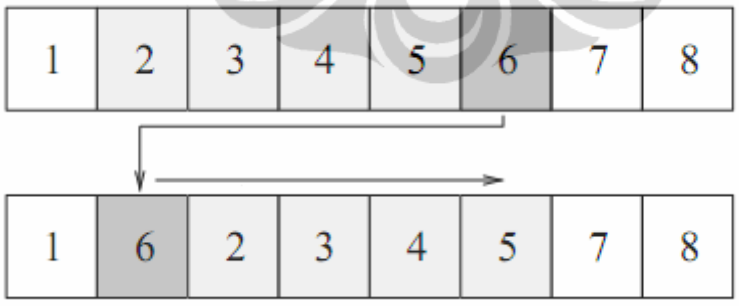
2. Inisialisasi *Tabu Search*, menentukan skema *tabu tenure* serta nilai untuk tiap parameternya. Kemudian menentukan jumlah total iterasi T_itr , B_itr = 0 (*best iteration counter*) dan jumlah solusi tetangga.
3. Melakukan iterasi

   Lakukan move untuk membuat solusi tetangga, lalu tentukan solusi saat ini (*current solution*) S menjadi S', C_itr = C_itr+1.

   - Mekanisme pembentukan solusi tetangga

Homberger dan Gehring (2004) menggunakan mekanisme himpunan struktur solusi tetangga, $N = \{N_{insert}, N_{2\text{-}opt}, N_{change}\}$ yaitu dengan memilih secara acak salah satu struktur solusi tetangga tersebut pada setiap iterasi.

a. Struktur solusi tetangga N-*insert* merupakan mekanisme pergerakan untuk memindahkan satu titik dari satu rute ke rute lain dengan notasi m_{insert} (i,k,i+) yang artinya pemindahan titik k untuk diletakkan antara titik i dan titik i+. Metode ini diperkenalkan oleh Saverlsberg (1992). Ilustrasinya dapat dilihat pada gambar dibawah ini : m_{insert} (i,k,i+) = m_{insert} (1,6,2).

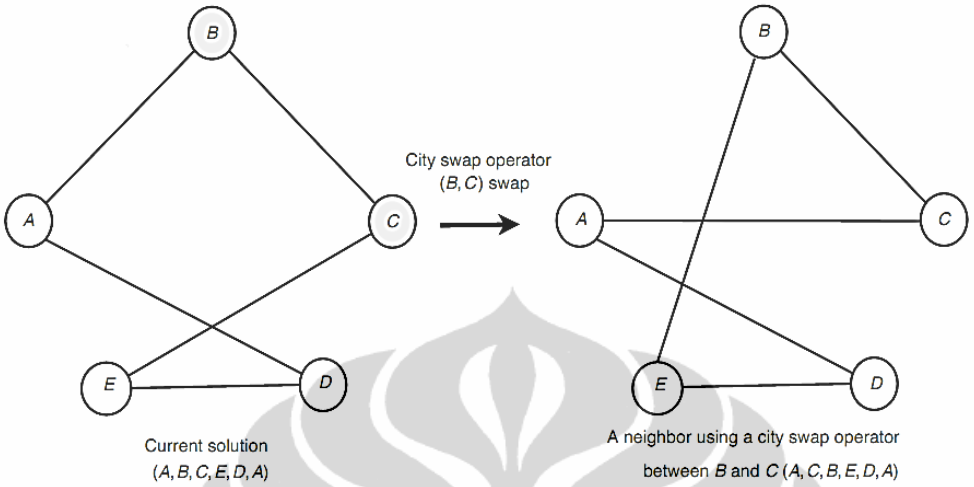


Gambar 2.3 Ilustrasi *N-insert*

Sumber: El-Ghazali Thalabi, 2009

b. Struktur solusi tetangga N_{change} atau N_{swap} merupakan mekanisme pergerakan untuk menukarkan satu titik dari satu rute ke rute lain dengan notasi m_{change} (k,i) yang artinya menukarkan k dengan i. Metode ini

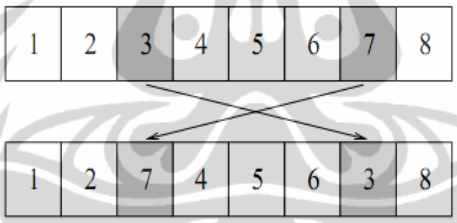
berdasarkan 1-*interchange* yang diperkenalkan oleh Osman (1993). Ilustrasi dapat dilihat digambar dibawah ini m_{change} (k,i) = m_{change} (B,C).



Gambar 2.4 Ilustrasi *N swap*

Sumber: El-Ghazali Thalabi, 2009

m_{change} (k,i) = m_{change} (3,7)



Gambar 2.5 Ilustrasi *N change*

Sumber: El-Ghazali Thalabi, 2009

4. Untuk menghindari terulangnya langkah yang diambil, maka dilakukan *tabu test*. *Tabu test* memanfaatkan *tabu list* yang sudah ada. Tujuan dari *tabu list* bukan untuk mencegah terulangnya langkah-langkah yang telah diambil, tetapi lebih kepada agar tidak mundur untuk mencegah perulangan, daftar solusi yang telah dicapai disimpan dalam sebuah tabel.
5. Alternative *move* yang lolos pada tabu test harus melewati aspiration test. Jika tidak dapat melewati *aspiration test*, maka tidak akan diteruskan ke iterasi berikutnya. Jika alternative *move* mempunyai *aspiration criteria* yang lebih

baik daripada *aspiration threshold*, maka dilakukan eksekusi terhadap alternative move tersebut dan memperbaharui memori yang tidak relevan.

6. Jika aturan pemberhentian sudah memenuhi syarat pemberhentian, maka pencarian berhenti. Kriteria penghentian pada *Tabu Search* adalah :
    1. N(i,k+1) = 0 (tidak ada solusi yang mungkin di solusi tetangga i)
    2. k lebih besar dari jumlah iterasi maksimum yang diperbolehkan
    3. Jumlah iterasi sejak perbaikan terakhir dari i* lebih besar dari angka tertentu
    4. Dapat dibuktikan bahwa solusi optimum telah diperoleh

BAB 3
PENGUMPULAN DATA

Bab ini menjelaskan mengenai data yang dibutuhkan untuk melaksanakan penelitian dan pengolahan data berupa model matematika. Data yang digunakan adalah data operasi kapal mulai Januari 2010 s/d Desember 2010 diperoleh dari VMIS (*Vessel Management Information System*) Operasi Perkapalan Pertamina meliputi data pergerakan kapal, jumlah muatan yang diangkut, jumlah pemakaian bahan bakar, dan jumlah muatan yang dimuat maupun yang dibongkar, dll.

3.1 Profil Perusahaan

Penelitian ini dilakukan di PT Pertamina (Persero), perusahaan ini adalah perusahaan minyak dan gas bumi milik pemerintah Indonesia (*National Oil Company*), yang berdiri sejak tanggal 10 Desember 1957 hingga berubah status menjadi perusahaan perseroan pada tanggal 17 September 2003 berdasarkan Undang-Undang Republik Indonesia Nomor 22 tahun 2001 pada tanggal 23 November 2001 tentang Minyak dan Gas Bumi.

Adapun tujuan dari Perusahaan Perseroan adalah untuk:

1. Mengusahakan keuntungan berdasarkan prinsip pengelolaan Perseroan secara efektif dan efisien.
2. Memberikan kontribusi dalam meningkatkan kegiatan ekonomi untuk kesejahteraan dan kemakmuran rakyat.

Untuk mencapai maksud dan tujuan tersebut, Perseroan melaksanakan kegiatan usaha sebagai berikut:

1. Menyelenggarakan usaha di bidang minyak dan gas bumi beserta hasil olahan dan turunannya.
2. Menyelenggarakan kegiatan usaha di bidang panas bumi yang ada pada saat pendiriannya, termasuk Pembangkit Listrik Tenaga Panas Bumi (PLTP) yang telah mencapai tahap akhir negosiasi dan berhasil menjadi milik Perseroan.
3. Melaksanakan pengusahaan dan pemasaran *Liquified Natural Gas* (*LNG*) dan produk lain yang dihasilkan dari kilang *LNG*.

4. Menyelenggarakan kegiatan usaha lain yang terkait atau menunjang kegiatan usaha sebagaimana dimaksud dalam nomor 1, 2, dan 3.

Sesuai dengan ketentuan dalam Undang-Undang Migas baru, perusahaan ini tidak lagi menjadi satu-satunya perusahaan yang memonopoli industri Migas dimana kegiatan usaha minyak dan gas bumi diserahkan kepada mekanisme pasar.

3.2 Pengoperasian Kapal Tanker

Dalam mendistribusikan bahan bakar, PT PERTAMINA (Persero) sangat membutuhkan armada kapal yang kuat karena wilayah Indonesia merupakan kepulauan yang dipisahkan oleh laut. Distribusi bahan bakar merupakan hasil kerjasama dan koordinasi fungsi-fungsi terkait seperti ISC (*Integrated Supply Chain*), perkapalan dan S&D (*Supply and Demand*) dimana setiap perwakilan dari fungsi tersebut ditempatkan dalam satu ruangan khusus untuk mempermudah komunikasi dan koordinasi.

Rapat evaluasi distribusi bahan bakar dijadwalkan setiap bulan untuk melihat kekurangan bahan bakar pada bulan sebelumnya sekaligus merencanakan distribusi bahan bakar selanjutnya. ISC bertugas menganalisa rencana pengadaan muatan berdasarkan informasi kebutuhan dari fungsi S&D, selanjutnya fungsi perkapalan akan mengatur jadwal kapal guna memenuhi kebutuhan tersebut. Jadwal kapal yang dibuat disebut dengan master program karena luasnya cakupan distribusi, beragamnya kebutuhan di tiap daerah dan bervariasinya tipe kapal yang dioperasikan, distribusi bahan bakar ini sering mengalami kendala baik yang disebabkan titik suplai, depot maupun kurangnya koordinasi dan komunikasi. Rute dan penjadwalan kapal yang dilaksanakan saat ini didasarkan pada nilai permintaan dan dikombinasikan dengan pengalaman dari para perencana tersebut.

3.3 Data

Data yang digunakan dalam penelitian ini merupakan data operasional kapal yang didapatkan dari fungsi operasional perkapalan, data pelabuhan yang didapatkan dari fungsi *port management* dan buku informasi pelabuhan edisi VI tahun 2006 yang diterbitkan oleh PT Pertamina (Persero), di tiap depot didapatkan

dari fungsi S&D. Data tersebut merupakan data satu tahun dari bulan Januari 2010 sampai dengan Desember 2010.

3.3.1 Data Pelabuhan

3.3.1.1 Pelabuhan *Loading* (Muat)

Pelabuhan *loading* ada empat buah dengan karakteristik yang berbeda-beda diantaranya adalah sebagai berikut :

1. BPP merupakan Kilang yang diperuntukkan untuk memenuhi kebutuhan bahan bakar di Kalimantan, Sulawesi, dan sebagian Irian Jaya. Pelabuhan ini terletak pada koordinat $01^0$15'10''LS/$116^0$48'30''BT
   Kapasitas tangki: Premium 8.100 T, Kerosin 17.000 T, Solar 12.870 T.
2. CLC merupakan Kilang yang memperoleh minyak mentah dari wilayah Arab dan diolah untuk memenuhi kebutuhan bahan bakar di pulau Jawa, Bali dan sebagian Sumatera. Pelabuhan ini terletak pada koordinat $07^0$42'01''LS/$108^0$59'28''BT.
   Kapasitas tangki : Premium 96.000 KL, Kerosin 112.000 KL, Solar 124.000 KL.
3. TTB adalah Depot utama yang baru dioperasikan mulai pertengahan tahun 2010. TTB diproyeksikan sebagai depot penunjang wilayah IV dan untuk menggantikan fungsi XPN. Pelabuhan ini terletak pada koordinat $06^0$43'52''LS/$112^0$09'20''BT.
   Kapasitas tangki : Premium 159.000 KL , Solar 200.000 KL.
4. XPN merupakan *Floating storage* sehingga tidak terdapat biaya pelabuhan saat kapal muat di area tersebut. Pelabuhan ini terletak pada koordinat : $07^0$37'24''LS/$113^0$54'36''BT
   Kapasitas tangki : Solar 80.000 KL

3.3.1.2 Pelabuhan *Unloading* (Bongkar)

Pelabuhan *unloading* ada empat dengan karakteristik berbeda-beda juga.

1. Pelabuhan SBY. Pelabuhan ini dapat dilayani oleh kapal GP (*General Purpose*) dan MR (*Medium Range*). Premium dianggap 0 karena secara operasional kapal pertamina perkapalan sangat jarang mensuplai premium

karena telah dipenuhi oleh kapal import dari Singapura yang operasionalnya ditangani oleh *Pertamina Energy Trading Limited* (PERTRAL) yang berada di Singapura dimana solar disuplai melalui pipa dari Tuban.

Terletak pada koordinat : $07^0$11'56''LS/$112^0$43'02.5''BT

Kapasitas Tangki : Premium 14.290 KL, Kerosin 31.710 KL, Solar 13.282 KL.

2. Pelabuhan TTM. Kargo premium di pelabuhan ini sebagian disuplai dari import, sehingga konsumsi per hari untuk premium nilainya akan dibagi dua.

   Terletak pada koordinat : $08^0$30'41''LS/$115^0$30'31''BT.

   Kapasitas tangki : Premium 34.500 KL, Kerosin 30.000 KL, Solar 57.500 KL.
3. Pelabuhan TWI. Pelabuhan ini mempunyai draft yang terbatas sehingga hanya kapal MR yang tidak dapat melakukan bongkar.

   Terletak pada koordinat : $08^0$08'00''LS/$114^0$24'05''BT

   Kapasitas tangki : Premium 23.818 KL, Kerosin 9.566 KL, Solar 18.900 KL.
4. Pelabuhan KUP. Pelabuhan ini mempunyai draft yang terbatas sehingga hanya kapal MR yang tidak dapat melakukan bongkar.

   Terletak pada koordinat : $10^0$12'23''LS/$123^0$31'16.6''BT

   Kapasitas Tangki : Premium 6.166 KL, Kerosin 8.452 KL, Solar 12.122 KL.

3.3.2 Data Kapal, Kapasitas dan Stok Awal

3.3.2.1 Data Kapal

Ada dua tipe kapal yang digunakan dalam mendistribusikan bahan bakar di wilayah IV, yaitu MR (Medium Range) dengan kisaran 25.000–45.000 LTDW dan GP (General Purpose) dengan kisaran 6.500–25.000 LTDW (*Long Ton Dead Weight*). Setiap jenis kapal memiliki karakteristik yang berbeda – beda. Berikut adalah data kapal GP (PGD) serta MR (GDN) yang akan dipakai dalam penelitian ini untuk mensuplai bahan bakar di wilayah IV.

Tabel 3.1 Karakteristik kapal

Karakteristik Kapal	General Purpose (17,781 LTDW)	Medium Range (32,042 LTDW)
Nama Kapal	PGD	GDN
Length Overall (Loa)	158.00 m	174.05 m
Length, between perpendiculars (Lpp)	150.00 m	171.00 m
Breadth Molded	27.70 m	31.28 m
Design Draft	6.88 m	9.21 m
Main engine	5700 BHP	9000 HP
Kargo Tank	24,814 m3	42,955 m3
Kargo Pump	3 x 600 m3/h	3 x 1200 m3/h
Speed	13.2 knot	14 knot
Complement	28 persons	30 persons

Sumber: Operasi Perkapalan, 2010

3.3.2.2 Kapasitas Kapal

Besaran kargo Premium, Kerosin dan Solar yang diangkut oleh kapal menggunakan kebiasaan (*rule of thumb*) yang dijalankan oleh operasi tanker. Sebagai contoh kapal GP dengan kapasitas 17.500 LTDW, maka kargo yang diangkut sebesar = 7 5 5 = (7000 ton Premium, 5000 Ton Kerosin, 5000 ton solar). Kapal MR dengan kapasitas 30000 DWT, kargo yang diangkut sebesar = 10 9 10 = 10000 ton Premium, 9000 ton Kerosin, 10000 ton Solar). Pada kondisi aktual, pembagian kargo ini fleksibel tergantung dari kebutuhan depot. Perbandingan kapasitas kompartemen kapal GP dan MR dapat dilihat dari dibawah.

Tabel 3.2 Kapasitas Kompartemen Kapal GP dan MR

Kapal	Kompartemen (KL)		
	Premium	Kerosin	Solar
PGD (GP)	9651	5004	9651
GDN (MR)	14506	9056	18506

Sumber: Operasi Perkapalan, 2010

3.3.3 Data Waktu

3.3.3.1 Data Waktu Berlayar

Waktu Berlayar adalah waktu yang dibutuhkan kapal untuk berlayar dari pelabuhan pemberangkatan ke pelabuhan tujuan. Waktu Berlayar dihitung mulai dari waktu (*Actual Time Departure* , ATD) pelabuhan sebelumnya sampai dengan (*Actual Time Arrival*, ATA) pelabuhan berikutnya. Waktu Berlayar dapat dipengaruhi oleh kondisi lautan, dimana pada waktu-waktu tertentu (September – Desember) laut mempunyai ombak yang besar sehingga waktu pelayaran kapal sedikit bertambah. Kondisi muatan kapal (*apakah full load atau ballast condition*) jika dilihat lebih detail juga mempengaruhi kecepatan kapal yang pada akhirnya akan berpengaruh pada waktu berlayar.

Tabel 3.3 Data Waktu Berlayar Kapal GP (hari)

Pelabuhan	SBY	CLC	TWI	TTB	TTM	BPP	KUP	XPN
SBY		1.85	0.62	0.22	0.79	1.45	2.62	0.58
CLC	2.05		1.32		1.32		3.01	
TWI	0.78	0.5		1.09	0.33	1.2	2.21	0.26
TTB	0.22		1.03		1.12		2.92	
TTM	0.79	1.32	0.39	1.1		1.82	1.94	0.52
BPP	1.58		1.63		2.8		2.91	
KUP	2.79	3.13	2.21	3.01	1.94	2.85	0	2.26
XPN	0.54		0.22		0.87		2.23	

Sumber: Operasi Perkapalan, 2010

Tabel 3.4 Waktu Berlayar Kapal MR (hari)

Pelabuhan	CLC	TTB	XPN	BPP	SBY	TTM
CLC					2.5	1.34
TTB					0.3	1.09
XPN					0.55	0.69
BPP					1.58	1.95
SBY	2.22	0.19	0.52	1.68		0.79
TTM	1.6	0.95	0.44	1.49	0.79	

Sumber: Operasi Perkapalan, 2010

3.3.3.2 Waktu *Loading – Unloading*

Waktu *loading* adalah waktu yang dibutuhkan kapal ketika berada di pelabuhan muat, dimana menggunakan pompa pelabuhan untuk memompa muatan ke dalam kapal. Waktu dihitung mulai dari dipompanya muatan ke kapal (*Commence Loading*, CML) sampai selesai proses pemompaan (*Completed Loading*, CPL).

Waktu *unloading* adalah waktu yang dibutuhkan kapal ketika berada di pelabuhan bongkar, dimana mengunakan fasilitas pompa kapal. Waktu dihitung mulai dari dipompanya muatan ke pelabuhan bongkar (*Commence Loading*, CML) sampai selesai proses pemompaan (*Completed Loading*, CPL). Lama *un(loading)* tergantung dari jumlah muatan yang dipompa.

Tabel 3.5 Data (Un) *Loading Rate*

No	Pelabuhan	Un *(Loading) rate*
1	CLC	0.00003
2	TTB	0.00004
3	BPP	0.00005
4	XPN	0.00003
5	Kapal GP	0.0001
6	Kapal MR	0.00004

Sumber: Operasi Perkapalan, 2010

3.3.4 Data Persediaan Pengaman, *Alarm Level*, Stok maksimal, *Consumption rate* dan *Inventory* Awal

3.3.4.1 Persediaan Pengaman, *Alarm Level,* Stok maksimal, *Consumption rate*

Masing–masing pelabuhan bongkar memiliki batas stok minimum (*persediaan pengaman*) yang harus dijaga keberadaannya dan batas stok maksimal yang dapat dipenuhi. Selain itu untuk setiap pelabuhan ada *Alarm Level* sebagai pengingat bahwa stok sudah mendekati persediaan pengaman yang harus dijaga keberadaannya. Setiap pelabuhan bongkar juga mempunyai tingkat permintaan per hari yang berbeda-beda.

Tabel 3.6 Data Persediaan Pengaman, Alarm Level, Stok maksimal, *Consumption rate* dan Penalti

Pelabuhan, Produk		Persediaan Pengaman (KL)	*Alarm Level* (KL)	Stok Maksimal (KL)	J	*Consumption Rate* (KL/*day*)
SBY						
1	Premium	0	0	0	0	0
2	Kerosin	1,899.568	2,213.028	32,000	-1	313.46
3	Solar	0	0	0	0	0
TWI						
1	Premium	7,076.245	8,326.465	27,818	-1	1250.22
2	Kerosin	561.642	6,60.8718	9,566	-1	99.23
3	Solar	4,155.345	4,889.506	21000	-1	734.16
TTM						
1	Premium	8,206.282	9,760.5	34,500	-1	1,554.22
2	Kerosin	3,239.227	3,852.717	30,000	-1	613.49
3	Solar	17,070.029	20,302.99	57,500	-1	3,032.96
KUP						
1	Premium	1,562.19	2,082.92	12,166	-1	520.73
2	Kerosin	591.84	789.12	14,425	-1	197.28
3	Solar	1,933.56	2,578.08	17,122	-1	644.52

Sumber: Operasi Perkapalan, 2010

Keterangan j = -1 menunjukkan pelabuhan konsumsi,

0 tidak terdapat konsumsi ataupun produksi,

1 menunjukkan pelabuhan produksi.

3.3.4.2 Data *Inventory* Awal

Pada awal perencanaan, setiap pelabuhan bongkar memiliki *inventory* di setiap tangki produk seperti dalam tabel di bawah ini.

Tabel 3.7 Data *Inventory* awal

No.	Pelabuhan	*Inventory* Awal		
		Premium	Kerosin	Solar
1	SBY	0	3178.49	0
2	TWI	12239.65	971.46	7187.43
3	TTM	15697.97	6096.25	35952.90
4	KUP	4937.94	2174.03	8272.71

Sumber: Operasi Perkapalan, 2010

3.3.5 Data Biaya

3.3.5.1 Data Biaya Kunjungan Pelabuhan dan Pemakaian Bahan Bakar

Biaya Bahan Bakar adalah *bunker consumption* yang digunakan saat kapal berlayar yang diperoleh dari Waktu Berlayar (hari) x Biaya Bahan Bakar (Rp/Hari). Biaya kunjungan pelabuhan adalah biaya tetap yang dikenakan pada kapal tiap kali memasuki pelabuhan disepanjang rute.

Tabel 3.8 Data biaya kunjungan pelabuhan dan bahan bakar

Keterangan	**Jenis Kapal**	
	PGD	GDN
Biaya Pelabuhan (Rp/ Kunjungan)	19,080,220.00	27,580,643.13
Harga HFO (*Heavy Fuel Oil*) (Rp/ KL)	5,912,500	5,830,000
Pemakaian Bahan bakar (KL/hari)	13,265	25,132
Biaya Bahan Bakar (Rp/hari)	76,928,158	14,652,1343

Sumber: Operasi Perkapalan, 2010

3.3.5.2 Data Penalti

Biaya penalti ini dikenakan pada kapal jika datang saat stok dibawah *alarm level*. Biaya penalti yang dikenakan Rp 100,000 x Jumlah stok (KL).

3.3.6 Model Matematis

Berikut ini adalah variabel, parameter, dan fungsi tujuan yang digunakan dalam penyelesaian permasalah optimasi ini.

Variabel :

1. Variabel untuk aliran *network*

• x_{imjnv} : adalah variabel aliran = 1 jika kapal v berlayar dari (i,m) ke (j,n); jika tidak, 0.

• y_{im} : Bernilai 1 jika (i,m) tidak dikunjungi; jika tidak, 0.

2. Variabel untuk *loading* dan *unloading*

• l_{imvk} : Jumlah produk k dalam kompartemen kapal v ketika kapal meninggalkan pelabuhan (i,m).

• q_{imvk} : Jumlah produk k yang di (un)*loading* (dari)ke kompartemen kapal v saat di pelabuhan (i,m).

3. Variabel untuk aspek waktu

• o_{imvk} : Binary variabel 1 jika produk k di *loading* atau *discharge* di pelabuhan (i, m), jika tidak 0.

• t_{im} : Waktu awal *service* di pelabuhan (i,m).

• t_{Eim} : Waktu berakhirnya *service* di pelabuhan (i,m).

4. Variabel untuk *inventories*

• s_{imk} : Stok produk k ketika *service* dimulai di pelabuhan (i,m).

• s_{Eimk} : Stok level produk k ketika *service* berakhir di pelabuhan (i,m).

• a_{imk}^{-}: Stok level produk k di pelabuhan (i,m) yang berada di bawah alarm *inventory* level.

Parameter :

1. Parameter untuk aliran *network*

• i_v : Pelabuhan awal untuk kapal v.

• m_v : Urutan kedatangan kapal v di pelabuhan i_v.

2. Parameter untuk *loading* dan *unloading*

• J_k : sama dengan 1 jika pelabuhan produksi, atau -1 jika pelabuhan konsumsi, dan 0 jika produk k tidak di produksi ataupun di konsumsi pada pelabuhan j.

• Q_{vk} : Jumlah produk k di kapal v saat awal *planning horizon*.

• CAP_{vk} : Kapasitas kompartemen produk k di kapal v.

3. Parameter untuk aspek waktu

- TQ_{ik} : Waktu yang dibutuhkan untuk *load* atau *unload* satu unit produk k di pelabuhan i.
- T_{ijv} : Waktu yang dibutuhkan untuk berlayar dari pelabuhan i ke j dengan kapal v.

4. Parameter untuk inventori

- IS_{ik} : Stok awal produk k di pelabuhan i.
- R_{ik} : Rata-rata konsumsi atau produksi produk k di pelabuhan i.
- S_{MNik} : Minimum stok produk k di pelabuhan i.
- A_{MNik} : Jumlah persediaan pengaman produk k di pelabuhan i.
- S_{MXik} : Maksimum level stok produk k di pelabuhan i.
- T : Waktu Perencanaan

5. Parameter untuk biaya

- CS_{ijv} : Biaya berlayar kapal v dari pelabuhan i ke j.
- CF_{ijv} : Biaya pelabuhan kapal v di pelabuhan i dan pelabuhan j.
- Cp_{ik} : Biaya penalti produk k di pelabuhan i.

Himpunan :

- z : Himpunan Biaya
- S_T : Himpunan semua pelabuhan kedatangan (i,m) dimana i Є H_T dan m Є M_i.
- H_T : Himpunan semua pelabuhan
- M_i : Himpunan jumlah kedatangan di pelabuhan i
- S_0 : Himpunan posisi awal
- S_N : Himpunan semua posisi yang mungkin ditempati oleh kapal setelah meninggalkan posisi awal
- V : Himpunan semua kapal v
- A_v : Himpunan *feasible arc* untuk kapal v
- K : Himpunan produk k
- K_v : Himpunan produk yang dapat dibawa oleh kapal v
- K_i^H : Himpunan produk yang di pelabuhan i

Kendala :

1. **Kendala posisi awal:** Pada awal perencanaan, kapal *v* harus berangkat dari pelabuhan (baik pelabuhan muat maupun bongkar).

$$\sum_{(j,n)\varepsilon S_N} x_{i_v m_v jnv} = 1, \text{ untuk setiap kapal } v \varepsilon V \quad \text{.....(3.1)}$$

2. **Kendala draft:** Kapal *Medium Range* (MR) tidak dapat berlayar menuju pelabuhan KUP dan TWI.

$$\sum_{(j,n)\varepsilon S_N} x_{i_v m_v jnv} = 0, \text{ untuk kapal } v = 2 \text{ dan } j = 1, 2 \quad \text{.....(3.2)}$$

$$\sum_{(j,n)\varepsilon S_T} x_{imjnv} = 0, \text{ untuk kapal } v = 2 \text{ dan } j = 1, 2 \quad \text{.....(3.3)}$$

3. Kendala muatan awal produk k

Jumlah produk k dalam kapal v pada saat berangkat dari posisi awal $(i_v, m_v)_{i_v m_v vk}$ = jumlah muatan awal kapal Q_{vk} ditambah kargo q_{ivmvk} (yang dimuat $J_{ivk} = +1$ dan $Ji_vk = -1$ untuk bongkar.)

$$Q_{vk} + J_{ik}\, qi_v m_v vk = l_{i_v m_v vk}, \text{ untuk setiap } v \varepsilon V \text{ dan } k \varepsilon K_V \quad \text{.....(3.4)}$$

4. Kendala *inventory* awal di pelabuhan i

Pada saat awal perencanaan, pelabuhan yang tidak memiliki kapal, stok produk k di pelabuhan i pada kedatangan kapal pertama ($s_{i1k)}$ adalah jumlah produk k pelabuhan i pada awal perencanaan ($IS_{ik)}$ ditambah jumlah yang diproduksi ($Ji_vk = +1$) atau dikurangi jumlah yang dikonsumsi ($Ji_vk = -1$) sampai t_{i1} kedatangan kapal pertama.

$$S_{i1k} = IS_{ik} + J_{ik}\, R_{ik}\, t_{i1}, \text{ untuk setiap } (i,k) \varepsilon H \text{ x } K_i^H \quad \text{.....(3.5)}$$

5. Kendala aliran konservasi

Kedatangan ke m kapal v di pelabuhan i harus meninggalkan pelabuhan tersebut dan pergi kepelabuhan lain jika belum memenuhi waktu perencanaan atau mengakhiri rutenya disana jika itu merupakan posisi akhir jadwal kapal v (Z=1), dan Z=0 jika (i,m) posisi *intermediate.*

$$\sum_{(j,n)\varepsilon S_T} x_{jnimv} - \sum_{(j,n)\varepsilon S_N} x_{imjnv} - Z_{imv} = 0, \text{ untuk setiap } (v,i,m)\ \varepsilon\ V \text{ x } S_N \quad \text{.....(3.6)}$$

6. Kendala akhir rute

Di akhir periode perencanaan, setiap kapal berada di pelabuhan.

$$\sum_{(i,m)\ \varepsilon\ S_N} Z_{imv} = 1, \text{ untuk setiap } v\ \varepsilon\ V \quad \text{.....(3.7)}$$

7. Kendala satu kali kunjungan

Setiap *port call* (i,m) dikunjungi paling banyak sekali, y_{im} adalah *binary variable* = 1 jika posisi (i,m) tidak dikunjungi .

$$\sum_{v \varepsilon V} \sum_{(j,n) \varepsilon S_T} x_{jnimv} + Y_{im} = 1, \text{untuk setiap } (i,m) \varepsilon S_T \text{.....(3.8)}$$

8. Kendala urutan kedatangan

Pelabuhan i memiliki kedatangan *m* secara pasti memiliki kedatangan *(m-1)*

$$Y_{im} - Y_{i(m-1)} \geq 0, \text{untuk setiap } (i,m) \varepsilon S_N \text{.....(3.9)}$$

9. Kendala muatan kapal

Jumlah produk k pada kapal saat meninggalkan (j,n) adalah Jumlah produk pada kapal saat berangkat dari (i,m) ditambah kargo yang dimuat atau dikurangi muatan yang dibongkar di j,n. x_{imjnv} =1 jika kapal v berlayar dari (i,m) ke (j,n).

$$x_{imjnv} [l_{imnk} + J_{jk} q_{jnvk} - l_{jnvk}] = 0, \text{untuk setiap } v \varepsilon V \text{ dan } (i,m,j,n,k) \varepsilon A_v \text{ x } K_v \text{.....(3.10)}$$

10. Kendala kapasitas kompartemen

Jumlah produk k pada kapal v saat berangkat tidak boleh melebihi kapasitas kompartemen CAPvk yang diperuntukkan untuk produk k. Tapi, ini berlaku jika kapal v mengunjungi (i,m) yaitu $\sum_{(j,n) \varepsilon S_T} x_{i_v m_v jnv} = 1$, jika tidak maka $l_{imvk} = 0$

$$l_{imvk} \leq \sum_{(j,n) \varepsilon S_T} CAP_{vk} \, x_{jnimv}, \text{untuk setiap } v \varepsilon V \text{ dan } (k,i,m) \varepsilon K_v \text{ x } S_N \text{.....(3.11)}$$

11. Kendala pelayanan produk :

a. Pelabuhan *Unloading*

Jumlah produk k yang di-*unloading* dari kapal v pada pelabuhan (i,m) tidak akan melebihi kapasitas kompartemen CAPvk kapal v untuk produk k.

$$q_{imvk} \leq CAP_{vk} \, o_{imvk}, \text{untuk setiap } v \varepsilon V \text{ dan setiap } (k,i,m) \varepsilon K_v \text{ x } S_T \text{.....(3.12)}$$

b. Pelabuhan *Loading*

Jumlah produk k yang di-*loading* ke kapal v pada pelabuhan (i, m) harus sama dengan kompartemen CAPvk kapal v untuk produk k.

$$q_{imvk} = CAP_{vk} \, o_{imvk}, \text{untuk setiap } v \varepsilon V \text{ dan setiap } (k,i,m) \varepsilon K_v \text{ x } S_T \text{.....(3.13)}$$

D. Kendala untuk aspek waktu

12. Kendala urutan waktu pelayanan

Minimum jeda waktu antara kedatangan kapal t_{im} dengan kedatangan kapal sebelumnya $t_{i(m-1)}$ lebih besar dari 0.1

$$t_{im} - t_{i(m-1)} > 0.1, \text{untuk setiap } (i,m) \varepsilon S_N \quad \text{.....(3.14)}$$

13. Kendala Akhir Pelayanan

Waktu keberangkatan kapal disuatu pelabuhan, t_{Eim} = waktu kedatangan t_{im} ditambah waktu yang diperlukan untuk layanan semua produk

$$\sum_{v \varepsilon V} \sum_{k \varepsilon K_v} TQ_{ik}\, q_{imvk}$$

$$t_{im} + \sum_{v \varepsilon V} \sum_{k \varepsilon K_v} TQ_{ik}\, q_{imvk} - t_{Eim} = 0, \text{untuk setiap } (i,m) \varepsilon S_T \quad \text{.....(3.15)}$$

14. Kendala kesesuaian rute dan jadwal

Jika kapal v berlayar dari posisi (i,m) ke (j,n) dimana $x_{imjnv = 1,}$ maka waktu kedatangan di (j,n), t_{jn}, adalah jumlah waktu keberangkatan dari (i,m) setelah kapal melaksanakan *service* t_{Eim}, dan waktu perjalanan dari pelabuhan i ke pelabuhan j dengan kapal v, T_{ijv}.

$$x_{imjnv}\,[t_{Eim} + T_{ijv} - t_{jn}] \le 0, \text{untuk setiap } v \varepsilon V \text{ dan } (i,m,j,n) \varepsilon A_v \quad \text{.....(3.16)}$$

E. Kendala untuk *inventory*

15. Kendala level *Inventory*

Stok level saat selesai *service* ($s_{Eim)}$ sama dengan tingkat level (s_{imvk}) sebelum kapal v datang dikurangi jumlah q_{imvk} yang di-*loading*kan atau ditambah jumlah q_{imvk} yang dibongkar atau ditambah jumlah yang diproduksi saat kapal v sedang dimuat atau dikurangi jumlah yang dikonsumsi saat kapal v bongkar dengan rata-rata R_{ik}, selama periode waktu t_{Eim}-t_{im}.

$$S_{imk} - \sum_{v \varepsilon V} j_{ik}\, q_{mvk} + J_{ik}\, R_{ik}\, (t_{Eim} - t_{im}) - S_{Eimk} = 0, \text{untuk setiap } (i,m,k) \varepsilon S_T x\, K_i^H \text{..(3.17)}$$

Inventory saat kapal datang

$$S_{Ei(m-1)k} + J_{ik}\, R_{ik}\, (t_{im} - t_{Ei(m-1)}) - S_{imk} = 0, \text{untuk setiap } (i,m,k) \varepsilon S_T x\, K_i^H \text{....(3.18)}$$

16. Kendala stok level selama *planning horizon*

Di setiap posisi (i,m) stok produk k harus berada diantara level minimum s_{MNik} dan maksimum S_{MXik}.

$$S_{MNik} \le S_{imk} \le S_{MXik}, \text{untuk setiap } (i,m,k) \varepsilon S_T \text{ x } K_i^H \text{.....(3.19)}$$

$$S_{MNik} \le S_{Eimk} + J_{ik}\, R_{ik} (T - T_{Eim}) \le S_{MXik}, \text{untuk setiap } (i,m,k) \varepsilon S_T \text{ x } K_i^H \text{...(3.20)}$$

17. Batasan *Stock Level*

Untuk pelabuhan *unloading*, stok level pelabuhan saat kapal datang ($S_{imk)}$ harus berada diatas persediaan pengaman ($A_{MNi}k$)

$$S_{imk} + a^{-}_{imk} \geq A_{MNik}, \text{untuk setiap } (i,m,k) \varepsilon S_T \text{ x } K_i^H \text{.....(3.21)}$$

Persediaan pengaman dikurangi minimum stok akan menghasilkan jumlah *inventory* dibawah persediaan pengaman.

$$a^{-}_{imk} \leq A_{MNik} - S_{MNik}, \text{untuk setiap } (i,m,k) \varepsilon S_T \text{ x } K_i^H \text{.....(3.22)}$$

F. Kendala Tanda dan Keutuhan

$$t_{im}, t_{Eim} \leq T$$

$$x_{imjnv}, y_{im}, o_{imvk} \varepsilon \{0,1\}$$

$$l_{imvk}, q_{imvk}, S_{imk}, S_{Eimk}, a^{-}_{im}, T \geq 0$$

Fungsi Tujuan :

Total Biaya = Biaya Berlayar (CS_{ijv}) + Biaya pelabuhan (CF_{ijv}) + Biaya Penalti (C_{pik}^{-})

$$\text{Min } Z = \sum_{v \varepsilon V} \sum_{(i,m,j,n) \varepsilon A_v} CS_{ijv} \, x_{imjnv} + \sum_{v \varepsilon V} \sum_{(i,m,j,n) \varepsilon A_v} CF_{ijv} \, x_{imjnv} + \sum_{v \varepsilon V} \sum_{(i,m,j,n) \varepsilon A_V} C^{-}_{Pik} \, a^{-}_{imk}$$

BAB 4
PENGOLAHAN DATA DAN ANALISIS

Optimasi rute dan jadwal kapal menggunakan algoritma *Tabu Search* akan dibuat perangkat lunak MATLAB versi 7.9.0 (2009b). MATLAB merupakan bahasa komputasi yang sangat baik dalam mengolah data dalam bentuk vektor dan matriks. Data akan dibuat dalam bentuk matriks – matriks sehingga bisa diproses dengan mudah.

4.1 Verifikasi dan Validasi Program

Sebelum program digunakan untuk pengolahan data, perlu dilakukan verifikasi dan validasi terhadap program.

4.1.1 Verifikasi Program

Verifikasi merupakan tahapan untuk melihat kesesuaian model dari program yang dibuat (tahap mengolah data dengan me-*run* program dengan aturan yang telah dibuat). Beberapa proses verifikasi yang dilakukan dalam proses pembuatan program ini:

- Memastikan kebenaran logika pemikiran, yaitu kesesuaian penulisan *source code* dengan konsep algoritma *Tabu Search* , semua batasan masalah dan fungsi tujuan
- Melakukan perubahan nilai parameter kontrol dan jika outputnya yang dihasilkan berubah-ubah maka program telah terverifikasi. Parameter kontrol yang diubah meliputi jumlah solusi tetangga, panjang iterasi, dan panjang tabu list.

Pada awalnya, dilakukan percobaan dengan jumlah solusi tetangga antara 50-200 dan parameter lain dibuat tetap dengan panjang *tabu list* = 10, iterasi = 10, diperoleh hasil percobaan dengan mengubah jumlah solusi tetangga seperti tabel berikut :

Tabel 4.1 Hasil Percobaan Jumlah Solusi Tetangga

Jumlah Solusi Tetangga	Biaya Optimal	Waktu Komputasi (detik)

50	5.70E+09	15.714358
100	5.66E+09	17.977124
150	4.41E+09	19.676298
200	4.31E+09	20.751833
250	4.81E+09	21.494511

Dengan menggunakan nilai panjang *tabu list* = 10, jumlah solusi tetangga = 200 dan jumlah iterasi yang diubah - ubah (10 – 50) maka diperoleh perbandingan biaya dan waktu komputasi sebagai berikut.

Tabel 4.2 Hasil Percobaan Jumlah Iterasi

Jumlah iterasi	Biaya Optimal	Waktu Komputasi (detik)
10	4.81E+09	15.635146
20	4.72E+09	16.311781
30	4.6600E+09	15.027617
40	4.6579E+09	17.693318
50	4.71E+09	13.199473

Dengan menggunakan nilai panjang *tabu list* = 10, jumlah solusi tetangga = 200 dan panjang tabu list yang diubah – ubah (10-15) maka diperoleh perbandingan biaya dan waktu komputasi sebagai berikut.

Tabel 4.3 Hasil Percobaan Panjang Tabu *List*

Panjang *tabu list*	Biaya Optimal	Waktu Komputasi (detik)
5	4.6958E+09	15.22744
10	4.6557E+09	14.779326
15	5.659E+09	17.851993

Dari proses verifikasi diatas, terlihat bahwa dengan mengubah nilai parameter, maka output pun berubah maka dinyatakan bahwa program telah terverifikasi. Setelah proses verifikasi selesai, dilanjutkan dengan melakukan validasi program.

4.1.2 Validasi Program

Validasi adalah membandingkan pengerjaan program dan pengerjaan secara manual. Validasi program yang dilakukan adalah dengan menggunakan data *dummy* dari permasalahan yang lebih sederhana. Data *dummy* yang digunakan adalah sebagai berikut.

Tabel 4.4 Data *Dummy* Waktu Berlayar (hari)

Pelabuhan	U1	L2	U3	U5	L4	U7
U1	0	0.15	0.27	0.19	0.2	0.13
L2	0.1	0	0.13		0.17	0.18
U3	0.2	0.13	0	0.2	0.13	0.14
U5	0.2		0.15	0	0.21	0.15
L4	0.12	0.14	0.1	0.19	0	0.13
U7	0.18	0.17	0.11	0.15	0.18	0

Keterangan : L = Pelabuhan *Loading*, U = Pelabuhan *Unloading*. Tidak ada pergerakan langsung antar pelabuhan *loading*.

Tabel 4.5 Data *Dummy* Inventori dan Kargo selama *Planning Horizon* T = 3 hari

Port.	Produk	Waktu inventori habis = (Inventori awal-Persediaan pengaman)/(*Consumption Rate*)	Minimum tambahan kargo yang memenuhi kebutuhan selama *Planning Horizon* = rata-rata demand * (Planning Horizon-waktu habisnya inventori)
U1	P	(10-0.5)/5 = 1.9	5x(3-1.9) = 5.5
	K	(10-0.5)/5 = 1.9	5x(3-1.9) = 5.5
	S	(15-0.5)/10 = 1.45	10x(3-1.45) = 15.5
U3	P	(10-0.5)/5 = 1.9	5x(3-1.9) = 5.5
	K	(10-0.5)/5 = 1.9	5x(3-1.9) = 5.5
	S	(10-0.5)/5 = 1.9	5x(3-1.9) = 5.5
U5	P	(15-0.5)/10= 1.45	10x(3-1.9) = 15.5
	K	(20-0.5)/10 = 1.95	10x(3-1.9) = 10.5
	S	(25-0.5)/10 = 2.45	10x(3-1.9) = 5.5
U7	P	(20-0.5)/10 = 1.95	10x(3-1.9) = 10.5
	K	(10-0.5)/5 = 1.9	5x(3-1.9) = 5.5
	S	(20-0.5)/10 = 1.95	10x(3-1.95) = 10.5

Tabel 4.6 Data *Dummy* Inventori Awal, Kecepatan *Loading*, Persediaan Pengaman, *Alarm Level*, *Consumption Rate*

Port	Kargo	Inventori Awal	Kec. *Loading*	Persediaan pengaman	*Alarm level*	*Consumption Rate* (hari)
U1	P	10	0.01	0.5	5	5
	K	10	0.01	0.5	10	5
	S	15	0.01	0.5	15	10
U3	P	10	0.02	0.5	10	5
	K	10	0.02	0.5	15	5
	S	10	0.02	0.5	10	5
U5	P	15	0.01	0.5	5	10
	K	20	0.01	0.5	10	10
	S	25	0.01	0.5	15	10
U7	P	20	0.02	0.5	10	10
	K	10	0.02	0.5	15	5

Konfigurasi parameter yang digunakan untuk validasi program ini adalah seperti tabel dibawah ini.

Tabel 4.7 Data *Dummy* Konfigurasi Parameter yang Digunakan dalam Validasi

Parameter	Nilai
Jumlah solusi tetangga	3
Panjang tabu list (T*list*)	2
Maksimum iterasi (*maxit*)	1

Dengan biaya berlayar = Rp 1/hari, Biaya kunjungan pelabuhan =Rp 1, Biaya penalti = Rp 0.3. Hasil *run* program dengan menggunakan data *dummy* dan konfigurasi parameter diatas menunjukkan solusi rute urutan perjalanan adalah Kapal 1 dengan rute L2-U1-U3,dan Kapal 2 dengan rute L4-U5-U7 dengan total biaya Rp.12,755. Kemudian dilakukan perhitungan manual dengan mengikuti langkah kerja program, yaitu skenario pemilihan acak untuk mendapatkan solusi tetangga. Iteasi perhitungan manual yang dilakukan adalah sebagai berikut.

1. Pembentukan Solusi Awal

   Pencarian solusi awal dilakukan secara *random* (acak) dan diperoleh urutan rute seperti dibawah ini :

Tabel 4.8 Solusi Awal Algoritma *Tabu Search*

V	K	l_{imvk}	Awal		l_{imvk}	Tujuan		l_{imvk}	Tujuan		l_{imvk}
			t_{im} (d)	t_{Eim}(d)		t_{im} (d)	t_{Eim}(d)		t_{im} (d)	t_{Eim}(d)	
			Pel. U2			**Pel. U1**			**Pel. U3**		
A	P	0	0	0.86	26	0.96	1.2	14.5	1.4	1.8	9.5
	K	0	0		30			24.5			19
	S	0	0		30			14.5			9
			Pel. L4			**Pel. U5**			**Pel. U7**		
B	P	0	0	0.86	26	1.07	1.4	10.5	1.5	2	0
	K	0	0		30			14.5			9.5
	S	0	0		30			24.5			4

Perhitungan biayanya adalah sebagai berikut :

Tabel 4.9 Perhitungan Biaya

V	K	Awal	Tujuan			Tujuan			Total
			CS_{ijv} (Rp)	**CF_{ijv} (Rp)**	**C_{Pik} (Rp)**	**CS_{ijv} (Rp)**	**CF_{ijv} (Rp)**	**C_{Pik} (Rp)**	
			Pel. U1			**Pel.U3**			
A	P	**Pel. L2**	**0.1**	**1**	0.09	**0.27**	**1**	0.8025	**4.8825**
	K				0.09			0	
	S				1.53			0	
			Pel.U5	**Total**	**1.71**	**Pel.U7**	**Total**	**0.8025**	
B	P	**Pel.L4**	**0.21**	**1**	1.86	**0.13**	**1**	1.695	**7.8725**
	K				0.36			0.9225	
	S				0			1.695	
				Total	**2.22**		**Total**	**4.3125**	**Rp.12.755**

2. Pencarian Solusi yang Optimal

   - Solusi Terbaik saat ini yaitu : Kapal A: **Pelabuhan L2 – Pelabuhan U1 – Pelabuhan U3** dan Kapal B : **Pelabuhan L4-Pelabuhan U5-Pelabuhan U7** dengan biaya **Rp. 12.755**

- Bentuk 3 solusi tetangga dengan menggunakan struktur solusi tetangga secara acak. Struktur pembentukan solusi tetangga menggunakan metode N_{change}.

- **Solusi tetangga 1 (xn_1): N_{change} U5 – U1**

  Rute Kapal A menjadi : Pelabuhan L2 - U5 - U3, dan Kapal B menjadi : Pelabuhan 4 - U1 - U7

Tabel 4.10 Solusi tetangga 1 (xn_1): N_{change} U5 – U1

V	K	l_{imvk}	Awal		l_{imvk}	Tujuan		l_{imvk}	Tujuan		l_{imvk}
			t_{im} (d)	t_{Eim} (d)		t_{im} (d)	t_{Eim} (d)		t_{im} (d)	t_{Eim} (d)	
			Pel.L2			**Pel. U5**			**Pel. U3**		
A	P	0	0	0.86	26	1	1.35	10.5	1.445	1.78	5
	K	0	0		30			19.5			14
	S	0	0		30			24.5			19
			Pel.L4			**Pel. U1**			**Pel. U7**		
B	P	0	0	0.86	26	1.1	1.33	20.5	1.425	1.96	10
	K	0	0		30			24.5			19
	S	0	0		30			14.5			4

Tabel 4.11 Perhitungan Biaya Solusi tetangga 1 (xn_1): N_{change} U5 - U1

V	K	Awal	Tujuan			Tujuan			Total
			CS_{ijv} (Rp)	CF_{ijv} (Rp)	C_{Pik} (Rp)	CS_{ijv} (Rp)	CF_{ijv} (Rp)	C_{Pik} (Rp)	
			Pel. U5			**Pel.U3**			
A	P	**Pel. L2**	**0.17**	**1**	0.65	**0.27**	**1**	2.725	**17.715**
	K				0.65			2.725	
	S				5.8			2.725	
			Pel.U1	**Total**	**7.1**	**Pel.U7**	**Total**	**8.175**	
B	P	**Pel.L4**	**0.2**	**1**	6.1	**0.13**	**1**	4.75	**21.655**
	K				1.1			2.625	
	S				0			4.75	
				Total	**7.2**		**Total**	**12.125**	**39.37**

Dari hasil N_{change} U5 - U1, diperoleh Total Biaya sebesar Rp. 39,37.

- **Solusi tetangga 2 (xn_2): N_{change} U3 -U7**

Rute Kapal A menjadi : Pelabuhan L2-U1-U7 ,dan Kapal B menjadi : Pelabuhan L4-U5-U3.

Tabel 4. 12 Solusi tetangga 2 (xn_2): N_{change} U3 - U7

V	K	l_{imvk}	Awal		l_{imvk}	Tujuan		l_{imvk}	Tujuan		l_{imvk}
			t_{im} (d)	t_{Eim} (d)		t_{im} (d)	t_{Eim} (d)		t_{im} (d)	t_{Eim} (d)	
			Pel. L2			**Pel. U1**			**Pel. U7**		
A	P	0	0	0.86	26	0.96	1.23	20.5	1.355	1.89	10
	K	0	0		30			24.5			19
	S	0	0		30			14.5			4
			Pel. L4			**Pel. U5**			**Pel. U3**		
B	P	0	0	0.86	26	1.07	1.39	10.5	1.485	1.82	5
	K	0	0		30			19.5			14
	S	0	0		30			24.5			19

Tabel 4.13 Perhitungan Biaya Solusi tetangga 2 (xn_2): N_{change} U3 - U7

V	K	Awal	Tujuan			Tujuan			Total
			CS_{ijv} (Rp)	CF_{ijv} (Rp)	C_{Pik} (Rp)	CS_{ijv} (Rp)	CF_{ijv} (Rp)	C_{Pik} (Rp)	
			Pel. U1			**Pel.U7**			
A	P	**Pel. L2**	**0.1**	**1**	0.09	**0.13**	**1**	4.05	**14.315**
	K				0.09			2.275	
	S				1.53			4.05	
			Pel.U5	**Total**	**1.71**	**Pel.U3**	**Total**	**10.375**	
B	P	**Pel. L4**	**0.21**	**1**	6.2	**0.1**	**1**	2.925	**18.485**
	K				1.2			2.925	
	S				0			2.925	
				Total	**7.4**		Total	**8.775**	**32.8**

Dari hasil N_{change} U3 – U7, diperoleh Total Biaya sebesar Rp 32,8.

- **Solusi tetangga 3 (xn_3): N_{change} L2 - L4**

Rute Kapal A menjadi : Pelabuhan L4-U1-U3 ,dan Kapal B menjadi : Pelabuhan L2-U5-U7.

Tabel 4.14 Solusi tetangga 3 (xn_3): N_{change} L2 - L4

V	K	l_{imvk}	Awal		l_{imvk}	Tujuan		l_{imvk}	Tujuan		l_{imvk}
			t_{im} (d)	t_{Eim} (d)		t_{im} (d)	t_{Eim} (d)		t_{im} (d)	t_{Eim} (d)	
			Pel. L4			**Pel. U1**			**Pel. U3**		
A	P	0	0	**0.86**	26	**1.06**	**1.33**	20.5	**1.455**	**1.79**	15
	K	0	0		30			24.5			19
	S	0	0		30			14.5			9
			Pel.L2			**Pel. U5**			**Pel. U7**		
B	P	0	0	**0.86**	26	**1.03**	**1.35**	10.5	**1.445**	**1.98**	0
	K	0	0		30			19.5			14
	S	0	0		30			24.5			14

Tabel 4.15 Perhitungan Biaya Solusi tetangga 3 (xn_3): N_{change} L2 - L4

V	K	Awal	Tujuan			Tujuan			Total
			CS_{ijv} (Rp)	CF_{ijv} (Rp)	C_{Pik} (Rp)	CS_{ijv} (Rp)	CF_{ijv} (Rp)	C_{Pik} (Rp)	
			Pel. U1			**Pel. U3**			
A	P	**Pel. L2**	**0.2**	**1**	0.24	**0.13**	**1**	2.775	**12.965**
	K				0.24			2.775	
	S				1.83			2.775	
			Pel. U5	**Total**	**2.31**	**Pel. U7**	**Total**	**8.325**	
B	P	**Pel.L4**	**0.17**	**1**	5.8	**0.1**	**1**	4.95	**21.495**
	K				0.8			2.725	
	S				0			4.95	
				Total	**6.6**		**Total**	**12.625**	**34.46**

- Pilih satu solusi terbaik dari xn → xn_2, xn_2 = x sel
- Xsel = xit → biaya Rp.32.8, Kapal 1 : Pelabuhan L2-U1- U7, Kapal 2: L4-U5-U3
- Solusi xit tidak lebih baik dari xbest, maka solusi akhir adalah xbest, yaitu biaya : Rp. 12.755 dengan rute kapal (**Kapal 1 : Pelabuhan L2- U1- U3, Kapal 2 : L4- U5- U7).**

Dapat dilihat bahwa hasil perhitungan manual diatas sama dengan hasil dari run program. Dengan demikian program telah tervalidasi.

4.2 Penetapan Parameter Kontrol

Parameter kontrol di dalam penggunaan Algoritma *Tabu Search* adalah

- Jumlah solusi tetangga yang dihasilkan pada tiap iterasi
- Panjang *tabu list* (*Tlist*)
- Jumlah maksimum iterasi (*maxit*)

Penentuan nilai parameter ini adalah dengan mencoba me-*run* dengan nilai parameter yang berbeda-beda seperti halnya ditampilkan pada bagian. Verifikasi program.

Berdasarkan percobaan verifikasi program, diperoleh parameter *tabu list* = 10, solusi tetangga = 200 dan iterasi = 30 yang menghasilkan total biaya paling minimum sesuai dengan fungsi tujuan. Parameter ini yang akan digunakan untuk mencari solusi paling optimal dengan me-*run* kembali beberapa kali hingga menghasilkan biaya paling minimum.

4.3 Pengolahan Data

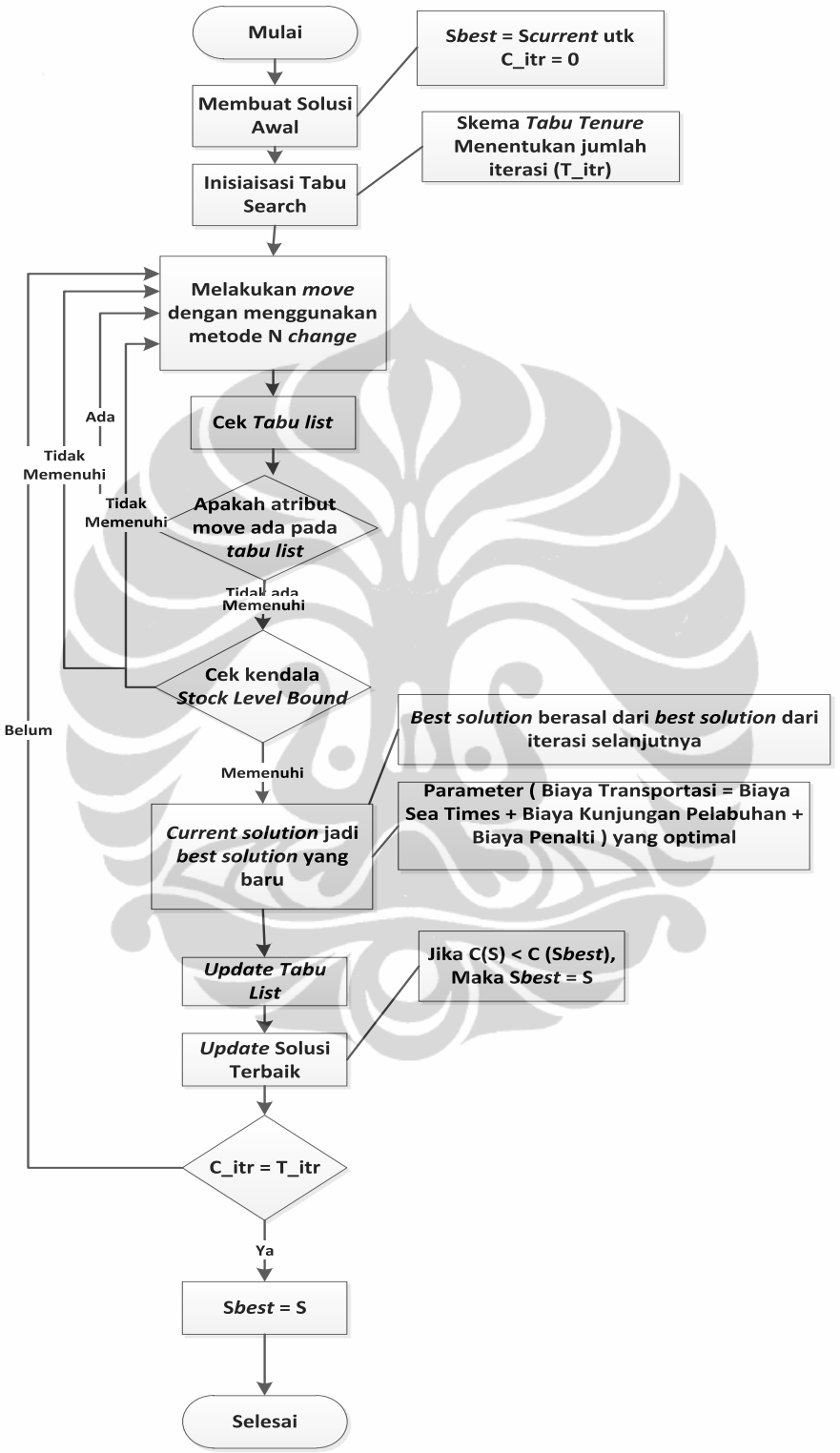
4.3.1 *Input* Data

Input data diperlukan untuk diolah dalam program. Data yang sudah di*input* dalam *spreadsheet* Ms.Excel dalam bentuk yang mudah dikonversikan ke dalam bentuk matriks untuk mempermudah pengolahannya dalam perangkat lunak MATLAB. Data yang di*input* adalah berupa data yang telah disebutkan dalam bab 3.

4.3.2 Langkah Pengerjaan Algoritma

Tahapan pertama adalah pembentukan solusi awal secara *random*. Solusi awal ini nantinya akan diperbaiki hingga menghasilkan solusi yang optimal melalui beberapa tahapan pengerjaan menggunakan algoritma *Tabu Search*. Program yang telah dirancang dalam perangkat lunak MATLAB versi 7.9.0 (2009b) ini, akan mencari sebuah solusi yang sudah memenuhi semua kendala ini pada saat pertama kali di-*run*. Solusi awal ini nantinya akan diolah menjadi solusi

yang optimal. Setelah solusi awal diperoleh, dilanjutkan dengan tahapan selanjutnya seperti yang ada dalam *flowchart* dibawah ini.



Gambar 4.1 Algoritma *Tabu Search*

Pengecekan kendala *stok level bound* pada Gb. 4.1 adalah pengecekan jumlah stok dimasing-masing pelabuhan tidak berada dibawah persediaan pengaman selama perencanaan.

Parameter yang digunakan dalam pemilihan solusi terbaik dari solusi saat ini digunakan total biaya transportasi yaitu jumlah dari biaya kapal berlayar, biaya pelabuhan, dan biaya penalti. *Source code* program Matlab ditampilkan pada lampiran.

4.3.3 Output

Output rute dan jadwal yang dihasilkan oleh Matlab terdapat di lampiran 2 dan 3.

4.4 Analisis

4.4.1 Analisis Algoritma

Pencarian solusi awal menggunakan metode random, hasilnya cukup baik. Mekanisme pembentukan solusi adalah N_{change}. N_{change} dilakukan dengan menukarkan pelabuhan *unloading* dengan *unloading* lainnya, dan pelabuhan *loading* dengan *loading* lainnya. Hal ini menghasilkan perbedaan output rute optimal pada setiap *run* program.

Skema *tabu tenure* yang digunakan adalah *fixed tabu tenure*. *Tabu tenure* adalah durasi status tabu atau lamanya waktu suatu solusi/atribut solusi dianggap tabu. *Tabu tenure* juga menunjukkan berapa panjang *tabu list (Tlist).* Skema *fixed tabu tenure* maksudnya panjang *tabu list* yang tetap sepanjang tahapan algoritma *Tabu Search* dijalankan. *Tabu list* yang digunakan di penelitian ini = 10, sehingga suatu solusi akan dianggap tabu selama sepuluh iterasi.

Kemungkinan terbentuknya solusi tetangga pada setiap iterasi sangat banyak karena pada saat dilakukan *change* antara pelabuhan satu dengan yang lainnya muatan yang dibongkar kapal pada pelabuhan secara otomatis akan berubah juga. Jadi rute yang sama pun, dapat menghasilkan muatan yang berbeda-beda. Namun, program ini dirancang untuk tidak menerima serangkaian rute yang sama tetapi muatan yang berbeda-beda pada saat mencari solusi tetangga pada iterasi yang sama. Hal ini bisa terjadi pada iterasi yang berbeda.

4.4.2 Analisis Studi Parameter

Dalam menentukan parameter, dilakukan percobaan seperti yang terdapat pada verifikasi program dan melihat dengan nilai parameter berapa paling berpengaruh pada waktu komputasi dan total biaya yang minimum dan tentunya memenuhi semua kendala. Nilai parameter yang menghasilkan biaya paling minimum dan memenuhi semua kendala yang ada, yang nantinya akan dipilih menjadi parameter kontrol.

4.4.3 Analisis Program

Program yang digunakan untuk menginterpretasikan algoritma *Tabu Search* adalah program yang dirancang dalam perangkat lunak Matlab. Secara umum, program yang dirancang untuk optimasi ini dapat menyelesaikan permasalahan penjadwalan kapal. Solusi akhir yang dihasilkan juga baik dibandingkan solusi awal. Hal ini dapat dilihat dengan total biaya yang dihasilkan lebih kecil.

Source code yang digunakan dalam program ini harus dipastikan benar dalam mengkonversikan *constraint* dan tahapan algoritma *Tabu Search* ke dalam bahasa Matlab. Kemudahan dalam mengkonversikan permasalahan ke dalam bahasa Matlab didukung juga oleh bagaimana bentuk data yang di*input* dalam *spreadsheet* Ms. Excel.

Output dihasilkan oleh program ini, berupa serangkaian pelabuhan yang dikunjungi tiap kapal. Kadang kala pada solusi yang optimal, muncul 2 pelabuhan secara berurutan pada rute kapal yang sama. Hal ini terjadi karena mekanisme pencarian solusi tetangga menggunakan N_{change} untuk menukarkan pelabuhan *unloading* dengan *unloading* lainnya, *loading* dengan *loading*, sehingga rentan akan munculnya 2 pelabuhan secara berurutan. Meskipun demikian, hal ini tidak menyebabkan terjadinya pencatatan ulang biaya dan waktu berlayar.

Pada program dirancang sebuah perintah tambahan untuk berhenti ketika selama pencarian rute yang optimal ketersediaan stok berada dibawah persediaan pengaman. Program akan secara otomatis berhenti lalu *user* me-*run* kembali untuk kembali mencari rute yang optimal dengan menjaga keberadaan persediaan pengaman.

Melihat beberapa hal yang harus diperhatikan ketika menggunakan program ini, *user* harus memperhatikan dengan teliti hasil yang benar-benar optimal. Jika terjadi kedua hal diatas, *user* perlu melakukan *run* kembali untuk mencari solusi yang benar-benar optimal.

4.4.4 Analisis Hasil Optimasi

4.4.4.1 Analisis Rute

Rute dan penjadwalan kapal di Pertamina dilaksanakan berdasarkan nilai permintaan dan dikombinasikan dengan pengalaman dari para perencana. Karena luasnya cakupan distribusi, beragamnya kebutuhan di tiap daerah, dan bervariasinya tipe kapal yang dioperasikan, distribusi bahan bakar ini sering mengalami kendala baik yang disebabkan titik suplai, depot utama maupun kurangnya koordinasi dan komunikasi. Oleh karena itu, rute usulan yang dihasilkan dapat digunakan sebagai acuan dalam mendistribusikan bahan bakar yang dibahas dalam penelitian ini. Selain biaya yang dihasilkan cukup optimal, dapat diperoleh dengan waktu yang relatif singkat dalam pencarian solusi yang optimal.

Pada penyelesaian permasalahan penjadwalan ini, kapal MR dikeluarkan dari model dan *dedicated* untuk memenuhi tingkat konsumsi pelabuhan TTM. Hal ini dikarenakan kapal MR yang ukuran dan kapasitasnya besar dan hanya bisa masuk ke pelabuhan SBY dan TTM. Pelabuhan SBY hanya mempunyai tangki Kerosin untuk dipenuhi dan jumlah konsumsi per harinya sangat sedikit, bila digunakan kapal MR untuk memenuhinya, biayanya akan lebih mahal. Berbeda halnya dengan konsumsi TTM yang sangat besar jumlah per harinya sehingga kapal MR cukup untuk memenuhi kebutuhan di pelabuhan ini. Selain itu, kapasitas kapal MR juga dapat mencukupi kebutuhan pelabuhan TTM. Pelabuhan CLC atau BPP dapat menjadi pilihan untuk menyuplai kebutuhan di pelabuhan TTM karena memiliki jarak yang cukup dekat dan memiliki tangki 3 produk yang dapat memenuhi kebutuhan pelabuhan ini. Karena kapal MR *dedicated* untuk pelabuhan TTM, maka pergerakan kapal MR ke pelabuhan dilakukan bongkar penuh.

Program dijalankan pada penjadwalan kapal selama ± 30 hari dengan parameter yang telah ditentukan. Penjadwalan 30 hari dilakukan dengan sekali *running* tanpa terpisah. Hal inilah yang menjadi kelebihan dari penggunaan algoritma *Tabu Search* sebagai penyelesaian permasalahan. Pada penelitian sebelumnya penyelesaian permasalahan ini dengan menggunakan *Branch and Bound* dengan perangkat lunak Lingo (Slamet,2011), perencanaan lebih dari 15 hari dilakukan secara terpisah karena keterbatasan variabel integernya. Berikut adalah hasil dari penjadwalan menggunakan algoritma *Tabu Search* dan Lingo.

✓ Penjadwalan = 15 Hari

Untuk penjadwalan perjalanan kapal selama 15 hari, pada awal perencanaan kapal 1 berada di pelabuhan BPP, kapal 2 berada di CLC. muatan kapal kondisinya kosong.

Kapal GP1 (PGD) menempuh rute BPP-SBY-KUP-TWI-TTB-SBY-TWI

Kapal GP2 (PGD) menempuh rute CLC-TWI-XPN-SBY-TTB-KUP-CLC

Kapal MR (GDN) menempuh rute TTM-CLC-TTM-BPP-TTM-CLC

Kapal pertama melakukan muat pada 2 pelabuhan yaitu BPP, TTB dan melakukan bongkar pada pelabuhan SBY, KUP, TWI. Pelabuhan SBY dan TWI dilayani sebanyak 2 kali. Hal ini disebabkan karena kapal membongkar sedikit muatan pada KUP dan butuh sering dikunjungi kapal untuk memenuhi kebutuhan KUP dengan menjaga persediaan pengaman di pelabuhan tetap terjaga. Sama halnya dengan kapal kedua yang melakukan muat pada 3 pelabuhan di pelabuhan yaitu CLC, XPN, TTB untuk memenuhi kebutuhan pelabuhan *unloading* TWI, KUP, SBY. Kedua kapal hanya mengunjungi pelabuhan *unloading* KUP sebanyak dua kali. Hal ini dikarenakan, KUP terletak sangat jauh di wilayah yang berbeda dari pelabuhan yang lain.

Rute yang dihasilkan oleh LINGO adalah sebagai berikut :

Kapal GP (PGD1) menempuh rute BPP-SBY-TWI-TTB-SBY-TWI-TTB-TWI-SBY-BPP-KUP

Kapal GP (PGD2) menempuh rute KUP-BPP-KUP-TWI-BPP-KUP

Kapal MR (GDN) menempuh rute CLC-TTM-TTB-TTM-TWI-CLC-TTM-SBY-TTB

Penyelesaian menggunakan Lingo dan Algoritma *Tabu Search* menghasilkan rute yang berbeda. Posisi awal kapal untuk kapal GP (PGD2) dan MR pada awal perencanaan letaknya berbeda. Kapal GP (PGD2) berangkat dari pelabuhan *unloading* yang akan berangkat ke pelabuhan loading untuk melakukan muat. Sedangkan penyelesaian menggunakan Algoritma, hasil optimal yang diperoleh, kapal GP (PGD2) berada pada pelabuhan muat dan akan segera memuat kargo pada kapal. Kapal GP (PGD1) mengunjungi lebih banyak pelabuhan dibandingkan kapal yang lain. Selain itu, pada hasil yang ditampilkan Lingo terdapat perbedaan beberapa rute perjalanan kapal.

✓ Model T= 30 hari

Posisi kapal pada saat awal perencanaan 30 hari sama dengan posisi kapal pada perencanaan 15 hari.

- Kapal GP (PGD1) menempuh rute BPP-SBY-KUP-TWI-TTB-SBY-TWI-CLC-SBY-TWI-TTB-KUP-TWI-CLC
- Kapal GP (PGD2) menempuh rute CLC-TWI-XPN-SBY-TTB-KUP-CLC-KUP-CLC-TWI-SBY-KUP
- Kapal MR (GDN) menempuh rute TTM-CLC-TTM-BPP-TTM-CLC-TTM-BPP-TTM

Kapal GP (PGD1) melakukan perjalanan sebanyak 5 kali ke pelabuhan loading. Pelabuhan CLC memiliki frekuensi kedatangan paling banyak dibandingkan pelabuhan *loading* lainnya karena tempatnya yang tidak terlalu jauh dari tiap pelabuhan *unloading*. Kapal GP (PGD2) mengunjungi pelabuhan XPN ketika pelabuhan bongkar seperti SBY membutuhkan solar. Pelabuhan SBY merupakan pelabuhan yang paling banyak dikunjungi kapal .

4.4.4.2 Analisis Persediaan

Dari hasil optimasi rute dan jadwal, diperoleh stok pelabuhan untuk masing – masing pelabuhan bongkar selama ± 30 hari perjalanan seperti di bawah ini.

Keterangan :

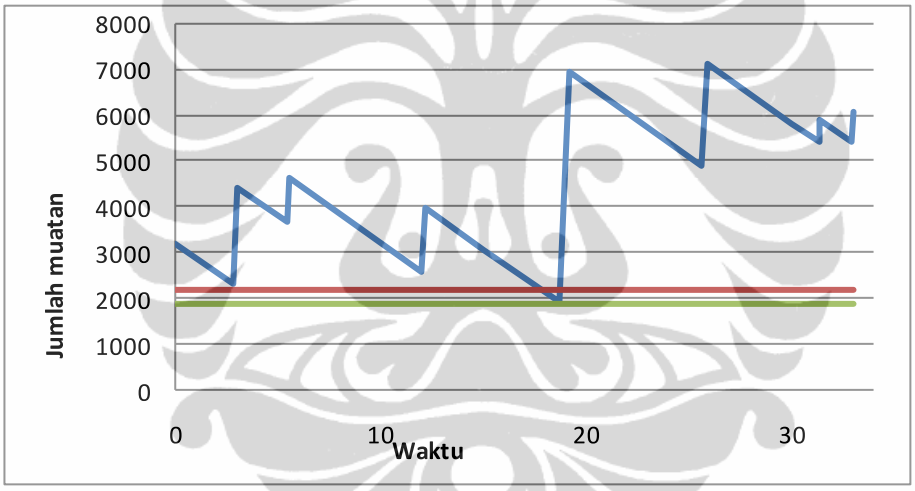
Jumlah produk di Pelabuhan

Persediaan pengaman

Alarm level

a. Pelabuhan SBY

Berikut adalah grafik Stok Kerosin di pelabuhan SBY sepanjang 30 hari.

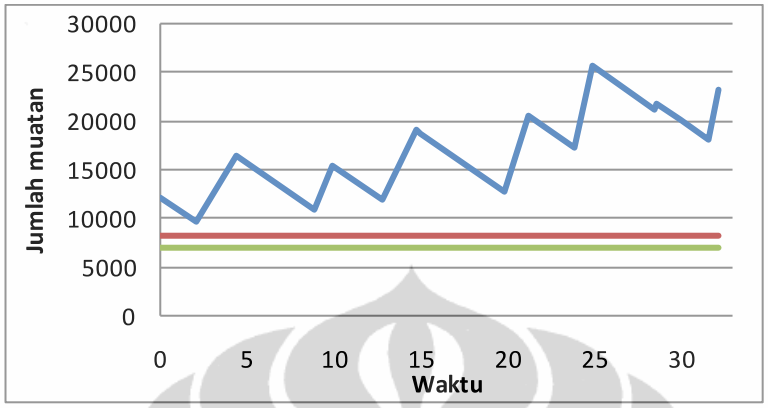


Gambar 4.2 Persediaan Kerosin pada Pelabuhan SBY

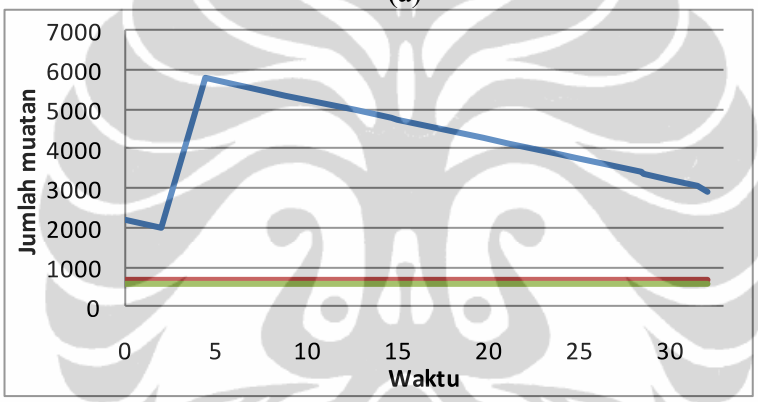
Fluktuasi jumlah stok di setiap pelabuhan bongkar di pengaruhi oleh intensitas kedatangan kapal melakukan bongkar di pelabuhan tersebut dan banyaknya muatan yang dibongkar. Dari Gambar 4.2 dapat dilihat bahwa pelabuhan SBY sering dikunjungi kapal untuk melakukan aktivitas bongkar Kerosin. Muatan yang paling banyak dibongkar untuk pelabuhan ini terjadi pada hari ke-18 yaitu sebesar 5004 KL. Selain itu, terdapat kondisi dimana persediaan pelabuhan berada dibawah alarm level inventory sehingga dikenai biaya penalti.

b. Pelabuhan TWI

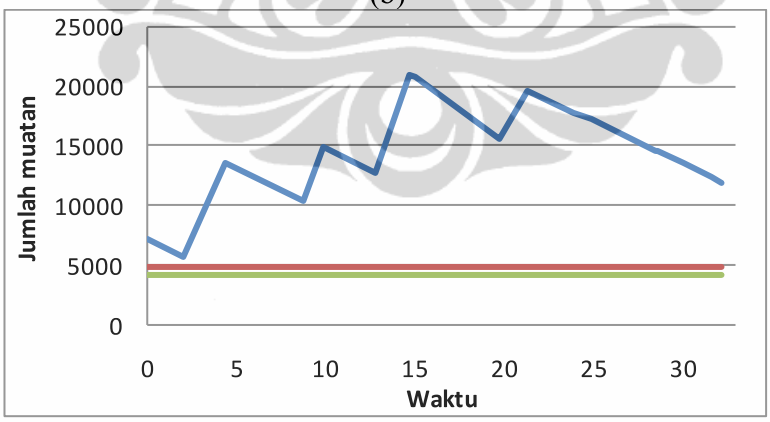
Berikut adalah grafik pergeseran stok Premium, Kerosin dan Solar di pelabuhan TWI.



(a)



(b)



(c)

Gambar 4.3 Persediaan Pelabuhan TWI
(a) Premium, (b) Kerosin, dan (c) Solar

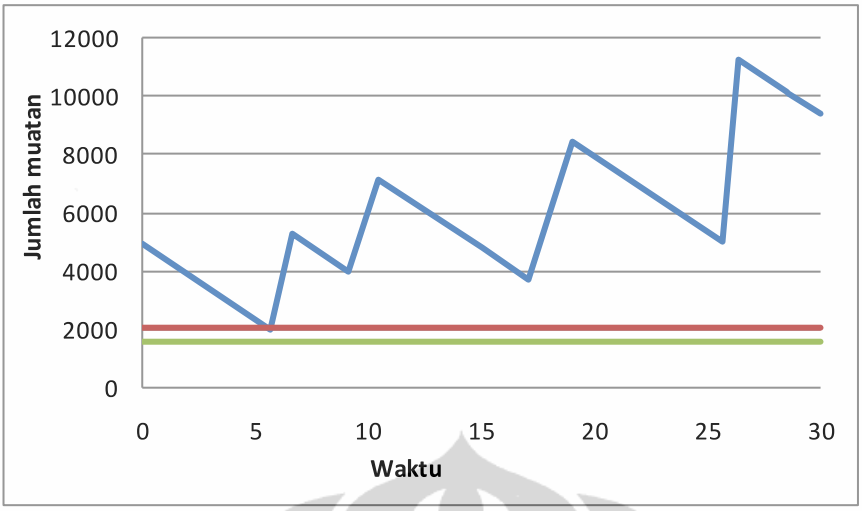
Besar muatan yang dibongkar untuk produk Premium, kurang lebih hampir sama di setiap kedatangan kapal. Intensitas kedatangan kapal pada pelabuhan ini juga termasuk banyak, terlihat dari fluktuasi grafik stok

premium dankerosene yang lebih intens diisi oleh kapal. Lain hanya dengan Kerosin yang pengisiannya hanya dilakukan sebanyak 1 kali dengan bongkar muatan sebesar 5004 KL. Meskipun demikian, persediaan premium, Kerosin dan solar pada pelabuhan TWI dapat terjaga selama 30 hari dan tetap berada diatas alarm level pada saat kapal datang. Namun, ada beberapa kondisi dimana kapal datang pada saat persediaannya mendekati *alarm level*.

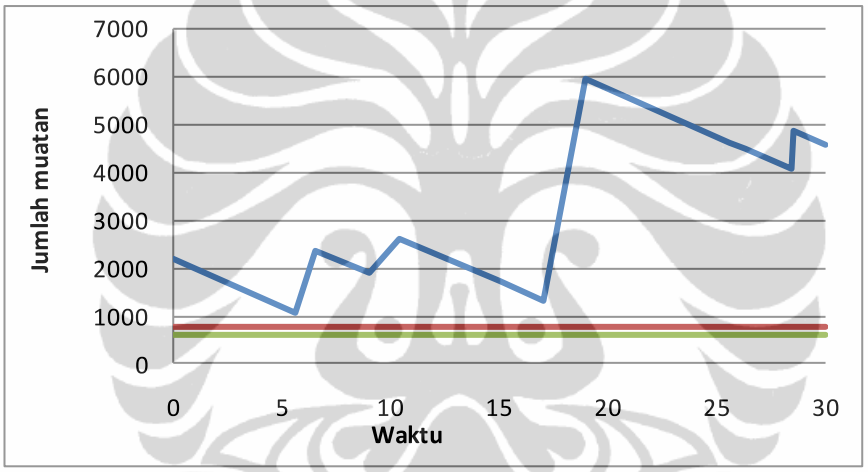
c. Pelabuhan KUP

Stok premium (Gb. 4.4a) pada pelabuhan ini pernah berada dibawah alarm level pada hari ke – 5.62 hari. Tetapi pada hari selanjutnya, stok berada diatas alarm level. Berbeda halnya dengan stok Kerosin (Gb. 4.4b) dan solar (Gb.4.4c) yang tidak pernah berada pada atau di bawah alarm level. Intensitas kapal melakukan pengisian pada setiap tangki premium, Kerosin dan solar hampir sama banyaknya.

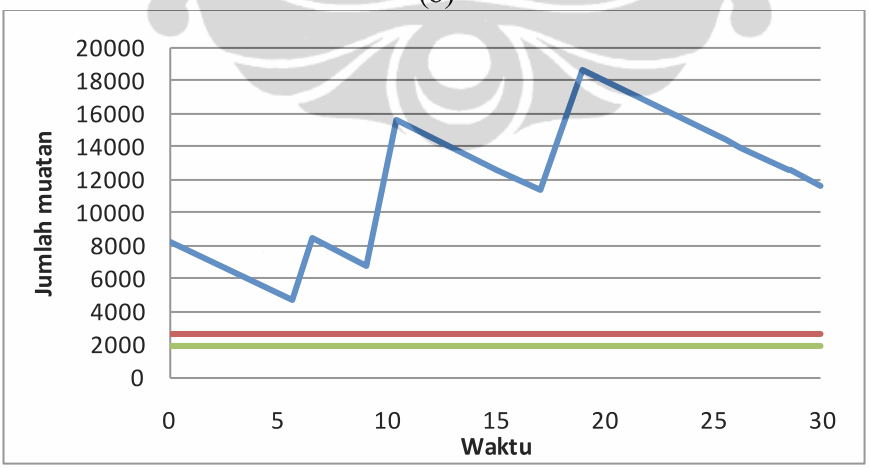
Berikut adalah grafik pergeseran stok premium, Kerosin dan solar pada pelabuhan KUP selama 30 hari.



(a)



(b)

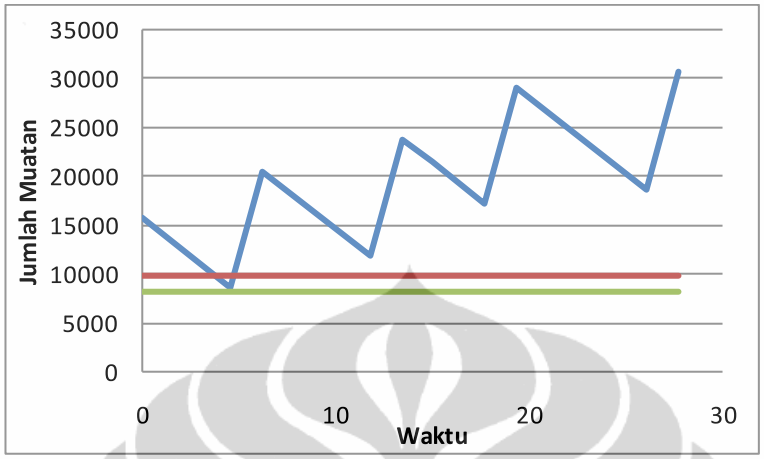


(c)

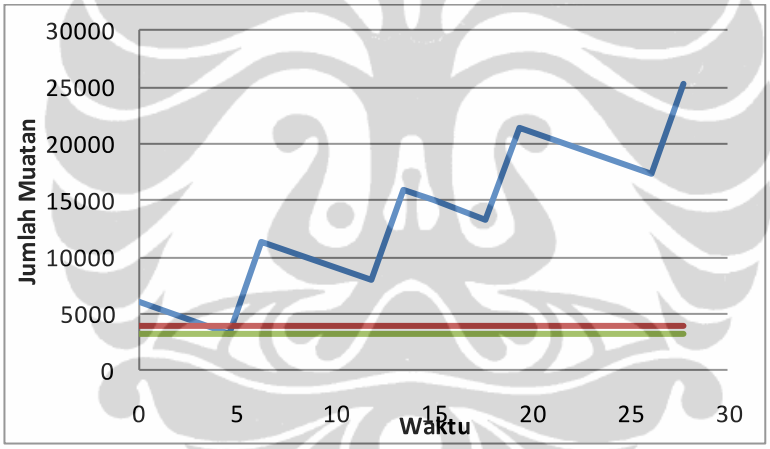
Gambar 4.4 Persediaan Pelabuhan KUP
(a) Premium, (b) Kerosin, dan (c) Solar

d. Pelabuhan TTM

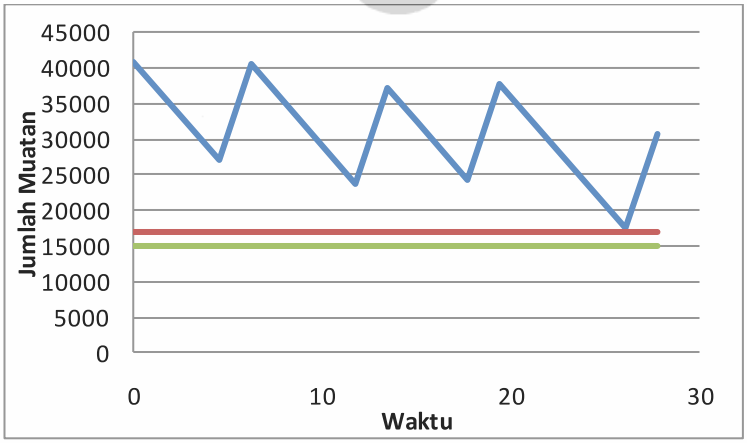
Berikut adalah grafik pergeseran stok Premium, Kerosin dan Solar di pelabuhan TTM.



(a)



(b)



(c)

Gambar 4.5 Persediaan Pelabuhan TTM
(a)Premium, (b) Kerosin, dan (c) Solar

Fluktuasi stok Premium, Kerosin dan Solar di pelabuhan ini adalah merata karena setiap kali kapal datang, kapal melakukan bongkar muatan untuk ke- 3 produk. Hal ini terjadi karena ketika kapal MR *dedicated* untuk pelabuhan ini, maka pergerakan kapal sehabis melakukan muat di pelabuhan *loading* kapal akan menuju pelabuhan ini lalu kembali ke pelabuhan *loading*, dan demikian seterusnya. Persediaan Premium dan Kerosin pernah berada dibawah alarm level pada kedatangan kapal dihari ke-4.54. Untuk hari selanjutnya, stok tidak pernah berada di bawah alarm level dan tentunya persediaan pengaman tetap terjaga.

Seluruh grafik persediaan di atas menunjukkan bahwa rute kapal yang dihasilkan adalah baik karena kapal dapat menjaga keberadaan persediaan pengaman di setiap pelabuhan *unloading*. Kapal selalu berupaya melakukan pendistribusian produk dengan baik termasuk melakukan perjalanan ke pelabuhan yang memiliki persediaan produk yang kritis untuk dipenuhi.

4.4.4.3 Analisis Biaya

Total biaya yang dihasilkan menggunakan Algoritma Tabu Search dengan perencanaan = 15 hari memiliki perbedaan dengan hasil yang diperoleh dengan menggunakan *Branch and Bound* dengan Lingo (Lampiran 4). Biaya yang dihasilkan oleh Lingo adalah sebesar Rp 3,274,090,581.6 (Lampiran 3) sedangkan dari penggunaan *Tabu Search* adalah Rp 2,887,816,160.62. Biaya yang dihasilkan dari penyelesaian menggunakan Algoritma *Tabu Search* menghemat sebesar 11,8 %. Untuk penjadwalan = 30 hari, biaya yang dihasilkan adalah sebesar Rp 6,265,337,216.33. Untuk perhitungannya dapat dilihat di Lampiran 3.

4.4.4.4 Analisis Sensitivitas

Analisis sensitivitas pada permasalahan ini dilakukan dengan mengubah jumlah kapal dan jumlah kapasitas kapal. Dengan menambah kapal yang kapasitasnya sama, rute yang ditempuh kapal lebih singkat karena masing-masing kapal dapat melayani 1 pelabuhan *unloading*. Tetapi, biaya yang dihasilkan akan lebih mahal. Sedangkan apabila dilakukan pengurangan kapal 1, kapal tidak dapat memenuhi kebutuhan 3 pelabuhan *unloading* dengan menjaga keberadaan persediaan pengaman. Selain karena dipengaruhi jarak ke-tiga pelabuhan *loading*

dengan *unloading* yang berbeda-beda besarnya, hal ini di pengaruhi juga kapasitas kapal yang hanya cukup memenuhi kebutuhan dalam waktu yang singkat dalam sekali perjalanan dari pelabuhan *loading*. Pelabuhan *unloading* akan sering mengalami krisis persediaan.

Analisis sensitivitas berikutnya adalah dengan menggunakan kapal dengan ukuran kapasitas kapal yang lebih kecil. Pengurangan kapasitas kapal tidak dapat memenuhi stok secara merata untuk semua pelabuhan *unloading*. Akan ada kondisi dimana pelabuhan menggunakan persediaan pengamannya untuk dikonsumsi karena kedatangan kapal yang lama di pelabuhan itu. Dengan kata lain, mengunakan kapal dengan kapasitas lebih kecil bukanlah upaya yang terlalu baik untuk mendistribusikan bahan bakar.

BAB 5
PENUTUP

5.1 Kesimpulan

Kesimpulan dari penelitian ini adalah diperolehnya sebuah usulan mengenai rute dan jadwal kapal dalam mendistribusikan premium, Kerosin dan solar dengan menjaga persediaan pengaman yang optimal pada setiap pelabuhan bongkar menggunakan Algoritma *Tabu Search*. Pada penyelesaian permasalahan ini, kapal MR dikeluarkan dari model dan dibuat *dedicated* memenuhi kebutuhan di pelabuhan TTM. Penyelesaian permasalahan menggunakan Algoritma *Tabu Search* ini memperoleh biaya yang lebih kecil dibandingkan dengan penelitian sebelumnya yang menggunakan metode *branch and bound* dengan Lingo. Hasil diperoleh dalam waktu yang singkat. Penggunaan Algoritma *Tabu Search* menghasilkan performansi yang lebih baik dibandingkan Lingo karena dapat melakukan perencanaan dalam waktu yang lebih panjang tanpa melakukannya secara terpisah.

5.2 Saran

Saran dari penelitian ini adalah dalam kondisi aktual, pelabuhan *loading* memiliki batas kapasitas dalam menghasilkan produk setiap harinya. *Production Rate* perlu dilibatkan dalam permasalahan ini untuk lebih menggambarkan kondisi yang lebih nyata di dalam sistem pendistribusian Premium, Kerosin dan Solar.

DAFTAR REFERENSI

Al-Khayyal, F & Hwang, S.(2007). *Inventory Constrained Maritime Routing and Schedulling for Multi-Commodity Liquid Bulk.Part:1 Applications and Model,* European Journal of Operational Research 176, 2007

Ballou, Ronald H. (1992). *Business Logistic Management*, 4 th ed. New Jersey: Prentice-Hall Inc.

Cesar Rego, Bahran Alidance. (2005). *Methaheuristic Optimization Via Memory and Evolution Tabu Search and Scatter Search*. United State of America: Academic Publishers.

Talbi, El-Ghazali. (2009). *Methaheuristic From Design To Implementation*. New Jersey: A John Willey & Sons.

Glover, F., and Laguna, M. (1997). *Tabu Search*. United State of America: Kluwer Academic Publisher.

Gunther, Zapfel.(2010). *Metaheuristic Search Concepts: A Tutorial with Application to Production and Logistic*. New York : Springer.

Harianto, Slamet. (2011). Penentuan Rute dan Jadwal Pengiriman Multi Produk Pada Distribusi Bahan Bakar Untuk Menjaga Persediaan pengaman Dengan Pendekatan Branch and Bound

Hillier, F.S. and Lieberman, G.J. (2005). *Introduction to Operations Research*. New York: McGraw-Hill. 8th Ed.

Korsvik, JE., Fagerholt, K., & Laporte, G.(2010). A Tabu Search Heuristic for Ship Routing and Scheduling. *Journal of Operational Research Society* Vol. 61, 594-603.

Siswanto, N., Essam,D. & Sarker,R.(2009). *Solving The Ship Inventory Routing and Schedulling Problem with Undedicated Compartment*.Computer and Industrial Engineering Elsevier.

Taha, Hamdy A. (2007). *Operation Research,an Introduction Eight Edition*. United State of America: Pearson- Prentice Hall

www.pertamina.com, (diakses pada February 28,2012)

www.pertaminashipping.com (diakses pada February 28,2012)

LAMPIRAN

Lampiran 1 : *Script M-File Program*

```
==============Program Utama Solusi Awal==================
Pasokan_Maksimal          = 15;
product                   = 3;
kapal                     = 2;

% Import Data Excel
Data_Sea_Time_GP          = xlsread('data ana.xlsx','Sea_time','H3:J46');
Data_Sea_Time_MR          = xlsread('data ana.xlsx','Sea_time','C4:E21');
Consumption_Rate          = xlsread('data ana.xlsx','Production dan Consumption','F10:H13');
Waktu_Load_Unload         = xlsread('data ana.xlsx','Waktu un(loading) muatan (TQi)','D4:D9');
Kapasitas_Kapal           = xlsread('data ana.xlsx','Kapasitas_kapal','D6:F7');
Inventory_Awal_Pelabuhan  = xlsread('data ana.xlsx','Inventory awal pelabuhan ','D6:F13');
Safety_Stock              = xlsread('data ana.xlsx','SMXi, SMNi','C5:E12');
Data_Port_Cost            = xlsread('data ana.xlsx','Biaya Pelabuhan','D4');
Biaya_Sea_Time            = xlsread('data ana.xlsx','Biaya_Berlayar','C4');

% 1) Create Solusi Awal
[Solusi_Awal,Data_Muatan,Data_Waktu,Sea_Time,Time_Elapsed,Lama_Pasokan,Biaya_Pinalti,
Datang_Pergi] =
Solusi_Awal_Terbaik(Inventory_Awal_Pelabuhan,Safety_Stock,Consumption_Rate,Pasokan_Ma
ksimal,Data_Sea_Time_GP,Data_Sea_Time_MR,product,Kapasitas_Kapal,Waktu_Load_Unload,
kapal);
rute = numel(Solusi_Awal(1,:));

%Biaya Kunjungan Pelabuhan
[Port_Cost]               = Get_Port_Cost(Data_Port_Cost,Solusi_Awal,rute,kapal);
[Sea_Time_Cost]           = Get_Transportation_Cost(Biaya_Sea_Time,Sea_Time,rute,kapal);
Total_Sea_Time_Cost       = sum(Sea_Time_Cost,2);

% Biaya Total
Biaya_Total_Per_Rute      = Total_Sea_Time_Cost + sum(sum(Biaya_Pinalti,2),3) +
sum(Port_Cost,2);
Biaya_Total_Semua_Kapal   = sum(Biaya_Total_Per_Rute,1)
Solusi_Awal




% ============== PART 2: TABU SEARCH
============================
%
=========================================================
===

Solusi_Awal

Max_Iterasi               = ;
Pasokan_Maksimal          = ;
Panjang_Tabu_List         = ;
```

```
Banyaknya_Solusi_Tetangga  = ;

Tabel_Tabu_List    = zeros(kapal,numel(Solusi_Awal(1,:)),1);
Index_Tabu_List    = 1;

%Inisialisasi Tujuan Tabu Search
Solusi_Agak_Baru   = Solusi_Awal;
Solusi_Terbaik     = Solusi_Awal;

Z_Awal         = Biaya_Total_Semua_Kapal;
Z_Lama         = Z_Awal;
Z_Terbaik      = Z_Awal;
Z_Lokal_Terbaik = Z_Awal;
toc;

tic;
Index_Solusi_Global = 1;
for iterasi = 1:Max_Iterasi
   Index_Solusi_Tetangga  = 1;
   for Iterasi_Tetangga = 1:Banyaknya_Solusi_Tetangga
      [Solusi_Tetangga]  = Perform_Swap(Solusi_Agak_Baru,kapal)
      [Result] = First_Inspector(Solusi_Tetangga,rute,kapal);
      if  Result == 'OK'

[Result_Baru,Sea_Time_Solusi_Tetangga,Data_Muatan_Solusi_Tetangga,Data_Waktu_Solusi_Te
tangga,Time_Elapsed_Solusi_Tetangga,Isi_Tangki_Kapal_Solusi_Tetangga,Isi_Pelabuhan_Solusi
_Tetangga,Ketersediaan_Solusi_Tetangga,Lama_Pasokan_Solusi_Tetangga,Biaya_Pinalti_Solusi
_Tetangga,Jadwal_Kapal_Solusi_Tetangga,Stok_Saat_Kapal_Datang_Baru] =
Inverse_Solusi(Solusi_Tetangga,Inventory_Awal_Pelabuhan,Safety_Stock,Consumption_Rate,Wa
ktu_Load_Unload,Kapasitas_Kapal,Data_Sea_Time_GP,Data_Sea_Time_MR,Pasokan_Maksimal
,kapal,rute);
         if Result_Baru == 'OK'
            [Sea_Time_Cost_Solusi_Tetangga]    =
Get_Transportation_Cost(Biaya_Sea_Time,Sea_Time_Solusi_Tetangga,rute,kapal);
            [Result] =
Second_Inspector(Lama_Pasokan_Solusi_Tetangga,Pasokan_Maksimal,Jadwal_Kapal_Solusi_Tet
angga);
            if Result == 'OK'
               Port_Cost_Tetangga              =
Get_Port_Cost(Data_Port_Cost,Solusi_Tetangga,rute,kapal);
               Z_Lokal_Baru                    =
sum(Sea_Time_Cost_Solusi_Tetangga,2)+sum(Port_Cost_Tetangga,2)+sum(sum(sum(Biaya_Pin
alti_Solusi_Tetangga,4),3),2);
               Z_Lokal_Baru                    = sum(Z_Lokal_Baru,1);
               Lokal_Z(Index_Solusi_Tetangga,1)   = Z_Lokal_Baru;
               Solusi_Lokal_Terbaik(:,:,Index_Solusi_Tetangga) = Solusi_Tetangga;

% Geser posisi solusi terbaik dari iterasi
%
=============================================================
===
% ====================== PART 0: INISIALISASI
====================
%
% ============== PART 1: GENERATE SOLUSI BERIKUTNYA
==============
```

```
%
==========================================================
===

          TAG_PELABUHAN_UNLOADING(PELABUHAN,1) = 1;

          if  PELABUHAN == 1
             Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 1;
          elseif PELABUHAN == 2
             Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 3;
          elseif PELABUHAN == 3
             Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 5;
          else
             Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 7;
          end
        end
      else
        if numel(find(Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,:) == 0)) >= 2
          Loading_Tersedia    = find(TAG_PELABUHAN_LOADING == 0);
          Pilih_Loading       = randsample([1 3],1);
          TAG_PELABUHAN_LOADING(Pilih_Loading,1)  = 1;

          if Pilih_Loading == 1
             Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 2;
          elseif Pilih_Loading == 3
             Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 6;
          else
             Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 8;
          end
        else
          if numel(find(Lama_Pasokan([1;3],:) >= Pasokan_Maksimal)) ~= 6
             if  Solusi_Awal(Kapal_Sekarang,Rute_Sekarang-1) == 5
                Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 1;
                TAG_PELABUHAN_UNLOADING(1,1) = 1;
             else
                Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 5;
                TAG_PELABUHAN_UNLOADING(3,1) = 1;
             end
          else
             Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 0;
          end

        end
      end
   else
      if Kapal_Sekarang == 1 || Kapal_Sekarang == 2 || Kapal_Sekarang == 3
        Prediksi_Sekarang              = Isi_Pelabuhan_Sekarang([1;3;5;7],:)-
Consumption_Rate*Time_Elapsed(Kapal_Sekarang,1);
        Pelabuhan_Unloading_Tersedia   = find(TAG_PELABUHAN_UNLOADING == 0);
        Muatan_Kapal_Tersedia          = find(Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,:) ~=
0);
        Muatan_Paling_Mau_Habis        =
min(min(Prediksi_Sekarang(Pelabuhan_Unloading_Tersedia,Muatan_Kapal_Tersedia)));

        [PELABUHAN,PRODUK] = find(Prediksi_Sekarang == Muatan_Paling_Mau_Habis);
        TAG_PELABUHAN_UNLOADING(PELABUHAN,1) = 1;
```

```
        if  PELABUHAN == 1
            Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 1;
        elseif PELABUHAN == 2
            Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 3;
        elseif PELABUHAN == 3
            Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 5;
        else
            Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 7;
        end
      else
        if numel(find(Lama_Pasokan([1;3],:) >= Pasokan_Maksimal)) ~= 6
          if  Solusi_Awal(Kapal_Sekarang,Rute_Sekarang-1) == 5
            Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 1;
            TAG_PELABUHAN_UNLOADING(1,1) = 1;
          else
            Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 5;
            TAG_PELABUHAN_UNLOADING(3,1) = 1;
          end
        else
          Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) = 0;
        end

      end
    end

    %
==========================================================
==
    % ============== PART 2: GENERATE SEA TIME
=======================
    %
==========================================================
==


    for b=1:44
      if Solusi_Awal(Kapal_Sekarang,Rute_Sekarang-1:Rute_Sekarang) ==
Data_Sea_Time_GP(b,2:3);
        Sea_Time(Kapal_Sekarang,Rute_Sekarang-1) = Data_Sea_Time_GP(b,1);
      end
      if Solusi_Awal(Kapal_Sekarang,Rute_Sekarang-1) ==
Solusi_Awal(Kapal_Sekarang,Rute_Sekarang)
        Sea_Time(Kapal_Sekarang,Rute_Sekarang-1) = 0;
      end
    end


    Time_Elapsed(Kapal_Sekarang,1) = Time_Elapsed(Kapal_Sekarang,1) +
Sea_Time(Kapal_Sekarang,Rute_Sekarang-1);

    %
==========================================================
==
    % ============== PART 3: GENERATE DATA MUATAN
====================
```

```
  %
==========================================================
===

  if Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) ~= 0
  Kunjungan(Solusi_Awal(Kapal_Sekarang,Rute_Sekarang),1) =
Kunjungan(Solusi_Awal(Kapal_Sekarang,Rute_Sekarang),1) + 1;
  for k = 1:product
    if Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) == 1      % ---------- Unloading
      Datang_Pergi(1,2*Kunjungan(1,1)-1) = Time_Elapsed(Kapal_Sekarang,1);
      % Cuma terima Kerosene
      if k == 1 || k ==3
        Unloading = 0;
        Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) = Unloading;
      else
        if Isi_Pelabuhan_Sekarang(1,2)./Consumption_Rate(1,2) < Pasokan_Maksimal
          Stok_Saat_Kapal_Datang(Kapal_Sekarang,Rute_Sekarang) =
Isi_Pelabuhan_Sekarang(1,2)-Consumption_Rate(1,2)*Datang_Pergi(1,2*Kunjungan(1,1)-1);

          if Stok_Saat_Kapal_Datang < Consumption_Rate(1,2)
            Penurunan_Muatan = Consumption_Rate(1,2)-Stok_Saat_Kapal_Datang;
            Biaya_Pinalti(Kapal_Sekarang,2,Rute_Sekarang) = Penurunan_Muatan*1e+5;
          else
            Biaya_Pinalti(Kapal_Sekarang,2,Rute_Sekarang) = 0;
          end

          Jangka_Waktu(Kapal_Sekarang,k) =
randsample([Hari_Minimal:0.1:Hari_Maksimal],1);
          SWITCH = round(rand) == 0;
          if SWITCH == 0
            Tuang_Sebagian = Jangka_Waktu(Kapal_Sekarang,k)*Consumption_Rate(1,k);
            if Tuang_Sebagian > Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k)
              Unloading = Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k);
              Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k) = 0;
              Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) = -Unloading;
              Isi_Pelabuhan_Sekarang(1,2) = Isi_Pelabuhan_Sekarang(1,2) + Unloading;
            else
              Unloading = Tuang_Sebagian;
              Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k) =
Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k) - Tuang_Sebagian;
              Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) = -Unloading;
              Isi_Pelabuhan_Sekarang(1,2) = Isi_Pelabuhan_Sekarang(1,2) + Unloading;
            end
          else
            Tuang_Semua = Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k);
            Unloading = Tuang_Semua;
            Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k) = 0;
            Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) = -Unloading;
            Isi_Pelabuhan_Sekarang(1,2) = Isi_Pelabuhan_Sekarang(1,2) + Unloading;

            %Adjust Maximum Stock Pelabuhan
            if Isi_Pelabuhan_Sekarang(1,2) > Consumption_Rate(1,2)*Pasokan_Maksimal
              Max_Need                = Consumption_Rate(1,2)*Pasokan_Maksimal;
              Balikin_Ke_Kapal        = Isi_Pelabuhan_Sekarang(1,2) - Max_Need;
              Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k)  = Balikin_Ke_Kapal;
              Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang)          =
Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) + Balikin_Ke_Kapal;
```

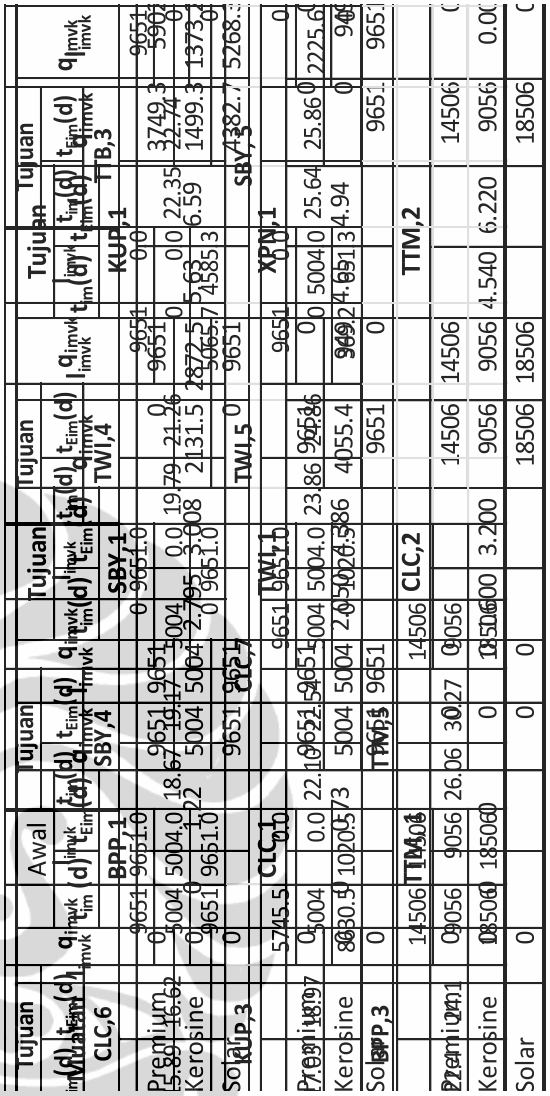
```
                Isi_Pelabuhan_Sekarang(1,2)     = Max_Need;
              end
            end
          else % Jika sudah mencapai pasokan maksimal
            Unloading = 0;
            Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) = -Unloading;
            Isi_Pelabuhan_Sekarang(1,2) = Isi_Pelabuhan_Sekarang(1,2) + Unloading;
          end
        end

        if Kapal_Sekarang == 1 || Kapal_Sekarang == 2 || Kapal_Sekarang == 3
          Waktu_Unload_Kapal = Waktu_Load_Unload(5,1);
        else
          Waktu_Unload_Kapal = Waktu_Load_Unload(6,1);
        end

        %Adjust Maximum Stock Pelabuhan
        if Isi_Pelabuhan_Sekarang(1,2) > Consumption_Rate(1,2)*Pasokan_Maksimal
          Max_Need                  = Consumption_Rate(1,2)*Pasokan_Maksimal;
          Balikin_Ke_Kapal          = Isi_Pelabuhan_Sekarang(1,2) - Max_Need;
          Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k)  = Balikin_Ke_Kapal;
          Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang)          =
Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) + Balikin_Ke_Kapal;
          Isi_Pelabuhan_Sekarang(1,2)     = Max_Need;
        end

        Data_Waktu(Kapal_Sekarang,Rute_Sekarang) = -
sum(Data_Muatan(Kapal_Sekarang,:,Rute_Sekarang),2)*Waktu_Unload_Kapal;
        if k == 3
          Datang_Pergi(1,2*Kunjungan(1,1)) = Datang_Pergi(1,2*Kunjungan(1,1)-1) +
Data_Waktu(Kapal_Sekarang,Rute_Sekarang);
          Time_Elapsed(Kapal_Sekarang,1) = Datang_Pergi(1,2*Kunjungan(1,1));
        end
      end

      if Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) == 2    % ------------------- Loading
        Datang_Pergi(2,2*Kunjungan(2,1)-1) = Time_Elapsed(Kapal_Sekarang,1);
        if Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k) == 0
          Loading = Kapasitas_Kapal(Kapal_Sekarang,k) -
Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k);
          Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k) =
Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k) + Loading;
          Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) = Loading;
          Data_Waktu(Kapal_Sekarang,Rute_Sekarang) =
sum(Data_Muatan(Kapal_Sekarang,:,Rute_Sekarang),2)*Waktu_Load_Unload(1,1);
          if k == 3
            Datang_Pergi(2,2*Kunjungan(2,1)) = Datang_Pergi(2,2*Kunjungan(2,1)-1) +
Data_Waktu(Kapal_Sekarang,Rute_Sekarang);
            Time_Elapsed(Kapal_Sekarang,1) = Datang_Pergi(2,2*Kunjungan(2,1));
          end
        else
          Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) = 0;
          Data_Waktu(Kapal_Sekarang,Rute_Sekarang) =
sum(Data_Muatan(Kapal_Sekarang,:,Rute_Sekarang),2)*Waktu_Load_Unload(1,1);
          if k == 3
            Datang_Pergi(2,2*Kunjungan(2,1)) = Datang_Pergi(2,2*Kunjungan(2,1)-1) +
Data_Waktu(Kapal_Sekarang,Rute_Sekarang);
```

```
            Time_Elapsed(Kapal_Sekarang,1) = Datang_Pergi(2,2*Kunjungan(2,1));
          end
        end
      end

      if Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) == 3  % ---------- Unloading
        Datang_Pergi(3,2*Kunjungan(3,1)-1) = Time_Elapsed(Kapal_Sekarang,1);
        if Isi_Pelabuhan_Sekarang(3,k)./Consumption_Rate(2,k) < Pasokan_Maksimal
          Jangka_Waktu(Kapal_Sekarang,k) =
randsample([Hari_Minimal:0.1:Hari_Maksimal],1);
          Stok_Saat_Kapal_Datang(Kapal_Sekarang,k) = Isi_Pelabuhan_Sekarang(3,k)-
Consumption_Rate(2,k)*Datang_Pergi(3,2*Kunjungan(3,1)-1);
          if Stok_Saat_Kapal_Datang < Consumption_Rate(2,k)
            Penurunan_Muatan = Consumption_Rate(2,k)-Stok_Saat_Kapal_Datang;
            Biaya_Pinalti(Kapal_Sekarang,k,Rute_Sekarang) = Penurunan_Muatan*1e+5;
          else
            Biaya_Pinalti(Kapal_Sekarang,k,Rute_Sekarang) = 0;
          end

          SWITCH = round(rand) == 0;
          if SWITCH == 0
            Tuang_Sebagian = Jangka_Waktu(Kapal_Sekarang,k)*Consumption_Rate(2,k);
            if Tuang_Sebagian > Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k)
              Unloading = Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k);
              Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k) = 0;
            else
              Unloading = Tuang_Sebagian;
              Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k) =
Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k) - Tuang_Sebagian;
            end
          else
            Tuang_Semua = Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k);
            Unloading = Tuang_Semua;
            Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k) = 0;
          end
        else
          Unloading = 0;
        end

        Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) = -Unloading;
        Isi_Pelabuhan_Sekarang(3,k) = Isi_Pelabuhan_Sekarang(3,k) + Unloading;
        %             Prediksi_Habis(2,k) = Isi_Pelabuhan_Sekarang(3,k)/Consumption_Rate(2,k);

        if Kapal_Sekarang == 1 || Kapal_Sekarang == 2 || Kapal_Sekarang == 3
          Waktu_Unload_Kapal = Waktu_Load_Unload(5,1);
        else
          Waktu_Unload_Kapal = Waktu_Load_Unload(6,1);
        end

        %Adjust Maximum Stock Pelabuhan
        if Isi_Pelabuhan_Sekarang(3,k) > Consumption_Rate(2,k)*Pasokan_Maksimal
          Max_Need                = Consumption_Rate(2,k)*Pasokan_Maksimal;
          Balikin_Ke_Kapal        = Isi_Pelabuhan_Sekarang(3,k) - Max_Need;
          Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k)  = Balikin_Ke_Kapal;
          Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang)          =
Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) + Balikin_Ke_Kapal;
          Isi_Pelabuhan_Sekarang(3,k)     = Max_Need;
```

```
        end
        Data_Waktu(Kapal_Sekarang,Rute_Sekarang) = -
sum(Data_Muatan(Kapal_Sekarang,:,Rute_Sekarang),2)*Waktu_Unload_Kapal;

        if k == 3
           Datang_Pergi(3,2*Kunjungan(3,1))    = Datang_Pergi(3,2*Kunjungan(3,1)-
1)+Data_Waktu(Kapal_Sekarang,Rute_Sekarang);
           Time_Elapsed(Kapal_Sekarang,1)      = Datang_Pergi(3,2*Kunjungan(3,1));
        end
     end

     if Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) == 4    % -------------------- Loading
        Datang_Pergi(4,2*Kunjungan(4,1)-1) = Time_Elapsed(Kapal_Sekarang,1);

        % Cuma produksi Premium dan Solar
        if k == 2
           Loading = 0;
           Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) = Loading;
        else
           if Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k) == 0
              Loading = Kapasitas_Kapal(Kapal_Sekarang,k) -
Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k);
              Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k) =
Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,k) + Loading;
              Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) = Loading;
           else
              Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) = 0;
           end
        end
        Data_Waktu(Kapal_Sekarang,Rute_Sekarang) =
sum(Data_Muatan(Kapal_Sekarang,:,Rute_Sekarang),2)*Waktu_Load_Unload(2,1);
        if k == 3
           Datang_Pergi(4,2*Kunjungan(4,1)) = Datang_Pergi(4,2*Kunjungan(4,1)-1) +
Data_Waktu(Kapal_Sekarang,Rute_Sekarang);
           Time_Elapsed(Kapal_Sekarang,1)   = Datang_Pergi(4,2*Kunjungan(4,1));
        end
     end

     if  Solusi_Awal(Kapal_Sekarang,Rute_Sekarang) == 5   % -------------------- Unloading
        Datang_Pergi(5,2*Kunjungan(5,1)-1) = Time_Elapsed(Kapal_Sekarang,1);
        % Cuma terima Kerosene
        if k == 1 || k ==3
           Unloading = 0;
           Data_Muatan(Kapal_Sekarang,k,Rute_Sekarang) = Unloading;
        else

        if Isi_Pelabuhan_Sekarang(5,2)./Consumption_Rate(3,2) < Pasokan_Maksimal
           Jangka_Waktu(Kapal_Sekarang,2) =
randsample([Hari_Minimal:0.1:Hari_Maksimal],1);
           Kedatangan_Kapal = Time_Elapsed(Kapal_Sekarang,1);
           Stok_Saat_Kapal_Datang(Kapal_Sekarang,2) = Isi_Pelabuhan_Sekarang(5,2)-
Consumption_Rate(3,2)*Datang_Pergi(5,2*Kunjungan(5,1)-1);

           if Stok_Saat_Kapal_Datang < Consumption_Rate(3,2)
              Penurunan_Muatan = Consumption_Rate(3,2)-Stok_Saat_Kapal_Datang;
              Biaya_Pinalti(Kapal_Sekarang,k,Rute_Sekarang) = Penurunan_Muatan*1e+5;
           else
```

```
            Biaya_Pinalti(Kapal_Sekarang,k,Rute_Sekarang) = 0;
        end

        SWITCH = round(rand) == 0;
        if SWITCH == 0
            Tuang_Sebagian = Jangka_Waktu(Kapal_Sekarang,2)*Consumption_Rate(3,2);
            if Tuang_Sebagian > Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,2)
                Unloading = Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,2);
                Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,2) = 0;
            else
                Unloading = Tuang_Sebagian;
                Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,2) =
Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,2) - Tuang_Sebagian;
            end
        else
            Tuang_Semua = Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,2);
            Unloading = Tuang_Semua;
            Isi_Tangki_Kapal_Sekarang(Kapal_Sekarang,2) = 0;
        end
    else
        Unloading = 0;
    end

    Data_Muatan(Kapal_Sekarang,2,Rute_Sekarang) = -Unloading;
    Isi_Pelabuhan_Sekarang(5,2) = Isi_Pelabuhan_Sekarang(5,2) + Unloading;
    %               Prediksi_Habis(3,k) =
Isi_Pelabuhan_Sekarang(5,k)/Consumption_Rate(3,k);
    end
    if Kapal_Sekarang == 1 || Kapal_Sekarang == 2 || Kapal_Sekarang == 2
        Waktu_Unload_Kapal = Waktu_Load_Unload(5,1);
    else
        Waktu_Unload_Kapal = Waktu_Load_Unload(6,1);
    end
```

Lampiran 2 Data Perhitungan Muatan (T= 30 hari)



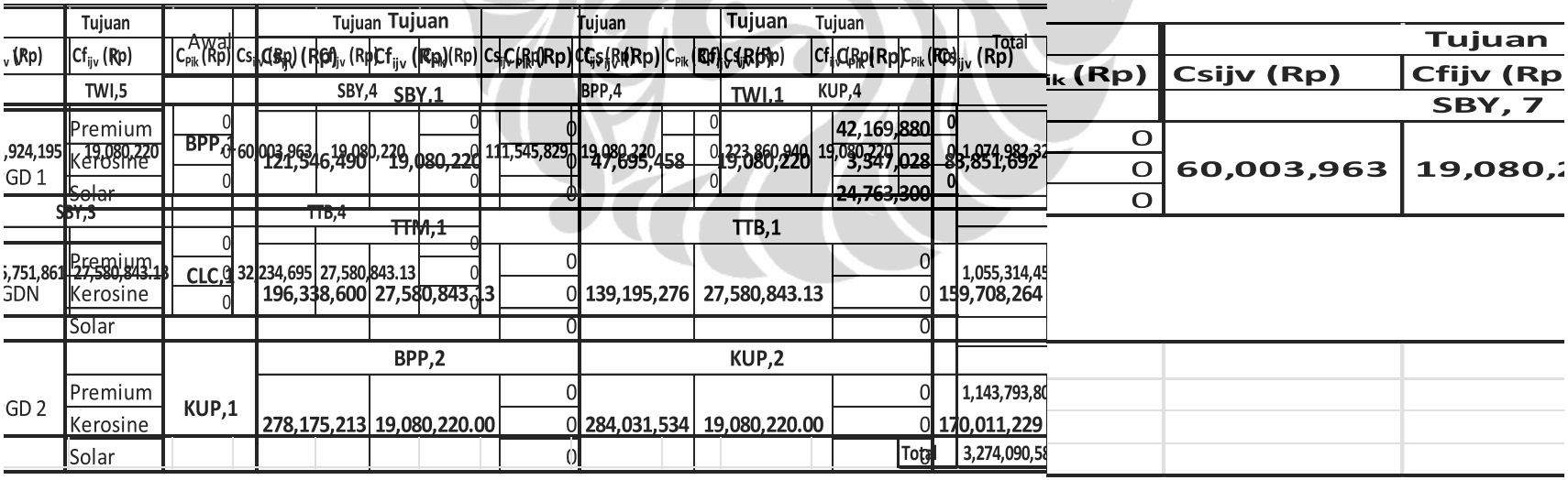
Lampiran 2 Data Perhitungan Muatan (T=30 hari) (lanjutan)

Lampiran 3 Perhitungan Biay

Awal (Rp)		Tujuan SBY, 1 (TWI, 4)				Tujuan TTB, 3				Tujuan KUP, 1 (KUP, 4)				Tujuan TWI, 2 (TWI, 5)			
		CS_{ijv} (Rp)	CF_{ijv} (Rp)	C_{Pik} (Rp)	Csijv (Rp)	Cfijv (Rp)	Csijv (Rp)	C_{Pik} (Rp)	Csijv (Rp)	Cf_{ijv} (Rp)	Cfijv (Rp)	BG_{ik} (Rp)	Csijv (Rp)	Csijv (Rp)	Cfijv (Rp)	Cf_{ijv} (Rp)	C_{Pik} (Rp)
0	BPP, 1 56,967	47,695,458 / 121,546,490	19,080,220 / 19,080,220	0 / 0 / 0	83,851,692	19,080,220	0 / 216,937,406 / 0	0 / 0	224,630,321	19,080,220 / 19,080,220	19,080,220	7,588,492 / 0 / 0	0	170,011,229 / 170,011,229	19,080,220 / 19,080,220	0 / 0 / 0	92 / 313,790
		Tujuan TWI,2 (TWI,5)				Tujuan SBY, 5				Tujuan XPN, 1 (KUP, 5)				Tujuan SBY, 2 (SBY, 6)			
0	CLC, 0	101,545,169 / 157,702,724	19,080,220 / 19,080,220	0 / 0 / 0	60,003,963	19,080,220	0 / 101,545,169 / 0	0 / 0	201,551,774	19,080,220 / 19,080,220	19,080,220	0 / 0 / 0	0 / 0	214,629,561 / 41,541,205	19,080,220 / 19,080,220	0 / 0 / 0	111,545,829
		CLC,2				TTM,2								BPP,2			
0	TTM, 1	234,434,149	27,580,843		0 / 0 / 0		196,338,600			27,580,843		111,868,880 / 54,171,560	0	218,316,801	27,580,843		

imvk	l_{imvk}	Tujuan t_{im}(d)	Tujuan t_{Eim}(d)	q_{imvk}	l_{imvk}	Tujuan t_{im}(d)	Tujuan t_{Eim}(d)
		CLC,8				TWI,6	
94.1	0.0	29.82	30.26	9651	9651.0	31.58	32.15
0	0.0			5004	5004.0		
0	4585.3			0	4585.3		
		BPP,2					
0	0.0	32.85	32.85	9651			
.233	1496.1			0			
0	651.3			0			

ampiran 4 Perhitungan Biaya Penjadwalan = 15 hari menggunakan *Branch and Bound* dengan LINGO **itungan Biaya (T= 30 Hari) (lanjutan)**



	Tujuan			Tujuan	Tujuan			Tujuan			Tujuan	Tujuan			Total
(Rp)	Cf_{ijv} (Rp)	C_{Pik} (Rp)	Cs_{ijv} (Rp)	C_{Pik} (Rp)	Cf_{ijv} (Rp)	C_{Pik} (Rp)	Cs_{ijv} (Rp)	C_{Pik} (Rp)	Cf_{ijv} (Rp)	C_{Pik} (Rp)	Cs_{ijv} (Rp)	Cf_{ijv} (Rp)	C_{Pik} (Rp)	Cs_{ijv} (Rp)	
	TWI,5			SBY,4	SBY,1			BPP,4			TWI,1	KUP,4			
	Premium					0	0			0		42,169,880	0		
,924,195	Kerosine	BPP,3	0	121,546,490	19,080,220	0	111,545,829	0	19,080,220	0	223,860,940	3,547,028	0	0	1,074,982,32
GD 1	Solar		0			0		0	47,695,458	0	19,080,220	24,763,300	0	88,851,692	
	SBY,3			TTB,4	TTM,1						TTB,1				
			0			0									
	Premium		0			0	0					0			1,055,314,45
,751,861	Kerosine	CLC,2	32	234,695	27,580,843.13	0									
GDN			0	196,338,600	27,580,843.13	0	0	139,195,276	27,580,843.13	0		0			159,708,264
	Solar					0						0			
				BPP,2				KUP,2							
	Premium	KUP,1				0				0					1,143,793,80
GD 2	Kerosine			278,175,213	19,080,220.00	0	284,031,534		19,080,220.00	0					170,011,229
	Solar					0						Total			3,274,090,58

	Tujuan	
ik (Rp)	Csijv (Rp)	Cfijv (Rp
		SBY, 7
0		
0	60,003,963	19,080,2
0		