OPTIMASI PEMILIHAN RUTE TRANSPORTASI DENGAN ALGORITMA *SIMULATED ANNEALING*

SANDY KOSASI

Program Studi Sistem Informasi
Sekolah Tinggi Manajemen Informatika dan Komputer Pontianak
sandykosasi@stmikpontianak.ac.id

ABSTRAK

Makalah ini menyajikan suatu model penyelesaian permasalahan optimasi solusi pemilihan rute transportasi terpendek (*Vehicle Routing Problem with Time Windows*-VRPTW) menggunakan algoritma Simulated Annealing (SA) dengan studi kasus pada CV. Indo Makmur. Algoritma SA memiliki kelebihan dimana dapat menghasilkan solusi yang mendekati optimal dalam waktu singkat, mudah dalam penerapannya,serta memiliki mekanisme yang bisa menghindarkannya dari jebakan *local optima*. Metode perancangan aplikasinya menggunakan *Object-Oriented Analysis and Design* (OOAD) dan pemodelan aplikasinya menggunakan *use case diagram*, *sequence diagram* dan *class diagram*. Hasil penelitian ini memperlihatkan melalui algoritma SA dapat menyelesaikan persoalan VRPTW pada CV. Indo Makmur dan dapat mencapai hasil yang optimal berupa pengurangan biaya operasional pendistribusian produk ke pelanggan.

Kata Kunci: **pendistribusian, vehicle routing problem with time windows, simulated annealing, object-oriented analysis and design**

ABSTRACT

This paper provides the problem-solving model of the shortest vehicle routing choice optimization, namely Vehicle Routing Problem with Time Windows (VRPTW) applying Simulated Annealing (SA) algorithm in the case study of CV. Indo Makmur. SA Algorithm has advantages such as giving solution approaching optimum in a short time, being easy in application, and having mechanism that can obviate the trap of local optima. The application design utilizes Object-Oriented Analysis and Design (OOAD) and the application modeling utilizes the use case diagram, sequence diagram and class diagram. The result of this research shows that SA algorithm can solve the problem of VRPTW at CV. Indo Makmur and reach an optimal result, namely the operational cost reduction of product distribution to consumers.

Keywords: ***distribution, vehicle routing problem with time windows, simulated annealing, object-oriented analysis and design***

1. PENDAHULUAN

CV. Indo Makmur merupakan perusahaan yang bergerak dalam usaha pendistribusian produk-produk berupa beras lokal, minyak goreng, air mineral dalam kemasan botol, minuman kaleng dan makanan ringan (snack) yang mencakup area pemasaran untuk seluruh wilayah Kalimantan Barat. CV. Indo Makmur memiliki sejumlah armada kendaraan untuk mengirimkan barang dari gudang kepada pelanggan yang tersebar di berbagai lokasi kota dan kabupaten. Perusahaan ini memasarkan produk multi item dengan sistem melakukan pemesanan terlebih dahulu dan pendistribusiannya dengan sistem kanvas. Saat ini, dalam mendistribusikan produk kepada pelanggannya, pemilihan rute pendistribusian ditentukan secara manual berdasarkan perkiraan mengenai lokasi, jumlah produk pesanan, dan waktu pengirimannya. Mekanisme pendistribusiannya bekerja hanya berdasarkan perkiraan saja dan lebih fokus kepada pengalokasian kendaraan untuk pelanggan pada area yang sama sehingga sering terjadi kapasitas kendaraan berlebih karena tidak adanya standar dalam melakukan pengelompokkan. Padahal seharusnya kapasitas yang berlebih ini dapat dimanfaatkan untuk mensuplai daerah yang tidak hanya berada pada satu area saja. Selain itu, pengelompokkan area ini tidak menghasilkan solusi rute. Rute yang akan ditempuh hanya ditentukan oleh sopir secara *trial* dan *error*. Namun banyaknya jumlah titik distribusi yang tersebar menyulitkan sopir dalam menentukan urutan rute yang akan ditempuh. Salah satu akibatnya adalah jarak yang ditempuh menjadi lebih panjang. Jika hal ini terus berlanjut maka biaya yang dikeluarkan perusahaan untuk mendistribusikan barang menjadi besar [1]. Padahal dengan mengatur kombinasi pelanggan-pelanggan untuk masing-masing kendaraan dengan baik, pemanfaatan kendaraan bisa lebih optimal dan bisa mengurangi jarak tempuh serta menghemat biaya untuk pendistribusian [3].

Untuk itu perlu dilakukan penentuan rute dan penjadwalan pengiriman produk yang dapat mengatasi permasalahan diatas dengan mengoptimalkan seluruh sumber daya yang dimiliki sehingga diperoleh solusi rute yang dapat meminimasi total biaya transportasi [1]. Sejumlah penelitian sudah pernah dilakukan untuk masalah yang berkaitan dengan optimasi solusi pemilihan rute transportasi ini, seperti menggunakan metode konvensional maupun heuristik. Metode konvensional menggunakan perhitungan matematis untuk mencari semua kemungkinan jawaban yang ada. Metode konvensional yang bisa digunakan adalah metode *Branch and Bound* dan metode *Branch and Cut*. Meskipun bisa menemukan solusi terbaik dengan cara ini, metode ini tidak efektif karena proses perhitungan untuk menelusuri semua solusi yang ada akan membutuhkan waktu yang sangat lama, apalagi kalau permasalahan yang dipecahkan lebih kompleks dengan jumlah pelanggan yang banyak. Melalui metode heuristik, solusi yang diperoleh tidak selalu merupakan solusi yang terbaik[7]. Tapi metode heuristik bisa menemukan solusi yang mendekati optimal dalam waktu lebih singkat[3]. Salah satunya adalah algoritma *Simulated Annealing* (SA). Algoritma ini dapat menghindari jebakan local optima yang buruk dan mudah implementasinya [9].

Dalam penelitian sebelumnya mendapatkan bahwa persentase perbaikan yang dihasilkan oleh metode SA jauh lebih besar dari metode *Steepest Descent* (SD). Hal ini menunjukkan kemampuan model SA dalam menemukan solusi sudah hampir mendekati Global Optima [5]. Hasil penelitian ini diperkuat dengan penelitian lainnya, dimana penelitian pada pendistribusian produk suatu perusahaan penghasil produk air minum dalam kemasan menunjukkan bahwa dengan menggunakan metode SA bisa menghasilkan sebuah solusi untuk proses pendistribusian barang yang mencapai titik optimal untuk perusahaan tersebut [6].

2. TINJAUAN PUSTAKA

2.1. *Vehicle Routing Problem*

VRP (*Vehicle Routing Problem*) merupakan salah satu jenis masalah optimasi dalam melakukan pemilihan rute dengan biaya minimal [3]. VRP merupakan *NP-hard problems*, dimana waktu yang dibutuhkan untuk mencari solusi permasalahan bergerak secara ekponensial seiring dengan bertambahnya konsumen. VRP berhubungan dengan penentuan sekumpulan rute optimal yang dilalui oleh armada kendaraan dalam mendistribusikan barang antara depot (gudang) dan pengguna akhir [10]. Adapun varian dari VRP yang diterapkan dalam penelitian ini adalah VRPTW (*Vehicle Routing Problem with Time Windows*). VRPTW adalah pengembangan

dari CVRP (*Capacitated Vehicle Routing Problem*) dimana *servis* pada setiap pelanggan harus mulai dalam *time window* yang bersangkutan dan kendaraan harus tinggal pada lokasi pelanggan selama *servis*. *Soft time windows* bisa dilanggar dengan biaya tambahan tertentu, sedangkan *hard time windows* tidak mengizinkan kendaraan untuk tiba di tempat pelanggan setelah batas waktu terakhir untuk memulai *servis*. Apabila kendaraan tiba sebelum pelanggan siap memulai *servis*, maka kendaraan akan menunggu [3].

VRPTW dapat diilustrasikan dalam graf. Misalkan G = (V, A) adalah graf lengkap, dimana V = {0, ..., n, n + 1} adalah himpunan node (simpul) dan A adalah himpunan arc (sisi). Node i = 1, ..., n mengacu kepada lokasi pelanggan, sedangkan node 0 dan n + 1 mengacu kepada depot. Sebuah nilai nonnegatif, cij, diasosiasikan dengan setiap arc (i, j) ∈ A dan melambangkan biaya perjalanan yang diperlukan untuk pergi dari node i ke node j. Jika cij = cji untuk semua (i, j) ∈ A, maka permasalahannya disebut simetris dan himpunan arc A digantikan oleh himpunan E yang merupakan edge (sisi tak berarah). Masing-masing pelanggan i (i = 1, ... ,n) diasosiasikan dengan nilai permintaan nonnegatif, di, yang harus dikirimkan, dan depot memiliki permintaan d0 = 0. Sekumpulan K kendaraan identik, masing-masing dengan kapasitas C, tersedia di depot dan diasumsikan di ≤ C untuk setiap i = 1, ... , n. Setiap kendaraan boleh menjalani maksimal satu rute, dan diasumsikan bahwa K tidak lebih kecil dari Kmin, dimana Kmin jumlah minimal kendaraan yang diperlukan untuk melayani semua pelanggan. Nilai Kmin dapat ditentukan dengan menyelesaikan Bin Packing Problem (BPP) yang diasosiasikan dengan CVRP [10].

Waktu saat kendaraan meninggalkan depot, waktu perjalanan, tij, untuk setiap arc (i, j) ∈ A dan sebuah waktu servis tambahan si untuk masing-masing pelanggan i juga diberikan. Servis untuk setiap pelanggan harus bermula didalam time window yang bersangkutan, dan kendaraan harus berhenti pada lokasi pelanggan selama waktu si. Dalam kasus kedatangan lebih awal ke lokasi pelanggan i, kendaraan umumnya diizinkan menunggu sampai waktu ai (sampai waktu servisnya mulai). Semua rute kendaraan yang mungkin dapat disamakan dengan jalur pada G yang mulai pada node 0 dan berakhir pada node n + 1. Sebuah time window diasosiasikan dengan node 0 dan n + 1, yaitu [a0,b0] = [an+1, bn+1] = [E, L], dimana E dan L masing-masing melambangkan waktu keberangkatan paling awal dari gudang dan waktu tiba paling lambat di gudang. Permintaan dan waktu servis 0 juga diberikan untuk kedua node ini, yaitu, d0 = dn+1 =s0 = sn+1 = 0. Solusi yang bisa diterima ada hanya jika a0 = E ≤ min∈V¥{0}bi - t0i dan bn+1 = L ≥ mini∈V¥{0} ai + si + ti0. Sebuah arc (i, j) ∈ A bisa dieliminasi karena pertimbangan, jika ai + si + tij > bj,atau batasan kapasitas, jika di + dj > C, atau faktor lain [10].

VRPTW mencakup menemukan sejumlah K sirkuit (masing-masing berhubungan dengan satu rute kendaraan) dengan biaya minimal, yang didefinisikan sebagai jumlah biaya dari arc yang dimiliki sirkuit, dan sedemikian sehingga [10:2]:

a. Setiap sirkuit mengunjungi node depot;
b. Setiap node pelanggan dikunjungi oleh tepat satu sirkuit;
c. Jumlah permintaan dari node yang dikunjungi oleh sebuah sirkuit tidak melebihi kapasitas kendaraan, C; dan
d. Untuk setiap pelanggan i, servis dimulai dalam time window, [ai , bi], dan kendaraan berhenti selama waktu si.

2.2. *Cost Vehicle Routing Problem*

Untuk setiap solusi baru akan dihitung *cost* nya untuk kemudian dibandingkan dengan solusi terbaik yang sebelumnya sudah diperoleh. Berikut ini rumus untuk menghitung *cost* tiap rute [5]:

Ck = Jk + 0.01JkTk +0.02JkWk + 0.5JkPk + 100JkOk +0.1 JkLk

dimana:

Ck = cost untuk rute k
Jk = total jarak yang ditempuh kendaraan pada rute k
Tk = total waktu yang diperlukan untuk menjalani rute k
Wk = total waiting time (apabila kendaraan datang lebih awal dari time window awal)

dari CVRP (*Capacitated Vehicle Routing Problem*) dimana *servis* pada setiap pelanggan harus mulai dalam *time window* yang bersangkutan dan kendaraan harus tinggal pada lokasi pelanggan selama *servis*. *Soft time windows* bisa dilanggar dengan biaya tambahan tertentu, sedangkan *hard time windows* tidak mengizinkan kendaraan untuk tiba di tempat pelanggan setelah batas waktu terakhir untuk memulai *servis*. Apabila kendaraan tiba sebelum pelanggan siap memulai *servis*, maka kendaraan akan menunggu [3].

VRPTW dapat diilustrasikan dalam graf. Misalkan G = (V, A) adalah graf lengkap, dimana V = {0, ..., n, n + 1} adalah himpunan node (simpul) dan A adalah himpunan arc (sisi). Node i = 1, ..., n mengacu kepada lokasi pelanggan, sedangkan node 0 dan n + 1 mengacu kepada depot. Sebuah nilai nonnegatif, cij, diasosiasikan dengan setiap arc (i, j) ∈ A dan melambangkan biaya perjalanan yang diperlukan untuk pergi dari node i ke node j. Jika cij = cji untuk semua (i, j) ∈ A, maka permasalahannya disebut simetris dan himpunan arc A digantikan oleh himpunan E yang merupakan edge (sisi tak berarah). Masing-masing pelanggan i (i = 1, ... ,n) diasosiasikan dengan nilai permintaan nonnegatif, di, yang harus dikirimkan, dan depot memiliki permintaan d0 = 0. Sekumpulan K kendaraan identik, masing-masing dengan kapasitas C, tersedia di depot dan diasumsikan di ≤ C untuk setiap i = 1, ... , n. Setiap kendaraan boleh menjalani maksimal satu rute, dan diasumsikan bahwa K tidak lebih kecil dari Kmin, dimana Kmin jumlah minimal kendaraan yang diperlukan untuk melayani semua pelanggan. Nilai Kmin dapat ditentukan dengan menyelesaikan Bin Packing Problem (BPP) yang diasosiasikan dengan CVRP [10].

Waktu saat kendaraan meninggalkan depot, waktu perjalanan, tij, untuk setiap arc (i, j) ∈ A dan sebuah waktu servis tambahan si untuk masing-masing pelanggan i juga diberikan. Servis untuk setiap pelanggan harus bermula didalam time window yang bersangkutan, dan kendaraan harus berhenti pada lokasi pelanggan selama waktu si. Dalam kasus kedatangan lebih awal ke lokasi pelanggan i, kendaraan umumnya diizinkan menunggu sampai waktu ai (sampai waktu servisnya mulai). Semua rute kendaraan yang mungkin dapat disamakan dengan jalur pada G yang mulai pada node 0 dan berakhir pada node n + 1. Sebuah time window diasosiasikan dengan node 0 dan n + 1, yaitu [a0,b0] = [an+1, bn+1] = [E, L], dimana E dan L masing-masing melambangkan waktu keberangkatan paling awal dari gudang dan waktu tiba paling lambat di gudang. Permintaan dan waktu servis 0 juga diberikan untuk kedua node ini, yaitu, d0 = dn+1 =s0 = sn+1 = 0. Solusi yang bisa diterima ada hanya jika a0 = E ≤ min∈V¥{0}bi - t0i dan bn+1 = L ≥ mini∈V¥{0} ai + si + ti0. Sebuah arc (i, j) ∈ A bisa dieliminasi karena pertimbangan, jika ai + si + tij > bj,atau batasan kapasitas, jika di + dj > C, atau faktor lain [10].

VRPTW mencakup menemukan sejumlah K sirkuit (masing-masing berhubungan dengan satu rute kendaraan) dengan biaya minimal, yang didefinisikan sebagai jumlah biaya dari arc yang dimiliki sirkuit, dan sedemikian sehingga [10:2]:

a. Setiap sirkuit mengunjungi node depot;
b. Setiap node pelanggan dikunjungi oleh tepat satu sirkuit;
c. Jumlah permintaan dari node yang dikunjungi oleh sebuah sirkuit tidak melebihi kapasitas kendaraan, C; dan
d. Untuk setiap pelanggan i, servis dimulai dalam time window, [ai , bi], dan kendaraan berhenti selama waktu si.

2.2. *Cost Vehicle Routing Problem*

Untuk setiap solusi baru akan dihitung *cost* nya untuk kemudian dibandingkan dengan solusi terbaik yang sebelumnya sudah diperoleh. Berikut ini rumus untuk menghitung *cost* tiap rute [5]:

Ck = Jk + 0.01JkTk +0.02JkWk + 0.5JkPk + 100JkOk +0.1 JkLk

dimana:

Ck = cost untuk rute k
Jk = total jarak yang ditempuh kendaraan pada rute k
Tk = total waktu yang diperlukan untuk menjalani rute k
Wk = total waiting time (apabila kendaraan datang lebih awal dari time window awal)

Pk = selisih antara Tk dengan waktu maksimal yang diizinkan untuk kendaraan tersebut (jika Tk lebih besar)
Ok= Overload (kelebihan muatan pada kendaraan)
Lk = total keterlambatan pada rute k
Nilai 0.01, 0.02, 0.5, 100, dan 0.1 adalah weight factor.

2.3. Algoritma *Simulated Annealing*

Simulated Annealing (SA) merupakan suatu metaheuristik dengan metode pencarian lokal secara acak, dimana modifikasi untuk solusi saat ini yang menyebabkan peningkatan biaya solusi bisa diterima dengan peluang tertentu. Mekanisme ini memungkinkan metode ini untuk lolos dari local optima yang buruk [4:2]. Dalam hubungan algoritma SA dengan VRP, merupakan sebuah solusi atau sekumpulan rute dapat disamakan dengan *state* (tingkat energi) dan biaya solusi dapat disamakan dengan energinya [9]. Pada setiap iterasi, solusi saat ini diubah dengan modifikasi yang dipilih secara acak berdasarkan kelas khusus dari modifikasi yang menggambarkan struktur *neighborhood* (tetangga). Jika solusi baru lebih baik dari solusi saat ini, ia akan menjadi solusi saat ini. Jika tidak, solusi baru diterima berdasarkan kriteria probabilistik, dimana modifikasi lebih mungkin diterima jika sebuah parameter yang disebut *temperature* (berdasarkan analogi dengan proses fisik) bernilai tinggi dan peningkatan biaya bernilai rendah. Selama prosedur berlangsung, parameter *temperature* secara bertahap diturunkan berdasarkan *cooling schedule* (jadwal pendinginan) yang telah ditetapkan terlebih dahulu, dan sejumlah iterasi akan dilakukan pada tiap tingkatan *temperature* [4]. Ketika *temperature* bernilai cukup rendah, hanya modifikasi yang meningkat (lebih baik) yang bisa diterima dan metode akan berhenti pada *local optimum* [3]. Adapun langkah-langkah dalam penyelesaian algoritma SA secara umum adalah [7]:

a. Evaluasi keadaan awal. Jika keadaan awal merupakan tujuan, maka pencarian berhasil dan KELUAR. Jika tidak demikian, lanjutkan dengan menetapkan keadaan awal sebagai kondisi sekarang.
b. Inisialisasi BEST_SO_FAR untuk keadaan sekarang.
c. Inisialisasi T sesuai dengan *annealing schedule.*
d. Kerjakan hingga solusi ditemukan atau sudah tidak ada operator baru lagi akan diaplikasikan ke kondisi sekarang.
    1. Gunakan operator yang belum pernah digunakan tersebut untuk menghasilkan kondisi baru.
    2. Evaluasi kondisi yang baru dengan menghitung:
       ΔE = nilai sekarang – nilai keadaan baru.
       - Jika kondisi baru merupakan tujuan, maka pencarian berhasil dan KELUAR.
       - Jika bukan tujuan, namun memiliki nilai yang lebih baik daripada kondisi sekarang, maka tetapkan kondisi baru sebagai kondisi sekarang. Demikian pula tetapkan BEST_SO_FAR untuk kondisi yang baru tadi.
       - Jika nilai kondisi baru tidak lebih baik dari kondisi sekarang, maka tetapkan kondisi baru sebagai kondisi sekarang dengan probabilitas: $p' = e^{-\Delta E / T}$
         Langkah ini biasanya dikerjakan dengan membangkitkan suatu bilangan random r pada range [0 1]. Jika $r < p'$, maka perubahan kondisi baru menjadi kondisi sekarang diperbolehkan. Namun jika tidak demikian, maka tidak akan dikerjakan apapun.
    3. Perbaiki T sesuai dengan *annealing scheduling.*
e. BEST_SO_FAR adalah jawaban yang dimaksudkan.

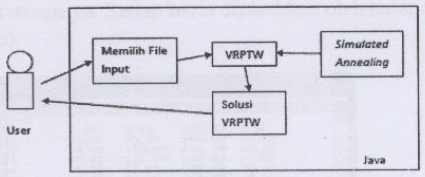
3. METODOLOGI PENELITIAN

Penelitian ini menggunakan bentuk studi kasus dengan metode penelitian dan pengembangan (*research and development*). Metode pengumpulan datanya melalui wawancara dan observasi. Wawancara dengan sejumlah responden terpilih dengan menggunakan teknik *purposive sampling.* Respondenya berasal dari pimpinan, manajemen dan staf/karyawan. Metode analisis dan perancangan menggunakan metode OOAD (*Object-Oriented Analysis and Design*).

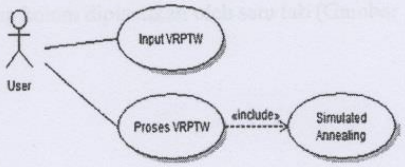
Untuk tahap analisis, metode ini memeriksa *requirement* (merupakan syarat/keperluan) yang harus dipenuhi sebuah sistem dari sudut pandang kelas-kelas dan objek-objek yang ditemui dalam ruang lingkup perusahaan. Sementara tahap desain, metode ini untuk mengarahkan arsitektur software yang didasarkan pada manipulasi objek-objek sistem atau subsistem [8].

4. HASIL PENELITIAN

Perancangan arsitektur aplikasi tersebut terlihat bahwa pada awalnya, user akan memilih file input untuk sistem. Selanjutnya oleh sistem, input tersebut akan diproses ke dalam bentuk VRPTW. Kemudian VRPTW akan diselesaikan oleh algoritma SA untuk menghasilkan solusi VRPTW. Selanjutnya solusi itu akan ditampilkan. Aplikasi penerapan VRPTW dengan algoritma SA akan dibangun dengan menggunakan bahasa pemrograman Java (Gambar 1). Berikut ini sistem dari aplikasi tersebut dimodelkan menggunakan *use case diagram*, *sequence diagram*, dan *class diagram*. Dalam *use case diagram* aktor yang berperan dalam sistem adalah user. User memiliki *use case* Input VRPTW dan Proses VRPTW, dimana *use case* Proses VRPTW meng-include *use case* lainnya dalam hal ini adalah algoritma SA (Gambar 2).

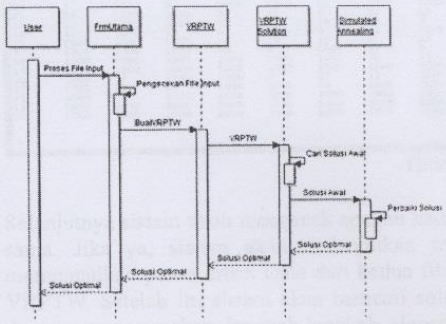


Gambar 1

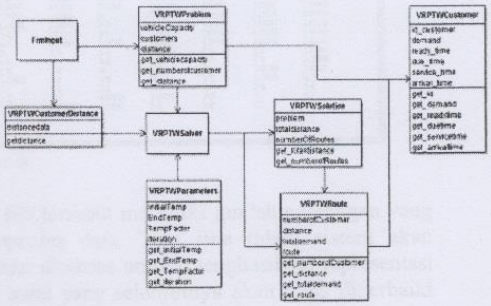


Gambar 2

Ketika user memilih untuk memproses file input, FrmUtama akan melakukan pengecekan pada File Input. Jika File Input sudah benar, maka FrmUtama akan memproses File Input tersebut ke bentuk VRPTW. Selanjutnya, dari VRPTW yang ada akan dicari solusi awalnya. Selanjutnya menggunakan sequence diagram untuk memperlihatkan solusi awal akan terus diperbaiki dengan algoritma SA sampai dihasilkan solusi yang optimal. Solusi optimal ini kemudian akan ditampilkan ke user (Gambar 3) dan class diagram (Gambar 4).



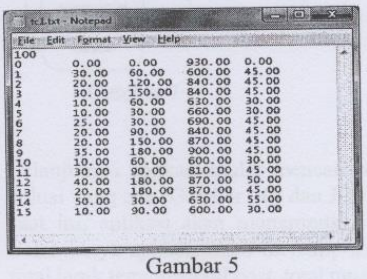
Gambar 3



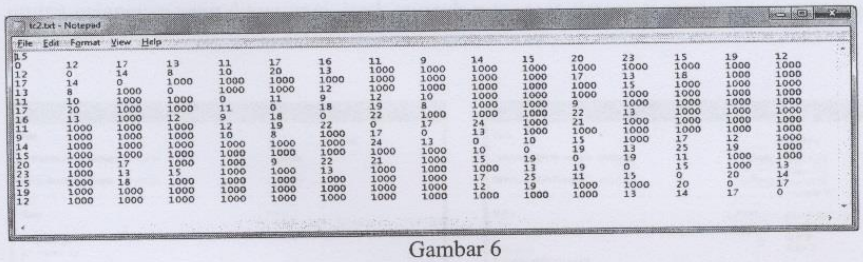
Gambar 4

Program aplikasi ini akan menerima input dari user berupa file. Untuk setiap masalah atau kasus, ada dua file yang digunakan, yaitu file data pelanggan dan file data jarak antar pelanggan. Kedua file memiliki ekstensi .txt. Setelah memilih kedua file tersebut, perangkat lunak akan mengecek apakah jumlah pelanggan yang ada pada kedua file adalah sama. Jika tidak, perangkat lunak akan memunculkan pesan error. Jika ya, perangkat lunak akan melanjutkan untuk memproses kedua file input tersebut ke dalam bentuk VRPTW yang selanjutnya akan

diselesaikan dengan menggunakan algoritma SA. Hasil keluaran dari aplikasi ini adalah jumlah kendaraan yang digunakan, total muatan, total jarak, total cost, pembagian rute masing-masing kendaraan beserta detilnya yaitu total muatan, total jarak, dan total cost untuk masing-masing rute. File inputan dari user berupa dua file .txt, yaitu satu file berisi data pelanggan dan satu file berisi data jarak antar pelanggan. File data pelanggan terdiri dari dua bagian. Bagian pertama adalah baris pertama yang menunjukkan kapasitas kendaraan. Bagian kedua dimulai dari baris kedua berupa bentuk seperti tabel yang terdiri dari lima kolom yang dipisahkan satu tab, sedangkan jumlah barisnya bergantung pada jumlah pelanggan. Antara tiap baris dipisahkan oleh enter. Baris pertama dari bagian kedua menunjukkan gudang, sedangkan baris-baris berikutnya menunjukkan pelanggan. Kelima kolom pada inputan tersebut masing-masing menunjukkan id pelanggan, jumlah permintaan, batas waktu awal, batas waktu akhir, dan waktu servis. Waktu servis diinputkan dalam satuan menit. Sedangkan untuk batas waktu awal dan batas waktu akhir adalah jumlah menit setelah kendaraan mulai meninggalkan depot (Gambar 5). File data jarak antar pelanggan dibuat dalam dua bagian. Bagian pertama menunjukkan jumlah pelanggan (tidak termasuk gudang). Sedangkan baris-baris berikutnya menunjukkan matriks jarak antar pelanggan dengan catatan baris pertama dan kolom pertama dimulai dengan gudang, pelanggan 1, dan seterusnya. Setiap baris dipisahkan oleh enter dan setiap kolom dipisahkan oleh satu tab (Gambar 6).



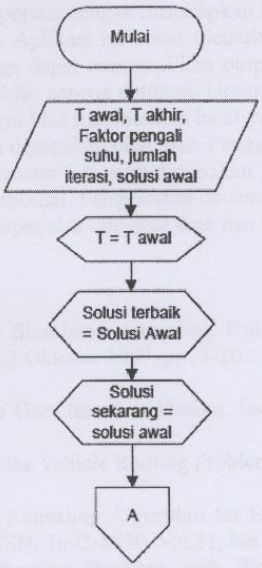
Gambar 5



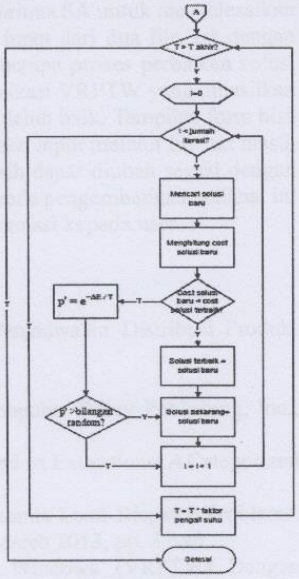
Gambar 6

Selanjutnya sistem akan mengecek apakah kedua file tersebut memiliki jumlah pelanggan yang sama. Jika ya, sistem akan melanjutkan memproses data. Tapi jika tidak, sistem akan memunculkan pesan error. Data dari kedua file akan diproses untuk menghasilkan representasi VRPTW. Setelah itu sistem akan mencari solusi awal yang selanjutnya akan terus diperbaiki dengan menggunakan langkah-langkah algoritma SA. Menjalankan algoritma SA diperlukan solusi awal yang seterusnya akan diperbaiki sampai mendapat solusi yang mendekati optimal. Berikut ini adalah flowchart untuk mencari solusi awal (Gambar 7). Berikut flowchart dari aplikasi untuk menyelesaikan VRPTW dan akan digunakan algoritma SA (Gambar 8) dan algoritma SA lanjutan (Gambar 9).

Gambar 7

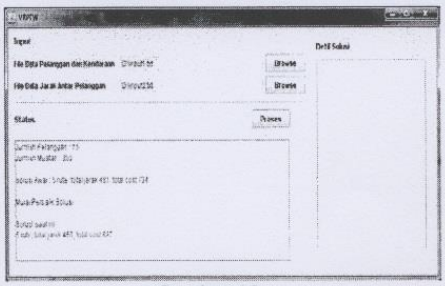


Gambar 8

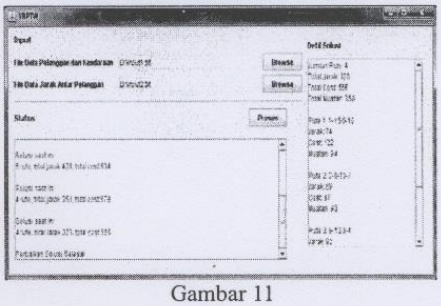


Gambar 9

Selanjutnya aplikasi mulai mencari solusi awal dan menampilkan hasil dari proses perbaikan solusi yang ada (Gambar 10) dan Jika ada solusi baru yang lebih baik atau bisa dijadikan solusi saat ini, aplikasi akan menampilkan solusi tersebut sampai proses perbaikan solusi selesai. Selanjutnya aplikasi akan menampilkan detil solusi yang terdiri atas jumlah rute (kendaraan), total jarak tempuh, total *cost*, total muatan, beserta detil masing-masing rute. Detil rute terdiri atas urutan pelanggan yang dikunjungi, jarak tempuh rute, *cost* rute, serta total muatan rute tersebut (Gambar 11).



Gambar 10



Gambar 11

5. KESIMPULAN

Aplikasi optimasi pemilihan rute transportasi dengan menerapkan algoritma SA untuk menyelesaikan VRPTW sudah sesuai dengan data inputannya. Aplikasi ini dapat menerima input dari dua file .txt dengan format yang sesuai. Selanjutnya aplikasi ini juga dapat menampilkan output berupa proses perbaikan solusi yang dilakukannya secara mendetil dan solusi akhir beserta detilnya. Untuk aplikasi VRPTW yang dihasilkan ini masih dapat dikembangkan lebih lanjut supaya bisa memberikan hasil yang lebih baik. Tampilan form bisa dikembangkan supaya lebih nyaman dan mudah digunakan oleh user. Penggunaan input melalui file .txt masih rentan akan kesalahan dan kurang praktis. Parameter SA yang digunakan masih dapat diubah sesuai dengan permasalahan untuk menghasilkan solusi yang optimal. Penggunaan database pada pengembangan aplikasi ini selanjutnya untuk mempermudah input dan memperjelas tampilan data dan informasi kepada user.

DAFTAR PUSTAKA

[1] Eri Wirdianto dkk. Penerapan Algoritma Simulated Annealing Pada Penjadwalan Distribusi Produk. *Jurnal Optimasi Sistem Industri*, Vol.7, No.1 Oktober 2007, pp. 7-20.

[2] Galitz, Wilbert O. The Essential Guide to User Interface Design. Indianapolis: Wiley Publishing, Inc., 2007.

[3] Gendreau, Michael dkk. Metaheuristics for the Vehicle Routing Problem and its Extensions: A Categorized Bibliography, 2007.

[4] Hardiansyah dkk. An Efficient Simulated Annealing Algorithm for Economic Load Dispatch Problems. *Jurnal Telkomnika* Terakreditasi dengan ISSN: 1693-6930, Vol.11, No.1 March 2013, pp. 37-46.

[5] Hun, Donald. Studi Tentang Vehicle Routing Problem with Time Windows (VRPTW) Dengan Menggunakan Metode Simulated Annealing, *Skripsi*, Fakultas Teknologi Industri, Jurusan Teknik Industri, Universitas Kristen Petra, Surabaya. 2005.

[6] Juniarto, Susilo Dwi. Optimasi Distribusi Barang Berdasarkan Rute dan Daya Tampung Menggunakan Metode Simulated Annealing, 2011.

[7] Kusumadewi, Sri.dkk. Aplikasi Simulated Annealing untuk Penentuan Tata Letak Mesin. *Seminar Nasional Aplikasi Teknologi Informasi (SNATI) 2004*, Yogyakarta, 2004.

[8] Mathiassen, Lars., et al. Object-Oriented Analysis & Design, Marko Publishing, Grand Rapids, 2000.

[9] Ratnasari, Aditya, dkk. Implemenetasi Algoritma Simulated Annealing Dalam Penyelesaian Masalah Penjadwalan Produksi Jenis Flowshop. *Jurnal Informatika* Vol.3 No.2 November 2007.

[10] Toth, P., dan Vigo, D. The Vehicle Routing Problem, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 2002.