

ALGORITMA PALGUNADI UNTUK MENYELESAIKAN SINGLE DAN MULTI PRODUCT VEHICLE ROUTING PROBLEM

Ika Tofika Rini¹, Y.S. Palgunadi², Bambang Harjito³

Program Studi Informatika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Sebelas Maret

Jl. Ir. Sutami No. 36 A Surakarta

Telp: (0271) 646994, 646761

E-mail: ¹ikatofikarini@gmail.com, ²palgunadi@uns.ac.id, ³harjitob2011@gmail.com

ABSTRAKS

Vehicle Routing Problem dengan Single dan Multi Product merupakan permasalahan perencanaan routing transportasi distribusi barang dengan satu dan lebih dari satu jenis komoditi barang. Permasalahan routing ini merupakan permasalahan yang memberikan banyak perbaikan dalam berbagai segi seperti mengetahui rute, penjadwalan pengiriman barang, juga mengenai pemakaian fasilitas kendaraan. Saat ini routing dilakukan tidak mempertimbangkan area customer secara keseluruhan dan tidak memaksimalkan kapasitas dari kendaraan yang digunakan dalam pendistribusiannya. Penelitian ini mengusulkan Algoritma Palgunadi sebagai algoritma baru untuk menentukan routing dalam kasus single dan multi product. Berdasarkan hasil penelitian yang diaplikasikan dalam program Java, dapat diperoleh hasil rute dan jumlah kendaraan yang digunakan sesuai dengan perhitungan manual yang telah dilakukan sebelumnya. Algoritma Palgunadi juga dapat digunakan untuk menyelesaikan kasus dalam skala besar dibuktikan dengan running time. Hubungan banyaknya agen yang harus dilayani per running time program menunjukkan kondisi kuadratik sedangkan hubungan banyaknya agen per banyaknya kendaraan menunjukkan kondisi linear. Algoritma ini juga dapat menyelesaikan kondisi infeasible. Dengan demikian Algoritma Palgunadi ini dapat diajukan sebagai salah satu algoritma alternatif pemecahan masalah penentuan rute kendaraan untuk kasus Vehicle Routing Problem dengan Single Product dan Multi Product.

Kata Kunci: Vehicle Routing Problem, Algoritma Palgunadi

1. PENDAHULUAN

Perkembangan zaman di era teknologi yang semakin canggih menjadi pendorong dalam peradaban dunia, tak terkecuali dalam dunia industri. Di dunia industri terdapat masalah *routing* yang merupakan masalah dalam penentuan rute pada jalur distribusi. VRP merupakan *hard combinationatorial optimization problem* yang erat kaitannya dengan *Travelling Salesman Problem* (TSP), sehingga adanya permasalahan ini algoritma heuristik menjadi salah satu alternatif yang dikembangkan. Dimana rute itu sendiri merupakan jalur distribusi yang diawali dan berakhir pada suatu depot. Permasalahan *routing* ini merupakan permasalahan yang memberikan banyak perbaikan dalam berbagai segi seperti mengetahui rute, penjadwalan pengiriman barang, juga mengenai pemakaian fasilitas kendaraan yang tersedia menurut (Rahayu, Ragil. 2012). Salah satu permasalahan perencanaan transportasi adalah penentuan rute dan jadwal kendaraan yang secara umum dikenal dengan istilah masalah penentuan rute kendaraan atau *Vehicle Routing Problem* menurut (Moolman *et al.* 2010).

Efisiensi dan efektifitas dalam suatu sistem rantai suplai merupakan kunci utama perusahaan dalam meningkatkan daya saing. Dalam struktur biaya suatu sistem rantai suplai, biaya transportasi mendominasi keseluruhan biaya yang dikeluarkan menurut (Ong, J.O. 2011). Perencanaan yang baik

memberikan penghematan yang signifikan terhadap total biaya yang dikeluarkan seperti yang ditulis Sehingga pendistribusian barang dari depot perusahaan menggunakan kendaraan menuju ke agen-agen membutuhkan perhitungan eksak yang konkrit. Dimana *routing* kendaraan menjadi faktor yang sangat penting di dalam VRP. VRP selain bertujuan untuk meminimalkan total jarak atau total biaya transportasi, dapat juga untuk meminimalkan jumlah kendaraan yang digunakan. Sehingga dengan jumlah kendaraan yang optimal maka akan dapat mengurangi pengeluaran perusahaan.

Masing-masing kendaraan melayani beberapa agen dengan kapasitas angkut tertentu dan setiap agen memiliki *demand* tertentu pula. Setiap pelanggan hanya boleh dikunjungi sekali dan total *demand* tidak boleh melebihi kapasitas angkut kendaraan yang dipakai. Permasalahan yang ada saat ini adalah bahwa perusahaan tidak memperhatikan rute dalam jangkauan area yang lebih besar serta tidak memaksimalkan kapasitas kendaraan.

Penelitian (Sarngadi, Palgunadi & Putri, E L.A. 2014) menggunakan algoritma Palgunadi untuk memperbaiki algoritma Tabu Search dalam menyelesaikan kasus VRP *with Time Window* dengan banyaknya produk hanya satu jenis. Diharapkan bahwa algoritma Palgunadi dapat digunakan untuk menyelesaikan VRPTW dan dapat diterapkan untuk kasus yang berbeda. Algoritma ini

terbukti dapat menghemat total waktu perjalanan dan mengurangi jumlah *customer* yang pengirimannya mengalami keterlambatan. *VRP with Single Product* yang pernah dilakukan (Slamet *et al.* 2014) menggunakan algoritma genetika untuk memecahkan masalah *routing* pendistribusian sayuran di dataran tinggi. Penelitian ini memiliki batasan masalah mencakup *heterogenous multifleet*, *single trip*, *single depot*, dan *delivery operation* tanpa *time window*. Di dalam penelitian tersebut dihasilkan jalur pendistribusian yang mendekati optimal.

Algoritma heuristik lain yaitu algoritma heuristik *clustering genetic* dengan menggunakan *k-means* untuk mendapatkan *hamiltonian cycle* untuk mengidentifikasi klusternya. Untuk proses pengetesan menggunakan Lindo dan CPLEX *Algorithm* dan menghasilkan hasil yang maksimal pada skala kecil (Chang, Yaw & Chen, Lin. 2007).

Melihat permasalahan di atas maka perlu dibuat rute pendistribusian yang dapat meminimumkan jumlah kendaraan dan memaksimumkan kapasitas kendaraan yang digunakan dalam proses pendistribusian dengan mengusulkan algoritma yaitu Algoritma Palgunadi.

Pada penelitian ini diasumsikan bahwa hanya ada satu buah depot yang memasok seluruh *demand customer* dengan kendaraan yang digunakan hanya terdiri dari satu jenis saja, sehingga kapasitas kendaraan yang digunakan pun sejenis. *Time stamp* atau waktu yang diperlukan dalam pendistribusian tidak diperhatikan. Barang yang didistribusikan juga hanya terdiri dari satu jenis untuk kasus *single product* dan ada lebih dari satu jenis barang untuk kasus *multi product*. Syarat dari VRP di antaranya: masing-masing kendaraan melayani beberapa pelanggan dengan kapasitas angkut tertentu, setiap pelanggan/*customer* memiliki *demand* tertentu yang diketahui, setiap pelanggan hanya boleh dikunjungi sekali dan total *demand* tidak boleh melebihi kapasitas angkut kendaraan yang dipakai dan juga setiap kendaraan harus berangkat dan kembali pada depot yang sama.

1.1 Penelitian Terkait

Beberapa penelitian terkait kasus VRP dalam penentuan rute untuk pendistribusian yang telah dilakukan, dijelaskan pada Tabel 1.

Tabel 1. Penelitian terkait, tujuan, metode, kelebihan, dan kelemahan

No	Penulis	Tujuan	Metode	Kelebihan	Kelemahan
1	Arunya Bookleaw, Nanthi Suthikarnnaru nai, dan Raawinkhan Srinon (2009)	Meminimalkan <i>total cost</i> namun tetap memperhatikan <i>constraint</i> mengenai kapasitas dan <i>time window</i> untuk mendistribusikan koran pagi dengan tepat waktu.	Menggunakan algoritma <i>Sweep</i> dengan mendefinisikan masalah terdiri dari waktu pemberangkatan, rute kendaraan, pusat lokasi pendistribusian, dan mengalokasikan <i>drop-off points</i> ke pusat distribusi.	Dapat menghasilkan rute terbaik untuk pendistribusian koran pagi agar dapat diterima oleh pelanggan tepat waktu dengan mengaplikasikan <i>Modified Sweep Method</i> .	<i>Demand</i> masing-masing pelanggan tidak dapat di-split.
2	S.R. Venkatesan, D. Logendran dan D. Chandramohan (2011)	Memperbaiki rute untuk melayani pelanggan dengan jarak minimum dan kapasitas maksimum.	Menggunakan algoritma <i>sweep</i> , <i>Clark and Wright</i> untuk membentuk <i>cluster</i> kemudian menggunakan metode <i>Particle Swarm Optimization</i> (PSO) untuk menemukan jarak optimal rute kendaraan.	Metode <i>sweep</i> terbukti menghasilkan kendaraan lebih sedikit dibanding metode <i>Clark and Wright</i> .	Tidak menunjukkan efisiensi dari masing-masing metode yang digunakan.
3	Sangheon Han dan Yoshio Tabata (2002)	Membuat rute dengan jarak terpendek dengan kapasitas maksimal kendaraan.	Menggunakan penggabungan <i>Genetic Algorithm</i> (GA) dengan algoritma <i>Sweep</i> .	Menampilkan akurasi masing-masing perhitungan menggunakan algoritma <i>Sweep</i> , GA, dan <i>Hybrid GA</i> .	Hanya fokus pada kecepatan algoritma.
4	Yuceer (2013)	Menentukan jumlah kendaraan untuk transportasi semua penumpang dengan minimum <i>cost</i> .	Menggunakan <i>Knapsack</i> dan <i>Sweep</i> .	Penggunaan algoritma <i>sweep</i> dapat dengan kasus skala besar.	<i>Knapsack solution</i> belum dipraktekan ke dalam kasus dengan skala besar.
5	N. Suthikarnnaru nai (2008)	Meminimalkan jumlah kendaraan.	Menggunakan metode heuristik <i>sweep</i> .	Dapat menangani <i>split-delivery</i> .	<i>Run time</i> terlalu lama.

1.2 VEHICLE ROUTING PROBLEM (VRP)

VRP dikenal sebagai *integer programming* dimana masuk ke dalam kategori NP-Hard *problem* menurut (Chang, Yaw & Chen, Lin. 2007). Di dalam VRP setiap rute kendaraan dimulai pada depot, melayani para pelanggan/*customer* pada rute tersebut, dan kembali ke depot. Rute ini harus memenuhi semua *constraint* yang berkaitan dengan masalah: kapasitas kendaraan, *time stamp*, jumlah agen, serta jarak antar agen. Sehingga seluruh pelanggan dapat dilayani sesuai dengan masing-masing *demand* tidak melebihi kapasitas maksimal kendaraan. VRP bertujuan untuk meminimalkan jarak keseluruhan perjalanan dengan kendaraan sementara mampu melayani semua pelanggan melalui rute-rute kendaraan yang keluar masuk depot menurut (Moolman *et al.* 2010).

Dalam perkembangannya, VRP memiliki beberapa variasi dalam aplikasinya, antara lain:

1. *Capacited Vehicle Routing Problem (CVRP)*, dengan faktor utama yaitu masing-masing kendaraan memiliki kapasitas tertentu. Penelitian (Frutos, Mariano & Tohme, Fernando. 2012) menyebutkan ketika kapasitas kendaraan lebih dari satu jenis maka kendaraan akan dibuat kompartemen sesuai banyaknya jenis barang.
2. *Vehicle Routing Problem with Time Windows (VRPTW)*, merupakan generalisasi dari VRP dengan penambahan *constraint* waktu menurut (Chang, Yaw & Chen, Lin. 2007).
3. *Multiple Agent Vehicle Routing Problem (MAVRP)*, dengan faktor utama yaitu satu distributor memiliki banyak agen. Tujuan utama dari MAVRP ini adalah untuk meminimalkan jumlah kendaraan yang diperlukan dan untuk memaksimalkan jumlah *demands* yang ditransportasikan menurut (Shah, Megha Mihir. 2012)
4. *Vehicle Routing Problem with Pick-Up and Delivering (VRPPD)*, dengan faktor utama yaitu pelanggan/*customer* dimungkinkan mengembalikan barang kepada agen asal. Penelitian (Montane, F.A.T & Galvao, Roberto D. 2006) menggunakan algoritma Tabu Search dimana VRP-SPD adalah salah satu variasi dari *classical vehicle routing problem*. Pengiriman barang disuplai dari satu depot pada titik awal pengiriman, sementara *pick-up* muatan kemudian diambil untuk dikembalikan ke depot. Karakteristik dari VRP –SPD adalah bahwa kendaraan yang digunakan pada suatu rute diisi oleh muatan barang yang dikirim dan muatan barang *pick-up*.
5. *Split Delivery Vehicle Routing Problem (SDVRP)*, merupakan variasi dari *capacitated vehicle routing problem (CVRP)* [15]. SDVRP memperbolehkan *customer* untuk dilayani dengan *multiple routes* menurut (Montane, F.A.T & Galvao, Roberto D. 2006).

6. *Stochastic Vehicle Routing Problem (SVRP)*, dengan faktor utama yaitu munculnya *random value* (jumlah pelanggan, jumlah permintaan, waktu pelayanan atau waktu perjalanan). Penelitian mengenai *stochastic demands* pernah dilakukan (Shanmugam *et al.* 2011) menyebutkan bahwa pada kondisi nyata *demand* dari konsumen merupakan bukan prioritas, penelitian ini bertujuan untuk meminimalkan *total cost* dari rute yang terbentuk dengan bergantung pada *random demands*

7. *Periodic Vehicle Routing Problem (PVRP)*, dengan faktor utama yaitu pengantaran hanya dilakukan di hari tertentu. Tujuan dari PVRP ini adalah untuk meminimalkan total jarak rute dan menyelesaikan permasalahan penentuan jadwal pelayanan *customer*. Masing-masing *customer* memiliki tipikal pelayanan secara periodik, misalnya sekali seminggu atau pada hari kerja menurut (Angelelli, Enrico & Speranza, M.G. 2002; Nguyen, P.K. *et al.* 2011).

1.3 VRP SINGLE PRODUCT

Di dalam VRP *Single Product* hanya terdapat satu jenis produk yang dikirimkan dari depot ke masing-masing agen oleh sejumlah kendaraan. Misalnya saja pendistribusian surat kabar atau bahan bakar minyak. Sehingga masing-masing *vehicle* dengan kapasitas tertentu hanya mengangkut satu jenis komoditi barang yang didistribusikan di setiap rutanya. Berikut ini merupakan penentuan rute kendaraan dalam kasus VRP *Single Product*:

1. Kendaraan berawal dan berakhir pada satu depot.
2. Setiap *customer* hanya dilayani dengan satu rute *vehicle*.
3. Hanya terdapat satu jenis barang.
4. *Demands* masing-masing *customer* tidak melebihi kapasitas maksimum *vehicle*.

Tujuan dari VRP *Single Product* ini adalah untuk mengoptimalkan jarak tempuh *vehicle* dan meminimumkan jumlah *vehicle* yang digunakan namun tetap melayani seluruh *demand* dari *customer*. *Output* yang dihasilkan dari perhitungan ini adalah rute pendistribusian masing-masing *vehicle* dan jumlah *vehicle* yang digunakan.

1.4 VRP MULTI PRODUCT

Produk yang didistribusikan adalah lebih dari satu jenis komoditi barang. Misalnya saja pendistribusian *soft-drinks*, produk makanan, dan lain sebagainya.

Dalam penelitian ini dibuat notasi *k* sebagai macam-macam produk dengan kondisi setiap agen memiliki permintaan produk adalah *k demands*. VRP *Multi Product* dapat diselesaikan dengan mengaplikasikan perhitungan VRP *Single Product* dalam beberapa kali perhitungan sesuai dengan banyaknya jenis barang yang didistribusikan.

Sehingga kendaraan yang digunakan untuk satu rute distribusi adalah sejumlah jenis barang (v_i, k).

Berikut merupakan penentuan rute kendaraan dalam kasus *VRP Multi Product*:

1. Kendaraan berawal dan berakhir pada satu depot.
2. Setiap *customer* hanya dilayani dengan satu rute *vehicle*.
3. Terdapat lebih dari satu jenis barang.
4. Digunakan beberapa kendaraan dalam satu rute pendistribusian tergantung dengan banyaknya jenis barang.
5. Satu *vehicle* hanya mengangkut satu jenis barang.
6. *Demands* masing-masing *customer* tidak melebihi kapasitas maksimum *vehicle*.

Tujuan dari *VRP Multi Product* ini adalah untuk mengoptimalkan jarak tempuh *vehicle* dan meminimumkan jumlah *vehicle* yang digunakan untuk mengangkut sejumlah k jenis barang, namun tetap melayani seluruh *demand* dari *customer*. Dimana *output* yang dihasilkan adalah rute pendistribusian masing-masing *vehicle* sesuai dengan jenis barang dan jumlah *vehicle* yang digunakan.

1.5 MODEL MATEMATIKA VRP

Pendekatan umum [7] untuk menyelesaikan *VRP* dapat dijelaskan ke dalam *integer programming* (IP), diberikan agen sebagai $A = \{0, 1, 2, \dots, m\}$, dan kendaraan yang tersedia di depot sebagai n , dan $x_{i,j}$ menjadi fungsi yang menunjukkan kendaraan v_i yang melayani agen j , i.e

$$x_{i,j} = \begin{cases} 0 \\ 1 \end{cases} \text{ dan } x_{i,j} = 1 \text{ ketika kendaraan } v_i \text{ melayani agen } j \text{ dan sebaliknya akan diset sebagai } 0.$$

Bahwa $x_{i,j} = 1$ berarti kendaraan v_i mengirim produk ke agen j . Di dalam *classical VRP*, kita dapat meminimalisir jumlah kendaraan yang digunakan sehingga $x_{i,j}$ akan dimaksimalkan, dengan demikian fungsi dari *classical VRP* adalah sesuai *constraint* sebagai berikut :

$$\max Z = \sum_{i=1}^n \sum_{j=1}^m x_{i,j} \quad (1)$$

Adapun syarat yang perlu diperhatikan, yakni syarat pertama adalah tidak ada satupun agen yang dilayani oleh dua atau lebih kendaraan, seperti *constraint* berikut :

$$\forall j \quad \sum_{i=1}^n x_{i,j} = 1 \quad (2)$$

Kedua, adalah kendala *demand* yang harus dipenuhi sesuai dengan *constraint* berikut:

1. Panjang lintasan maksimum untuk masing-masing kendaraan v_i adalah m (sesuai banyaknya jumlah agen)

$$\forall i \quad \sum_{j=1}^m x_{i,j} \leq m \quad (3)$$

2. *Total demand* maksimum dari agen oleh kendaraan v_i adalah Q_i [7]

$$\forall i, \quad \sum_{j=1}^m d_j x_{i,j} \leq Q_i \quad (4)$$

3. *Bounding integer variabels* yang digunakan untuk menunjukkan bahwa agen dikunjungi atau tidak adalah sesuai dengan *constraint* berikut:

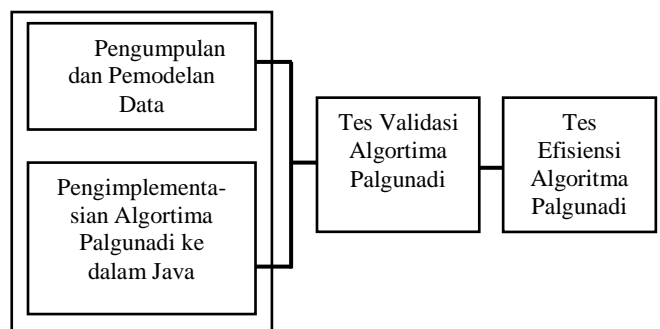
$$\forall i, j \quad x_{i,j} = \begin{cases} 0 \\ 1 \end{cases} \quad (5)$$

dimana :

- A = agen, nilai $A = 0, \dots, m$
- j = agen tujuan, nilai $j = 0, \dots, m$
- n = banyaknya kendaraan, nilai $n = 1, \dots, n$
- v_i = kendaraan i
- $x_{i,j}$ = kendaraan v_i melayani agen j
- Q_j = kapasitas maksimum kendaraan
- m = total kendaraan yang akan dipakai
- d_i = permintaan agen

1.6 METODE PENELITIAN

Metode penelitian ini adalah melalui beberapa tahap, yakni: tahap pengumpulan dan pemodelan data, pembuatan program, tes algoritma Palgunadi, dan tes efisiensi algoritma Palgunadi. Untuk tahapan penelitian ditunjukkan pada Gambar 1.



Gambar 1. Proses tahapan metode penelitian

Pengumpulan data dilakukan dengan mengamati keadaan perusahaan dan sistem distribusinya, juga dengan melakukan wawancara kepada pihak yang bersangkutan. Data yang didapatkan meliputi: jumlah kendaraan, kapasitas kendaraan, jumlah agen, lokasi agen, jarak antar agen, jenis barang yang didistribusikan, dan salah satu data *demand* pada satu kali pengiriman. Tahap pemodelan data dilakukan dengan membuat pemetaan matriks dari data yang telah didapat, meliputi data jarak antar agen dan data jarak depot ke masing-masing agen.

Pembuatan program menggunakan bahasa pemrograman Java dengan mengimplementasikan

algoritma Palgunadi untuk mencari solusi jarak terpendek. Sehingga algoritma ini menghasilkan rute dan jumlah kendaraan yang optimal. Penggunaan algoritma Palgunadi sebagai algoritma baru mampu menyelesaikan masalah routing dengan jumlah customer dalam skala besar. Pada dasarnya semakin banyak customer yang terlibat, maka semakin banyak iterasi yang dilakukan untuk mencari solusi optimal dan hal ini menyebabkan waktu perhitungan (running time) program menjadi lebih lama. Oleh karena itu diusulkan algoritma Palgunadi sebagai solusi untuk menyelesaikan masalah routing dengan customer dalam jumlah besar.

Algoritma Palgunadi dijelaskan dengan *code* pada Gambar 2.

```
i = 1
repeat
minD = MinDemand
if minD ≠ ∞ then
repeat
remainQ = Qi, start = 0
k = NearestNeighbour(start, remainQ)
if k > 0 then {
remainQ = remainQ - dk, dk = 0
totalCost = totalCost + cstart,k
xi,k = 1,
start = k
}
until (remainQ < minD)
i = i + 1
until (k = 0) OR (minD = ∞)
return (i - 1)
```

Gambar 2. Algoritma Palgunadi

Langkah-langkah penyelesaian *routing* kasus *single* dan *multi product* dengan algoritma Palgunadi secara garis besar adalah sebagai berikut:

1. Membuat matriks jarak antar *customer*, selanjutnya disebut agen.
2. Menentukan rute diawali dengan mencari agen terdekat dari depot (pemberangkatan pertama).
3. Mengecek apakah agen memiliki *demand* atau tidak sehingga dapat diputuskan bahwa agen tersebut akan dikunjungi atau tidak.
4. Mengecek apakah *demand* tersebut melebihi kapasitas kendaraan atau tidak. Jika tidak, maka kendaraan akan melayani agen tersebut. Jika melebihi, maka akan mengecek *demand* dari agen terdekat berikutnya.
5. Setelah kendaraan melayani *demand* agen maka dilakukan *update* kapasitas kendaraan.
6. Mencari agen dengan jarak terdekat dari posisi akhir kendaraan.
7. Melayani agen hingga kapasitas kendaraan tidak memenuhi *demand* agen-agen selanjutnya.

8. Proses berlanjut sampai semua agen sudah dikunjungi.

Data *input* yang digunakan disesuaikan dengan kondisi kasus yang dikerjakan. Pada penelitian ini pada dasarnya terdapat dua buah kasus utama yakni *single* dan *multi product*, namun muncul satu buah kondisi dimana kondisi tersebut selanjutnya disebut dengan kondisi *infeasible*. Kondisi ini adalah kondisi dimana *demand* lebih besar dari kapasitas maksimal kendaraan.

Selanjutnya dilakukan tes validasi algoritma Palgunadi dengan dua metode, yaitu tes perhitungan secara manual dan tes menggunakan perhitungan program yang telah diimplementasikan ke dalam bahasa pemrograman Java untuk mengetahui validasi perhitungannya. Untuk kasus *infeasible condition* termasuk ke dalam tipe VRP *Split Delivery*, kondisinya seperti pada *constraint* berikut:

$$\exists j \forall i d_j > Q_i \quad (6)$$

kondisi ini tidak dapat dikerjakan dengan Algoritma Palgunadi sesuai dengan syarat bahwa satu agen hanya dikunjungi oleh satu kendaraan. Solusi yang diusulkan adalah dengan membagi agen tersebut ke dalam sub-agen sehingga agen tersebut tetap dapat dilayani.

Input yang digunakan dalam kasus ini adalah matriks jarak antar agen, *demand* masing-masing agen (terdapat agen dengan *demand* lebih besar dari kapasitas maksimal kendaraan), dan kapasitas maksimal kendaraan. Selanjutnya akan diperoleh hasil *output* yakni rute kendaraan yang melayani agen dan sub-agen dan jumlah kendaraan yang digunakan.

Kemudian dilakukan tes efisiensi algoritma Palgunadi yang bertujuan untuk mengetahui *run time* algoritma Palgunadi untuk menyelesaikan perhitungan. Data yang digunakan meliputi data dengan *demand* terdiri dari 1-3 jenis barang, dengan jumlah agen yang digunakan sejumlah 20-200 dengan kelipatan 10, dan banyaknya agen dengan kondisi *infeasible* kapasitas sejumlah 1-5 agen.

2. PEMBAHASAN

Penelitian ini menggunakan data pendistribusian tabung LPG 3 kg dan 12 kg menggunakan kendaraan truk Mitsubishi L300 dengan kapasitas 300 buah tabung LPG 3 kg dan 100 buah untuk tabung LPG 12 kg. Hasil pengumpulan data diperoleh data *demand* 14 agen terhadap tabung LPG 3 kg dan LPG 12 kg adalah sesuai dalam Tabel 2 data data matriks jarak antar agen pada Tabel 3.

Tabel 2. Keterangan *demand* masing-masing agen

No	Agen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	LPG 3 kg	100	100	300	40	55	50	100	50	80	40	50	70	120	40
2	LPG 12 kg	20	30	40	20	18	16	20	12	16	14	20	25	30	10

Tabel 3. Keterangan demand masing-masing agen

No	Jarak (KM)	Depot	Agen 1	Agen 2	Agen 3	Agen 4	Agen 5	Agen 6	Agen 7	Agen 8	Agen 9	Agen 10	Agen 11	Agen 12	Agen 13	Agen 14
1	Depot	0	8.01	5	10.06	4.09	1.29	0.45	5.29	8.2	2.47	4.89	2.41	3.04	4.75	0.353
2	Agen 1	8.01	0	11.5	3.04	8.37	6.94	8.61	2.27	13.1	5.54	9.82	7.99	8.16	11.3	7.66
3	Agen 2	5	11.5	0	15.4	8.96	5.49	5.32	9.38	9.79	5.97	9.77	4.45	5.33	0.284	5.35
4	Agen 3	10.06	3.04	15.4	0	6.14	11.4	11.1	5.67	11.2	8.94	7.87	12.8	13.5	15	10.9
5	Agen 4	4.09	8.37	8.96	6.14	0	5.04	3.6	6.74	5.5	6.46	2.19	6.16	6.78	8.49	4.2
6	Agen 5	1.29	6.94	5.49	11.37	5.04	0	2.44	4.82	10.2	1.4	6.89	2.42	3.05	4.76	1.64
7	Agen 6	0.449	8.61	5.32	11.1	3.6	2.44	0	5.77	7.84	3.64	4.53	3.62	4.25	4.77	1.66
8	Agen 7	5.29	2.27	9.38	5.67	6.74	4.82	5.77	0	12.9	3.4	9.6	5.85	6.03	9.13	4.93
9	Agen 8	8.2	13.1	9.79	11.2	5.5	10.2	7.84	12.9	0	10.8	3.53	10.5	11.1	9.68	8.54
10	Agen 9	2.47	5.54	5.97	8.94	6.46	1.4	3.64	3.4	10.8	0	7.5	2.45	2.62	5.73	2.12
11	Agen 10	4.89	9.82	9.77	7.87	2.19	6.89	4.53	9.6	3.53	7.5	0	7.19	7.82	9.53	5.24
12	Agen 11	2.41	7.99	4.45	12.8	6.16	2.42	3.62	5.85	10.5	2.45	7.19	0	3.73	4.2	0.994
13	Agen 12	3.04	8.16	5.33	13.5	6.78	3.05	4.25	6.03	11.1	2.62	7.82	3.73	0	4.48	3.33
14	Agen 13	4.75	11.3	0.284	15	8.49	4.76	4.77	9.13	9.68	5.73	9.53	4.2	4.48	0	5.1
16	Agen 14	0.353	7.66	5.35	10.9	4.2	1.64	1.66	4.93	8.54	2.12	5.24	0.994	3.33	5.1	0

2.1 TES VALIDASI ALGORITMA PALGUNADI

Dari data Tabel 1 dan Tabel 2 dilakukan perhitungan manual menggunakan algoritma Palgunadi kemudian didapatkan hasil berupa rute kendaraan menuju ke masing-masing agen dan jumlah kendaraan yang digunakan.

2.1.1 PERHITUNGAN MANUAL VRP SINGLE PRODUCT

Data yang digunakan dalam perhitungan adalah data dari Tabel 1 dengan Demand LPG 3 kg dan matriks jarak pada Tabel 2, dihasilkan penyelesaian berupa rute kendaraan sebagai berikut:

- rute ke-1 memiliki lintasan:
A(0) - A(14) - A(11) - A(5) - A(9) - A(12) - A(0)
- rute ke-2 memiliki lintasan:
A(0) - A(6) - A(4) - A(10) - A(8) - A(13) - A(0)
- rute ke-3 memiliki lintasan:
A(0) - A(2) - A(7) - A(1) - A(0)
- rute ke-4 memiliki lintasan:
A(0) - A(3) - A(0)

dengan jumlah kendaraan yang digunakan sejumlah 4 buah.

2.1.2 PERHITUNGAN MANUAL VRP MULTI PRODUCT

Data yang digunakan dalam perhitungan adalah data dari Tabel 1 dengan Demand LPG 3 kg dan LPG 12 kg serta menggunakan matriks jarak pada Tabel 2 dihasilkan penyelesaian berupa rute kendaraan sebagai berikut:

a. Produk LPG 3 kg

- rute ke-1 memiliki lintasan:
A(0) - A(14) - A(11) - A(5) - A(9) - A(12) - A(0)
- rute ke-2 memiliki lintasan:
A(0) - A(6) - A(4) - A(10) - A(8) - A(13) - A(0)
- rute ke-3 memiliki lintasan:
A(0) - A(2) - A(7) - A(1) - A(0)
- rute ke-4 memiliki lintasan:
A(0) - A(3) - A(0)

b. Produk LPG 12 kg

- rute ke-1 memiliki lintasan:
A(0) - A(14) - A(11) - A(5) - A(9) - A(12)
- rute ke-2 memiliki lintasan:
A(0) - A(6) - A(4) - A(10) - A(8) - A(13) - A(0)
- rute ke-3 memiliki lintasan:
A(0) - A(2) - A(7) - A(1) - A(0)
- rute ke-4 memiliki lintasan:
A(0) - A(3) - A(0)

dengan jumlah kendaraan yang digunakan sejumlah 4 buah.

2.1.3 PERHITUNGAN APLIKASI VRP SINGLE PRODUCT

Data input yang digunakan adalah matriks jarak antar agen dan demand masing-masing agen terhadap satu barang. Jumlah agen dalam perhitungan ini sebanyak 14 agen. Dihasilkan tampilan program seperti pada Gambar 3.

The screenshot shows the 'Single & Multi Product Vehicle Routing Problem with Palgunadi Algorithm' window. It displays a distance matrix for 14 agents (Agen 1 to Agen 14) and two products (LPG 3 kg and LPG 12 kg). The routing results show the optimal path for each product and the total number of vehicles required.

Gambar 3. Hasil perhitungan VRP single product

Hasil perhitungan menggunakan program yang diimplementasikan ke dalam Java menampilkan hasil perhitungan yang sama dengan hasil perhitungan manual untuk kasus VRP *Single Product*.

2.1.4 PERHITUNGAN APLIKASI VRP MULTI PRODUCT

Data *input* yang digunakan adalah matriks jarak antar agen dan *demand* masing-masing agen terhadap dua barang, yaitu LPG 3 kg dan 12 kg. Jumlah agen dalam perhitungan ini sebanyak 14 agen. Dihasilkan tampilan program seperti pada Gambar 4.

The screenshot shows the 'Single & Multi Product Vehicle Routing Problem with Palgunadi Algorithm' window. It displays a distance matrix for 14 agents (Agen 1 to Agen 14) and two products (LPG 3 kg and LPG 12 kg). The routing results show the optimal path for each product and the total number of vehicles required.

Gambar 4. Hasil perhitungan VRP multi product

Hasil perhitungan menggunakan program yang diimplementasikan ke dalam Java tersebut menampilkan hasil perhitungan yang sama dengan hasil perhitungan manual untuk kasus VRP *Multi Product*.

2.2 TES EFISIENSI ALGORITMA PALGUNADI

Aplikasi yang dibangun juga perlu dilakukan tes mengenai efisiensinya dengan mengetahui *run time* dari aplikasi maka dapat diketahui kinerja algoritma Palgunadi yang diterapkan ke dalam VRP. Tes efisiensi ini menggunakan data *random* meliputi data matriks jarak antar agen, banyaknya agen, *demands* masing-masing agen, dan jenis barang.

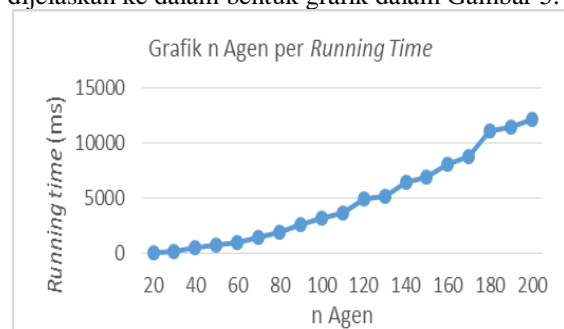
2.2.1 EFISIENSI VRP SINGLE PRODUCT

Tes ini dilakukan dengan data *input* meliputi: jumlah agen antara 20-200 dengan kelipatan 10. Dihasilkan data *running time* dan jumlah kendaraan seperti dalam Tabel 3.

Tabel 3. Hasil perhitungan single product

No	n Agen	Running Time (ms)	n Vehicle
1	20	134	10
2	30	233	16
3	40	492	24
4	50	771	27
5	60	1049	31
6	70	1430	42
7	80	1887	43
8	90	2655	51
9	100	3244	57
10	110	3728	62
11	120	4942	67
12	130	5211	71
13	140	6494	84
14	150	6906	87
15	160	8133	92
16	170	8805	94
17	180	11070	100
18	190	11495	112
19	200	12129	116

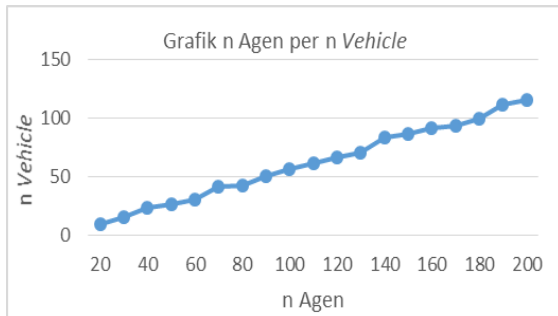
Dari Tabel 3 di atas didapatkan hubungan antara banyaknya agen dengan *running time* kemudian dijelaskan ke dalam bentuk grafik dalam Gambar 5.



Gambar 5. Grafik hubungan n agen per running time

Dari Gambar 5 tersebut didapatkan grafik kuadratik dari hubungan antara banyaknya agen dengan *running time* yang dihabiskan aplikasi untuk menyelesaikan proses perhitungan.

Selanjutnya adalah grafik hubungan antara banyaknya agen dengan banyaknya *vehicle*.



Gambar 6. Grafik hubungan n agen per n vehicle

Dari Gambar 6 didapatkan grafik linear dari hubungan antara banyaknya agen dengan banyaknya *vehicle* yang dibutuhkan untuk melakukan proses pendistribusian.

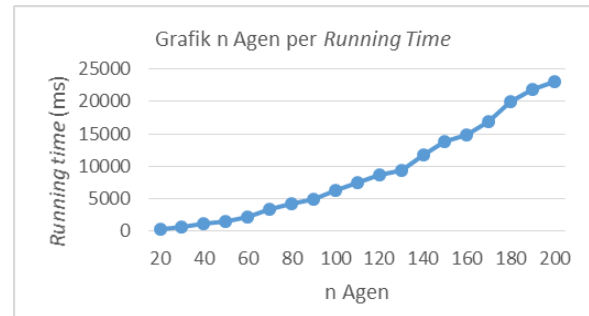
2.2.2 EFISIENSI VRP DUA PRODUK

Didapatkan data *running time* dan banyaknya kendaraan yang dibutuhkan dalam proses pendistribusian dua produk dalam Tabel 4.

Tabel 4. Hasil perhitungan untuk dua produk

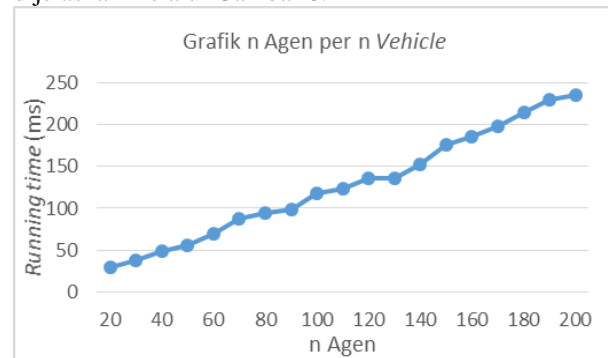
No	n Agen	Running Time (ms)	n Vehicle
1	20	294	29
2	30	574	38
3	40	1143	49
4	50	1558	56
5	60	2235	69
6	70	3331	87
7	80	4256	95
8	90	4926	98
9	100	6327	118
10	110	7547	123
11	120	8702	136
12	130	9392	136
13	140	11710	153
14	150	13895	176
15	160	14770	185
16	170	16875	198
17	180	20022	215
18	190	21780	230
19	200	23148	235

Selanjutnya didapatkan grafik yang menunjukkan hubungan antara banyaknya agen dengan *running time*, dijelaskan pada Gambar 7.



Gambar 7. Grafik hubungan n agen per running time

Dari Gambar 7 didapatkan grafik kuadratik dari hubungan antara banyaknya agen dengan *running time* yang dibutuhkan untuk melakukan proses perhitungan. Selanjutnya adalah hubungan antara banyaknya agen dengan banyaknya *vehicle*, dijelaskan melalui Gambar 8.



Gambar 8. Grafik hubungan n agen per n vehicle

Dari Gambar 8 didapatkan grafik linear dari hubungan antara banyaknya agen dengan *banyaknya vehicle* yang dibutuhkan untuk melakukan proses pendistribusian.

2.2.3 EFISIENSI VRP TIGA PRODUK

Dari hasil perhitungan menggunakan tiga produk maka didapatkan data *running time* dan banyaknya kendaraan yang diperlukan, dijelaskan oleh Tabel 5.

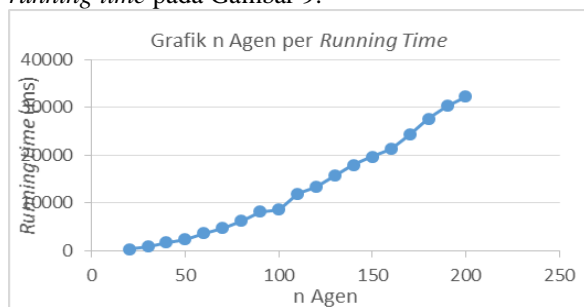
Tabel 5. Hasil perhitungan untuk tiga produk

No	n Agen	Running Time (ms)	n Vehicle
1	20	293	31
2	30	856	53
3	40	1693	76
4	50	2418	86
5	70	4720	125
6	80	6285	142
7	90	8150	165
8	100	8647	171
9	60	3706	114
10	110	11946	187
11	120	13292	207

Tabel 5. Hasil perhitungan untuk tiga produk (lanjutan)

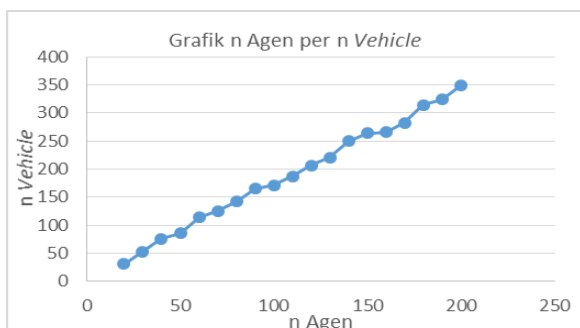
No	n Agen	Running Time (ms)	n Vehicle
12	130	15658	221
13	140	17957	250
14	150	19708	264
15	160	21274	266
16	170	24367	283
17	180	27611	314
18	190	30307	324
19	200	32236	349

Selanjutnya didapatkan grafik yang menunjukkan hubungan banyaknya agen dengan *running time* pada Gambar 9.



Gambar 9. Grafik hubungan n agen per *running time*

Berdasarkan grafik Gambar 9 tersebut didapatkan grafik kuadratik hubungan antara banyaknya agen dengan *running time* yang diperlukan. Selanjutnya adalah hubungan antara banyaknya agen dan banyaknya kendaraan yang dibutuhkan dalam proses pendistribusiannya pada Gambar 10.



Gambar 10. Grafik hubungan n agen per n vehicle

Grafik pada gambar 10 didapatkan grafik linear hubungan antara banyaknya agen dengan banyaknya *vehicle* yang diperlukan untuk menyelesaikan proses pendistribusian.

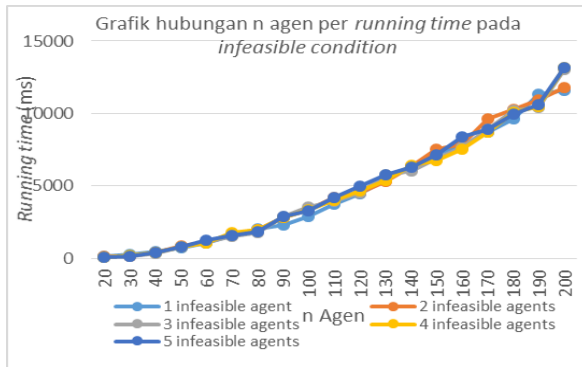
2.2.4 EFSISIENSI VRP *INFEASIBLE* *CONDITION SINGLE PRODUCT*

Perhitungan ini menggunakan satu produk, *demand* dan matriks jarak antar agen dengan kondisi *random*, dan terdapat 1-5 agen dengan kondisi *infeasible*. Dari hasil perhitungan didapatkan data seperti dalam Tabel 6.

Tabel 6. Hasil perhitungan n agen dan *running time* kondisi *infeasible* satu produk

No	n Agen	Running time (ms) dengan agen kondisi <i>infeasible</i> sebanyak 1-5				
		1 Agen	2 Agen	3 Agen	4 Agen	5 Agen
1	20	116	111	102	107	100
2	30	254	151	159	179	170
3	40	456	431	322	415	407
4	50	760	862	821	815	815
5	60	1052	1099	1125	1051	1251
6	70	1560	1600	1536	1757	1586
7	80	2021	1942	1806	1985	1856
8	90	2281	2857	2901	2796	2904
9	100	2860	3354	3562	3347	3309
10	110	3740	4210	3970	3979	4169
11	120	4461	4500	4541	4663	4975
12	130	5441	5327	5791	5397	5738
13	140	6271	6362	6048	6412	6323
14	150	7229	7560	6881	6744	7129
15	160	7707	7874	7652	7546	8356
16	170	8731	9616	8899	8683	8893
17	180	9642	10292	10236	10110	9949
18	190	11365	10938	10398	10501	10641
19	200	11618	11816	13016	13186	13179

Data pada Tabel 6 di atas menunjukkan kondisi *infeasible* dengan banyaknya agen 20-200 dan dengan kondisi *infeasible* sebanyak 1-5 agen. *Running time* yang dihasilkan adalah semakin banyak jumlah agen maka semakin bertambah *running time* yang diperlukan. Untuk mengetahui hubungan antara banyaknya agen dan *running time* selanjutnya dari data pada tabel 7 di atas didapatkan gambar grafik pada Gambar 11 hubungan antara banyaknya jumlah agen dan *running time*.



Gambar 11. Hubungan n agen per running time

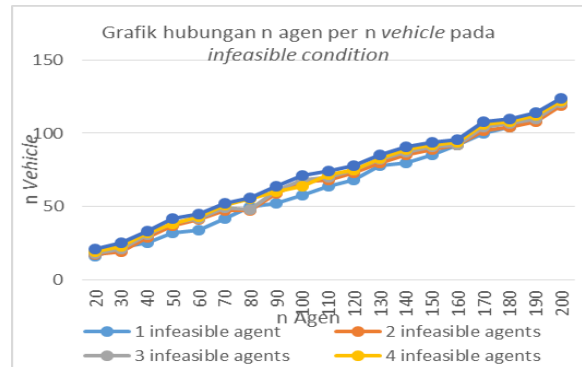
Grafik pada Gambar 11 menunjukkan kondisi kuadratis dari hubungan antara banyaknya agen dengan *running time* yang diperlukan untuk menyelesaikan perhitungan.

Selanjutnya didapatkan juga data banyaknya kendaraan yang diperlukan untuk distribusi, dijelaskan di dalam Tabel 7.

Tabel 7. Hasil perhitungan n agen dan n vehicle kondisi infeasible satu produk

No	n Agen	Running time (ms) dengan agen kondisi infeasible sebanyak 1-5				
		1 Agen	2 Agen	3 Agen	4 Agen	5 Agen
1	20	16	17	18	19	21
2	30	22	19	21	23	25
3	40	25	29	31	32	33
4	50	32	37	39	38	42
5	60	34	41	42	43	45
6	70	42	47	49	51	52
7	80	50	47	48	55	56
8	90	52	59	61	60	64
9	100	58	67	68	64	71
10	110	64	68	70	72	74
11	120	68	73	75	75	78
12	130	78	80	82	83	85
13	140	80	85	87	89	91
14	150	85	89	91	92	94
15	160	92	92	92	94	96
16	170	100	102	104	106	108
17	180	104	104	107	108	110
18	190	114	108	110	112	114
19	200	121	119	121	122	124

Tabel 7 tersebut menjelaskan bahwa semakin banyak agen yang harus dilayani mengakibatkan semakin bertambahnya jumlah kendaraan yang diperlukan untuk distribusi. Hubungan mengenai banyaknya agen dengan banyaknya kendaraan dijelaskan pada Gambar 12.



Gambar 12. Hubungan n agen per n vehicle

Grafik yang dihasilkan dari Gambar 12 tersebut menunjukkan kondisi linear dari hubungan antara banyaknya agen dengan banyaknya kendaraan yang diperlukan.

2.2.5 EFSISIENSI VRP INFEASIBLE CONDITION DUA PRODUK

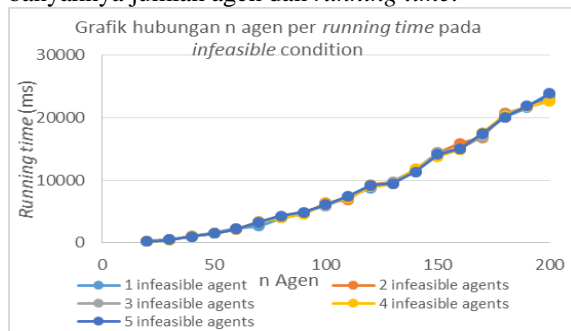
Perhitungan ini menggunakan dua produk, *demand* dan matriks jarak antar agen dengan kondisi *random*, dan terdapat 1-5 agen dengan kondisi *infeasible*. Dari hasil perhitungan didapatkan data seperti dalam Tabel 8.

Tabel 8. Hasil perhitungan n agen dan running time kondisi infeasible dua produk

No	n Agen	Running time (ms) dengan agen kondisi infeasible sebanyak 1-5				
		1 Agen	2 Agen	3 Agen	4 Agen	5 Agen
1	20	328	272	288	267	254
2	30	548	541	519	477	535
3	40	1063	1060	1052	1069	1051
4	50	1562	1540	1560	1502	1529
5	60	2306	2160	2265	2238	2214
6	70	2645	3352	3323	3342	3314
7	80	3940	4152	4194	3960	4295
8	90	4668	4664	4909	4592	4886
9	100	6274	6415	5865	6281	6096
10	110	7360	6914	7296	7156	7481
11	120	8766	9310	9079	8943	9171
12	130	9740	9625	9718	9424	9472
13	140	11616	11474	11719	11899	11354
14	150	14378	14301	14395	13783	14091
15	160	14925	15888	15029	14934	15054
16	170	17336	16743	16924	17558	17436
17	180	20075	20683	20514	20312	20045
18	190	21568	21706	21667	21720	21853
19	200	23492	22861	22704	22658	23834

Data pada Tabel 8 di atas menunjukkan kondisi *infeasible* dengan banyaknya agen 20-200 dan agen dengan kondisi *infeasible* sebanyak 1-5 agen.

Running time yang dihasilkan adalah semakin banyak jumlah agen maka semakin bertambah *running time* yang diperlukan. Untuk mengetahui hubungan antara banyaknya agen dan *running time* selanjutnya dari data pada tabel 7 di atas didapatkan gambar grafik pada Gambar 13 hubungan antara banyaknya jumlah agen dan *running time*.



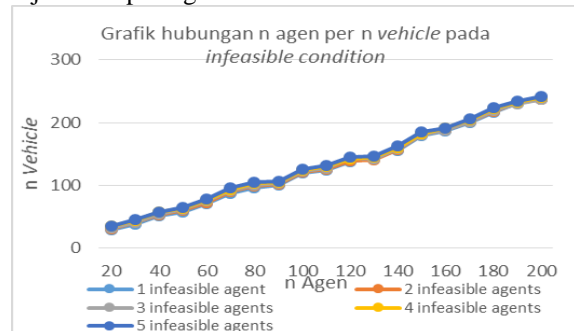
Gambar 13. Grafik hubungan n agen per *running time* kondisi *infeasible* dua produk

Grafik pada Gambar 13 yang dihasilkan menunjukkan kondisi kuadratis dari hubungan antara banyaknya agen dengan *running time* yang diperlukan untuk menyelesaikan perhitungan. Selanjutnya didapatkan juga data banyaknya kendaraan yang diperlukan untuk distribusi, dijelaskan di dalam Tabel 9.

Tabel 9. Hasil perhitungan n agen dan n vehicle kondisi *infeasible* dua produk

No	n Agen	Running time (ms) dengan agen kondisi <i>infeasible</i> sebanyak 1-5				
		1 Agen	2 Agen	3 Agen	4 Agen	5 Agen
1	20	30	31	33	35	36
2	30	39	41	42	44	46
3	40	51	53	55	57	58
4	50	58	60	62	64	65
5	60	71	73	75	77	79
6	70	88	90	92	94	96
7	80	96	99	101	103	105
8	90	100	102	104	105	107
9	100	120	122	123	125	126
10	110	125	126	128	131	132
11	120	138	140	144	142	145
12	130	145	141	143	145	147
13	140	155	157	158	160	163
14	150	179	181	181	182	185
15	160	187	188	189	191	192
16	170	200	201	202	204	206
17	180	216	218	220	222	224
18	190	231	232	230	233	234
19	200	237	238	238	241	242

Tabel 9 tersebut menjelaskan bahwa semakin banyak agen yang harus dilayani mengakibatkan semakin bertambahnya jumlah kendaraan yang diperlukan untuk distribusi. Hubungan mengenai banyaknya agen dengan banyaknya kendaraan dijelaskan pada gambar 14 berikut:



Gambar 14. Grafik hubungan n agen per n vehicle kondisi *infeasible* dua produk

Grafik pada Gambar 14 yang dihasilkan menunjukkan kondisi linear dari hubungan antara banyaknya agen dengan banyaknya kendaraan yang diperlukan untuk distribusi.

2.2.6 EFSISIENSI VRP INFEASIBLE CONDITION TIGA PRODUK

Perhitungan ini menggunakan tiga produk, *demand* dan matriks jarak antar agen kondisi *random*, dan terdapat 1-5 agen dengan kondisi *infeasible*. Dari hasil perhitungan didapatkan data seperti dalam Tabel 10.

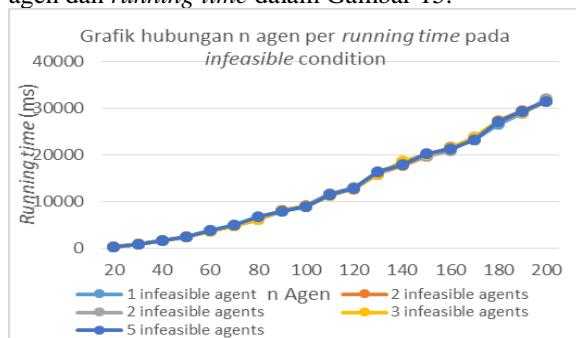
Tabel 10. Hasil perhitungan n agen dan *running time* kondisi *infeasible* tiga produk

No	n Agen	Running time (ms) dengan agen kondisi <i>infeasible</i> sebanyak 1-5				
		1 Agen	2 Agen	3 Agen	4 Agen	5 Agen
1	20	296	331	312	290	300
2	30	859	828	843	856	865
3	40	1657	1656	1625	1644	1634
4	50	2386	2399	2415	2455	2404
5	60	3710	3797	3722	3633	3764
6	70	4736	4819	4749	4760	4998
7	80	6096	6071	6183	6153	6728
8	90	7991	8094	8103	8048	7991
9	100	9099	9142	8989	8971	8914
10	110	11614	11623	11569	11286	11390
11	120	12889	12664	13012	12792	12900
12	130	16350	16044	15728	15721	16394
13	140	18113	17739	18378	18637	17873
14	150	19799	19762	19651	20002	20368
15	160	21727	21630	20980	21480	21269
16	170	23304	23253	23887	23909	23277

Tabel 10. Hasil perhitungan n agen dan $running$ time kondisi *infeasible* tiga produk (lanjutan)

No	n Agen	Running time (ms) dengan agen kondisi <i>infeasible</i> sebanyak 1-5				
		1 Agen	2 Agen	3 Agen	4 Agen	5 Agen
17	180	26545	27321	27198	27407	27152
18	190	28917	29563	28908	29040	29288
19	200	31598	31515	32043	31413	31492

Tabel 10 menunjukkan kondisi *infeasible* dengan banyaknya agen 20-200 dan agen dengan kondisi *infeasible* sebanyak 1-5 agen. Semakin banyak jumlah agen maka semakin bertambah $running$ time yang diperlukan. Untuk mengetahui hubungan antara banyaknya agen dan $running$ time selanjutnya dari data pada Tabel 10 didapatkan gambar grafik hubungan antara banyaknya jumlah agen dan $running$ time dalam Gambar 15.



Gambar 15. Grafik hubungan n agen per $running$ time kondisi *infeasible* tiga produk

Grafik pada Gambar 15 yang dihasilkan menunjukkan kondisi kuadratis dari hubungan antara banyaknya agen dengan $running$ time yang diperlukan untuk menyelesaikan perhitungan. Selanjutnya didapatkan juga data banyaknya kendaraan yang diperlukan untuk distribusi, dijelaskan di dalam Tabel 11.

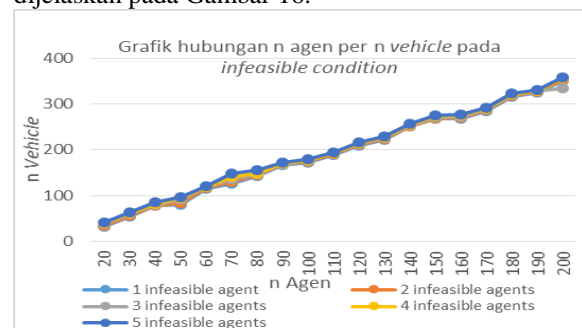
Tabel 11. Hasil perhitungan n agen dan n vehicle kondisi *infeasible* tiga produk

No	n Agen	Running time (ms) dengan agen kondisi <i>infeasible</i> sebanyak 1-5				
		1 Agen	2 Agen	3 Agen	4 Agen	5 Agen
1	20	33	35	37	39	42
2	30	55	57	60	61	63
3	40	78	79	81	82	85
4	50	80	84	92	94	96
5	60	116	117	118	119	121
6	70	127	129	135	141	148
7	80	143	145	147	148	155
8	90	167	168	169	171	173
9	100	173	174	175	177	179
10	110	189	190	192	193	195
11	120	209	211	213	215	216

Tabel 11. Hasil perhitungan n agen dan n vehicle kondisi *infeasible* tiga produk (lanjutan)

No	n Agen	Running time (ms) dengan agen kondisi <i>infeasible</i> sebanyak 1-5				
		1 Agen	2 Agen	3 Agen	4 Agen	5 Agen
12	130	222	224	226	228	229
13	140	251	252	254	255	257
14	150	267	268	269	273	275
15	160	268	270	271	275	277
16	170	284	286	288	290	292
17	180	316	317	320	322	323
18	190	325	326	328	329	331
19	200	351	353	335	357	358

Tabel 11 tersebut menjelaskan bahwa semakin banyak agen yang harus dilayani mengakibatkan semakin bertambahnya jumlah kendaraan yang diperlukan untuk distribusi. Hubungan mengenai banyaknya agen dengan banyaknya kendaraan dijelaskan pada Gambar 16.



Gambar 16. Grafik hubungan n agen per n vehicle kondisi *infeasible* tiga produk

Grafik pada Gambar 16 yang dihasilkan menunjukkan kondisi linear dari hubungan antara banyaknya agen dengan banyaknya kendaraan yang diperlukan untuk distribusi.

3. KESIMPULAN

Berdasarkan penelitian kasus *VRP Single dan Multi Product* menggunakan algoritma *Palgunadi* yang diaplikasikan dalam program *Java*, dapat diperoleh hasil rute kendaraan dan jumlah kendaraan yang digunakan. Hasil yang ditunjukkan program sesuai dengan perhitungan manual yang telah dilakukan sebelumnya. Algoritma *Palgunadi* juga dapat digunakan untuk menyelesaikan kasus dalam skala besar dibuktikan dengan $running$ time yang diperlukan untuk perhitungan. Hubungan banyaknya agen yang harus dilayani dengan $running$ time program menunjukkan kondisi kuadratik sedangkan hubungan banyaknya agen dengan banyaknya kendaraan yang dibutuhkan menunjukkan kondisi linear. Algoritma *Palgunadi* tidak memerlukan penentuan perhitungan *coordinate polar* untuk melakukan pencarian agen terdekat dari depot seperti yang dilakukan algoritma *Sweep* sehingga dapat mempercepat perhitungan. Untuk penelitian

mengenai VRP yang dapat dilakukan selanjutnya adalah bagaimana menyelesaikan *Vehicle Routing Problem Pick-Up Delivery with Single and Multi Product* menggunakan algoritma Palgunadi.

PUSTAKA

Angelelli, Enrico & Speranza, M. Grazia. 2002. *The Periodic Vehicle Routing Problem With Intermediate Facilities*. European Journal of Operation Research 137 page 233-247.

Archetti, C & Speranza, M.G. 2006. *A Tabu Search Algorithm For The Split Delivery Vehicle Routing Problem*. Transportation Science Vol. 40 No 1 pp. 66-73 ISSN 0041-1655.

Boonkleaw, Arunya, Nanthi, S. & Srinon, R. 2009. *Strategic Planning and Vehicle Routing Algorithm for Newspaper Delivery Problem: Case Study of Morning Newspaper, Bangkok, Thailand*. Proceedings of the World Congress on Engineering and Computer Science 2009 Vol II San Francisco, USA ISBN:978-988-18210-2-7.

Chang, Yaw & Chen, Lin. 2007. *Solve The Vehicle Routing Problem With Time Window Via A Genetic Algorithm*. Discrete And Continuous Dynamical Systems Supplement 2007 pp. 240-249 USA.

Frutos, Mariano & Tohme, Fernando. 2012. *A New Approach To The Optimization Of The CVRP Through Genetic Algorithms*. American Journal of Operations Research pp. 495-501.

Han, Sangheon dan Tabata, Yoshio. 2002. *A Hybrid Genetic Algorithm For The Vehicle Routing Problem With Controlling Lethal Gene*. Asia Pacific Management Review page 405-426.

Hubert, Joseph & Cavalier, Tom M. 2012. *A Genetic Algorithm For Split Delivery Vehicle Routing Problem*. American Journal of Operations Research Vol 2 No 2 page 207-216.

Montane, F.A.T & Galvao, Roberto D. 2006. *A Tabu Search Algorithm For The Vehicle Routing Problem With Simultaneous Pick-Up And Delivery Service*. Computer & Operation Research 33 (2006) 595-619.

Moolman, A.J. Koen, K, & Westhuizen, J.V.D. 2010. *Activity-Based Costing for Vehicle Routing Problem*. South African Journal of Industrial Engineering Nov 2010 Vol 21(2) p: 161-171.

Nguyen, P.K., Crainic, T.G., Toulouse, M. 2011. *A Hybrid Genetic Algorithm For The Periodic Vehicle Routing Problem With Time Windows*. Interuniversity Research Centre on Enterprise Network, Logistics, and Transportations.

Ong, Johan Oscar. 2011. *Algoritma Sequential Untuk Mengatasi Masalah Rute Kendaraan Dengan Backhaul, Rute Majemuk Dan Time Window*. Jurnal Tekno Insentif Kopwil4 Vol 5 No.2 ISSN: 1907-4964 Hal: 16-27.

Putri, Elkagianda L.A. & Sarngadi, Palgunadi. 2014. *Implementasi Algoritma Palgunadi Sebagai Algoritma Baru Dalam Optimalisasi Vehicle*

Routing Problem With Time Window (VRPTW). Prosiding Seminar Nasional TI 2014 ISSN 1829-9156 Vol II No 1 Fakultas Teknologi Informasi Universitas Tarumanegara Jakarta.

Rahayu, Ragil. (2012). *Penentuan Rute Kendaraan Logistik Menggunakan Metode Heuristik (Studi Kasus Gudang Bulog Kalasan Utama Divre Yogyakarta)*. Skripsi Fakultas Sains dan Teknologi Program Studi Teknik Industri UIN Yogyakarta.

Shah, Megha Mihir. 2012. *Artificial Intelligence: Vehicle Routing Problem and Multi Agent System*. National Conference on Developing of Reliable Information System, Techniques and Related Issues (DRISTI) Proceeding published in International Journal of Computer Applications (IJCA).

Shanmugam, G., Ganesan, P., Vanathi, P.T. 2011. *Meta Heuristic Algorithms For Vehicle Routing Problem With Stochastic Demands*. Journal of Computer Science 7 (4): pp 533-542 ISSN 1549-3636.

Slamet, Alim Setiawan, Siregar, Hariman Hidayat, & Kustiyo, Azis. 2014. *Vehicle Routing Problem (VRP) By Genetic Algorithm On The Distribution Of Highland Vegetables*. Jurnal Teknologi Industri Pertanian 24(1):1-10 Bogor.

Suthikarnnarunai, N. 2008. *A Sweep Algorithm For The Mix Fleet Vehicle Routing Problem*. Proceedings of the International MultiConference of Engineers and Computer Scientists Vol 2 ISBN:978-988-17012-1-3 Hong Kong.

Venkatesan, S.R., Logendran, D., & Chandramohan, D. 2011. *Optimization Of Capacitated Vehicle Routing Problem Using PSO*. International Journal of Engineering Science and Technology ISSN:0975-5462 Vol 3 No 10 page: 7469-7477.

Yuceer: *An Employee Transporting Problem*. Journal of Industrial Engineering International 2013.