PENYELESAIAN *SPLIT DELIVERY VEHICLE ROUTING PROBLEM* MENGGUNAKAN *INTEGER LINEAR PROGRAMMING* DAN ALGORITME *TABU SEARCH*

R. Puspaningrum, P. T. Supriyo, *H. Mayyani, dan A. Aman

Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam,
Institut Pertanian Bogor, Jl. Meranti, Kampus IPB Dramaga Bogor.
rahmapn81@gmail.com, praptosu@apps.ipb.ac.id,
mayyani_mat15@apps.ipb.ac.id ***corresponding author**
aaman@apps.ipb.ac.id

Abstrak

Suatu masalah penentuan rute pendistribusian barang ke para pelanggan yang dimulai dan diakhiri di suatu depot disebut sebagai *Split Delivery Vehicle Routing Problems* (SDVRP). Setiap pelanggan dapat dikunjungi lebih dari satu kali. Model SDVRP merupakan kasus *NP-Hard Problem* yang dapat diselesaikan menggunakan metode eksak, *heuristic* maupun *metaheuristic*. Pada karya ilmiah ini, SDVRP diselesaikan menggunakan metode eksak *Integer Linear Programming* (ILP) dan metode *metaheuristic* algoritme *Tabu Search 2-opt.* Hasil uji coba memperlihatkan bahwa waktu eksekusi menggunakan algoritme *Tabu Search* 2-*opt* 61,240 kali lebih cepat dibandingkan dengan metode ILP. Akan tetapi, algoritme *Tabu Search* 2-*opt* hanya menghasilkan solusi pendekatan dengan selisih jarak sebesar 15.55% dari hasil optimal yang diperoleh dengan metode ILP.

Kata kunci: 2-*opt*, penentuan rute, SDVRP, *Tabu Search*

1 Pendahuluan

Dalam dunia industri, setiap perusahaan dituntut untuk bertahan dalam persaingan usaha yang ketat. Hal yang harus diperhatikan oleh setiap perusahaan, selain proses produksi yaitu proses distribusi. Pada proses distribusi, sebuah kendaraan menyalurkan barang yang telah diproduksinya ke berbagai distributor sebelum dapat digunakan oleh konsumen. Permasalahan penentuan proses distribusi ini sering disebut sebagai *Vehicle Routing Problem* (VRP).

VRP merupakan masalah pencarian rute optimal dalam pendistribusian barang dengan mempertimbangkan jumlah kendaraan dan rute yang akan dilalui serta memenuhi kendala tertentu. Setiap pelanggan hanya dikunjungi satu kali dan semua kendaraan dimulai dan diakhiri di depot [10]. Masalah VRP ini merupakan generalisasi dari TSP dengan menyertakan kendala kapasitas kendaraan yang digolongkan ke dalam *NP-Hard Problem* [4]. Terdapatbeberapa VRP yang telah dikembangkan, salah satunya SDVRP dan VRPSDP[8].

Karya ilmiah ini membahas masalah pendistribusian barang yang dimodelkan sebagai *Split Delivery Vehicle Routing Problem* (SDVRP). Proses pendistribusian ini diawali dari depot dan kembali lagi ke depot. Setiap pelanggan dapat dikunjungi lebih

2020 Mathematics Subject Classification: 90C08, 90C10, 90C59.
Diajukan:16/12/2024, diterima: 31/12/2024. DOI: https://doi.org/10.29244/milang.20.2.123-133
MILANG Journal of Mathematics and Its Applications, Vol.20, No.2, pp.123-133 ISSN: 2963-5233

dari satu kendaraan, dikarenakan permintaan setiap pelanggan bisa lebih besar dari kapasitas kendaraan. Model *Split Delivery VehicleRouting Problem* (SDVRP) merupakan kasus *NP-Hard Problem* [1]. Kasus *NP-Hard* dapat diselesaikan menggunakan metode eksak, *heuristic* maupun *metaheuristic* [4].

Pada karya ilmiah ini akan dibandingkan waktu eksekusi dan total jarak yang diperoleh antara metode eksak dan metode *metaheuristic.* Metode *metaheuristic* yang digunakan adalah algoritme *Tabu Search*. Algoritme ini digunakan untuk mencari solusi terbaik SDVRP, yaitu rute yang memiliki total jarak tempuh minimum dengan mempertimbangkan kapasitas kendaraan. Tujuan dari penelitian ini adalah membandingkan waktu eksekusi dan total jarak yang diperoleh antara metode eksak *Integer Linear Programming* (ILP) dan metode pendekatan (*metaheuristic) Tabu Search 2-opt* dalam menyelesaikan SDVRP.

2 Formulasi Masalah

2.1 Model Matematik SDVRP

Model matematik untuk permasalahan SDVRP adalah sebagai berikut [2]:

Notasi:
S : himpunan lokasi depot dan distributor $= \{1, 2, \dots, s\}$
V : himpunan kendaraan yang tersedia $= \{1, 2, \dots, v\}$

Indeks:
i, j, p : indeks untuk menyatakan depot dan distributor, dengan $i = 1$ merupakan depot
k : indeks untuk menyatakan kendaraan

Parameter:
Q_k: kapasitas kendaraan k
$demand_i$: banyaknya permintaan distributor i
$jarak_{i,j}$: jarak dari distributor i ke distributor j (dalam km)
N: banyaknya distributor

Variabel Keputusan:
$x_{ijk} = \begin{cases} 1, & \text{jika kendaraan } k \text{ mengunjungi distributor } j \text{ setelah distributor } i \\ 0, & \text{lainnya} \end{cases}$
y_{ik} = banyaknya permintaan i yang dikirim oleh kendaraan k
u_{ik} = variabel tambahan yang digunakan dalam eliminasi *subtour* untuk distributor i oleh kendaraan k

Fungsi Tujuan:
Fungsi objektif dalam masalah pendistribusian ini adalah meminimumkan jarak distribusi, yaitu

$$\min \sum_{i \in S} \sum_{j \in S} \sum_{k \in V} jarak_{i,j} \times x_{ijk} .$$

Kendala:
1. Setiap distributor dikunjungi minimal sebanyak satu kali,

$$\sum_{i \in S} \sum_{k \in V} x_{ijk} \geq 1 \, , \forall j \in S.$$

2. Kendaraan yang mengunjungi distributor harus meninggalkan distributor tersebut (kontinu),
$$\sum_{i \in S} x_{ipk} - \sum_{i \in S} x_{pjk} = 0, \forall p \in S, \forall k \in V.$$
3. Kendala eliminasi *subtour* yang bertujuan membentuk adanya *tour* yang fisibel,
$$u_{ik} - u_{jk} + N x_{ijk} \leq N - 1, \forall i, j \in S, \forall k \in V.$$
4. Distributor dikunjungi oleh kendaraan yang digunakan,
$$demand_i \sum_{j \in S} x_{ijk} \geq y_{ik}, \forall i \in S, \forall k \in V.$$
5. Setiap distributor mendapatkan permintaannya
$$\sum_{i \in S} y_{ik} = demand_i, \forall k \in V.$$
6. Banyaknya barang yang dibawa tidak melebihi kapasitas maksimum kendaraan,
$$\sum_{i \in S} y_{ik} \leq Q_k, \forall k \in V.$$
7. Tidak ada perjalanan ke distributor yang sama,
$$\sum_{j \in S} x_{iik} = 0, \forall k \in V.$$
8. Kendala biner dan kendala ketaknegatifan,
$$x_{ijk} \in \{0,1\}, \forall i, j \in S, \forall k \in V,$$
$$y_{ik} \geq 0, \forall i \in S, \forall k \in V.$$

2.2 Algoritme *Tabu Search*

Algoritme *Tabu Search* pertama kali ditemukan pada tahun 1986 oleh FredGlover. Algoritme ini merupakan salah satu metode *meta-heuristic* yang dapat membantu dalam mencari solusi mendekati optimal VRP yaitu rute yang memiliki total jarak tempuh minimum dengan mempertimbangkan kapasitas kendaraan [6]. Algoritme *Tabu Search* mempunyai dua struktur memori berupa *short-term memory* dan *long term memory* yang mempunyai fungsi untuk mengevaluasi solusi yang pernah ditemukan. *Short-term memory* terdiri dari *tabu list*, *tabu tenure* dan kriteria aspirasi, sedangkan *long-term memory* digunakan untuk fase intensifikasi dan diversifikasi [5].

Pencarian solusi awal

Penentuan solusi awal menggunakan metode *Nearest Neighbor,* yaitu metode pencarian pelanggan terdekat. Pertama-tama semua rute kendaraan masih kosong. Dimulai dari rute kendaraan pertama, dengan memasukkan satu per satu pelanggan terdekat ke dalam rute kendaraan dan tidak melebihi kapasitas maksimum kendaraan tersebut. Kemudian proses pencarian pelanggan terdekat dilakukan untuk kendaraan berikutnya sampai semua pelanggan telah dikunjungi [3].

Pencarian solusi *neighbourhood*

Solusi *neighbourhood* dalam algoritme *Tabu Search* didefinisikan sebagai solusi yang diperoleh dengan melakukan satu "*move*". *Move* merupakan proses penghilangan sisi yang ada dalam solusi kemudian menambahkan sisi yang baru kedalam solusi. Setiap *move* menghasilkan satu solusi *neighbourhood*. *Move* yang digunakan dalam karya ilmiah ini adalah *2-opt*. Metode *2-opt* termasuk *intra-route movement*, yaitu menghilangkan dua jalur pada rute yang sama kemudian menghubungkan kembali jalur tersebut dengan pasangan node yang berbeda [7].

Setiap *move* membuat solusi *neighbourhood*-nya masing-masing. Pencarian solusi *neighbourhood* dilakukan secara iteratif hingga kriteria penghentian terpenuhi.Pencarian solusi *neighbourhood* yang digunakan pada karya ilmiah ini menggunakanpemilihan *move* berdasarkan *best admissible solution*. *Best admissible solution* menggunakan *move value* yaitu, selisih jarak total sisi yang ditambahkan dengan jarak total yang dihilangkan dan semua *move* yang dilakukan disimpan dalam *tabucontour* [5].

Short-term memory

Short-term memory merupakan struktur memori jangka pendek yang digunakan pada algoritme *Tabu Search. Short-term memory* terdiri dari *tabu list*, *tabu tenure* dan kriteria aspirasi. Fungsi dari *short-term memory* itu sendiri, yaitu menyimpan semua *move* yang telah dilakukan dalam pencarian solusi untuk mencegah terjadinya *cycle* terhadap solusi yang pernah ditemukan pada iterasi-iterasi sebelumnya.

Komponen pertama yang ada dalam *short-term memory* ialah *tabu list*. *Tabu list* adalah *list* yang digunakan untuk menyimpan sisi yang bersifat tabu. Sebuah sisi bersifat tabu ketika sisi tersebut telah digunakan dalam sebuah *move* pada iterasi sebelumnya. Pada karya ilmiah ini, diperlukan *list* lain untuk menyimpan semua *move* yang telah dilakukan dan dimisalkan sebagai *tabu contour*. Sisi yang disimpan dalam *tabu list* bersifat tabu-aktif untuk beberapa iterasi. Penetapan sisi sebagai tabu-aktif berguna untuk menghindari ditemukan kembali solusi yang sama pada iterasi selanjutnya. Selain sisi memiliki status tabu-aktif, sisi tersebut mempunyai nilai *tabu tenure.*

Tabu tenure merupakan suatu nilai untuk menentukan lamanya suatu sisi yang berada dalam *tabu list* bersifat tabu-aktif. Tidak terdapat ukuran yang pasti untuk menentukan banyaknya *tabu list* dan nilai *tabu tenure* pada suatu kasus. *Tabu tenure* yang tepat bergantung pada kasus itu sendiri, apabila terdapat rentang yang kuat akan memberikan kinerja yang baik [6]. Status *tabu* pada sebuah *move* dapat dilanggar apabila memenuhi kriteria aspirasi, yaitu *move* yang bersifat tabu-aktif menghasilkan solusi objektif yang lebih baik dari solusi terbaik yang pernah ditemukan sebelumnya. Misalkan solusi sekarang s^* dan solusi *neighbourhood* yang dihasilkan dari *move* yang berstatus *tabu* ialah s'. Untuk masalah minimisasi, status *tabu* dari *move* dapat dilanggar (memenuhi kriteria aspirasi) apabila $f(s') < f(s^*)$ dengan $f(s)$ adalah nilai fungsi objektif dari solusi s, maka solusi yang diperoleh dengan melanggar status *tabu* lebih baik dari solusi sekarang [9].

Long-term memory

Dalam beberapa kasus, *short-term memory* cukup menghasilkan solusi yang berkualitas sangat tinggi. Secara umum, *Tabu Search* menjadi lebih kuat secara signifikan dengan memasukkan *long-term memory*. Dalam *long-term memory*, pencarian dilakukan terhadap solusi yang sudah pernah ditemui. *Long-term memory*pada karya ilmiah ini

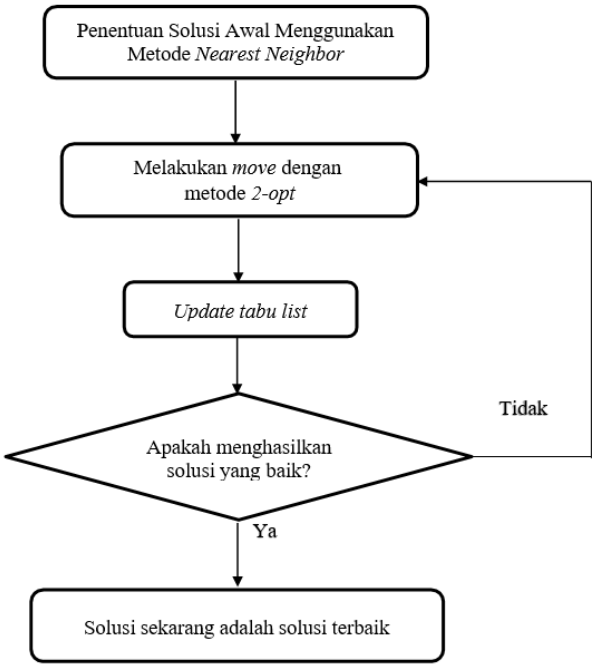
adalah struktur memori yang berbasis frekuensi. *Long-term memory* digunakan pada fase intensifikasi dan fase diversifikasi.

Fase intensifikasi bertujuan melakukan pencarian *neighbourhood* untuk menyelidiki solusi terbaik yang telah ditemukan secara lebih mendetail serta untuk meyakinkan bahwa solusi terbaik yang telah ditemukan pada fase sebelumnya telah ditemukan. Fase diversifikasi bermaksud untuk mengarahkan algoritme *Tabu Search* dalam melakukan pencarian solusi ke daerah solusi yang belum dikunjungi pada iterasi-iterasi sebelumnya [5].

Kriteria penghentian

Penghentian pencarian pada karya ilmiah ini, yaitu saat sudah tidak ada lagi *move* yang lebih baik (memenuhi kriteria aspirasi) sehingga proses algoritme *TabuSearch* berhenti. Jika tidak, maka kembali pada evaluasi *move* untuk mencari solusiterbaik di iterasi-iterasi berikutnya [7].

Setelah diketahui tahapan-tahapan algoritme *Tabu Search*, berikut ditampilkan diagram alir algoritme *Tabu Search*:



Gambar 1. Diagram alir algoritme *Tabu Search*

3 Hasil dan Pembahasan

3.1 Aplikasi Masalah

Pada bagian ini diberikan suatu kasus pendistribusian barang dengan model *Split Delivery Vehicle Routing Problem* (SDVRP). Data yang digunakan merupakan data hipotetik. Setiap distributor memiliki permintaannya masing-masing. Misalkan perusahaan memiliki 3 kendaraan untuk melakukan pendistribusian barang. Kendaraan tersebut dapat mengangkut sebanyak 100 barang dalam satu kali angkut sehingga total barang yang diangkut tidak melebihi kapasitas kendaraan. Setiap distributor boleh dikunjungi lebih dari satu kendaraan dikarenakan banyaknya permintaan yang melebihi

kapasitas kendaraan. Data jarak antar distributor berupa koordinat x dan y yang diperoleh secara hipotetik dengan fungsi random, disajikan pada Tabel 1 berikut:

Tabel 1. Data koordinat x dan y pada suatu kasus pendistribusian barang.

x	y
0	0
9	4
7	7
5	10
8	8
10	5
6	9
9	11

Dari data di atas, dicari jarak antarkota dengan menggunakan rumus *Euclidean* sehingga menghasilkan matriks jarak. Matriks jarak yang diperoleh dapatdilihat pada Tabel 2. Kedua model dieksekusi menggunakan bantuan laptop dengan spesifikasi Lenovo Think Pad X230 i5 RAM 4GB dan hasil dari kedua model akan dibandingkan dalam aspek waktu eksekusi dan total jarak yang diperoleh.

Tabel 2. Data jarak dan permintaan setiap *node.*

Distributor/ *node*	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	0	9,8480	9,8990	11,1800	11,3130	11,1800	10,8160	14,2120
2	9,8480	0	3,6055	7,2111	4,1231	1,4142	5,8309	7,0000
3	9,8990	3,6055	0	3,6055	1,4142	3,6055	2,2360	4,4721
4	11,1800	7,2111	3,6055	0	3,6055	7,0710	1,4142	4,1231
5	11,3130	4,1231	1,4142	3,6055	0	3,6055	2,2360	3,1622
6	11,1800	1,4142	3,6055	7,0710	3,6055	0	5,6568	6,0827
7	10,8160	5,8309	2,2360	1,4142	2,2360	5,6568	0	3,6055
8	14,2120	7,0000	4,4721	4,1231	3,1622	6,0827	3,6055	0
Permintaan	0	20	25	28	15	15	25	20

3.2 Penyelesaian SDVRP menggunakan *Integer Linear Programming* (ILP)

Penyelesaian SDVRP menggunakan *Integer Linear Programing* (ILP) dilakukan dengan bantuan *software* Lingo 17.0. Terdapat suatu kasus pendistribusian barang yang memiliki distributor sebanyak 7. Setiap distributor memiliki permintaannya masing-masing. Perusahaan memiliki 3 kendaraan untuk melakukan pendistribusian barang. Kendaraan tersebut dapat mengangkut sebanyak 100 barang dalam satu kali angkut sehingga total barang yang diangkut tidak melebihi kapasitas kendaraan.

Pada kasus SDVRP ini diperoleh solusi optimum global sebesar 55.4416 km. Solusi yang diperoleh pada kasus SDVRP ini merupakan solusi terbaik yang memenuhi semua kendala. Hasil rute yang diperoleh dari kasus tersebut dapat dilihat pada Tabel 3. Pada kasus SDVRP ini jumlah kendaraan yang diperlukan untuk melakukan pengiriman ke seluruh distributor sebanyak 2 buah. Dalam penyelesaian menggunakan ILP menggunakan kendaraan 1 dan 2, Tidak ada kriteria khusus dalam pemilihan kendaraan, karena kapasitas kendaraan adalah seragam yaitu sebesar 100 barang.

Tabel 3. Hasil rute yang diperoleh dari ILP.

Kendaraan	Rute	Jarak
1	1-5-8-7-4-1	
2	1-3-6-2-1	55.4416
3	-	

3.3 Penyelesaian SDVRP menggunakan Algoritme *Tabu Search*

Algoritme *Tabu Search* diawali dengan penentuan solusi awal menggunakan metode *Nearest Neighbor.* Penentuan solusi awal dilakukan dengancara memasukkan satu persatu distributor terdekat ke dalam satu rute kendaraan. Solusi *Nearest Neighbour* diperoleh yaitu urutan rute 1-2-6-3-5-7-4-8-1.

Setelah memperoleh solusi awal, selanjutnya menentukan *tabu list* dan *tabucontour* yang masih berupa himpunan kosong. Berikutnya, ditentukan *tabu tenure* dengan nilai maksimal 3. Nilai *tabu tenure* digunakan untuk menentukan batas tabu-aktif pada setiap *move*. Proses *move* yang digunakan dalam karya ilmiah ini yaitu metode *2-opt.* Metode ini, menghilangkan dua jalur pada rute yang sama kemudian menghubungkan kembali jalur tersebut dengan pasangan *node* yang berbeda. *Move* ini mempunyai *value* yang digunakan dalam pencarian solusi terbaik. Semua pencarian yang telah dilakukan disimpan dalam *tabu contour*, kemudian dicari hasil terbaik dari setiap iterasi untuk disimpan dalam *tabu list*.

Penyelesaian Kasus

Pada kasus SDVRP ini, solusi awal yang diperoleh adalah 1-2-6-3-5-7-4-8-1. Semua pendistribusian dimulai dan diakhiri di depot. Oleh karena itu, rute pendistribusianuntuk melakukan pertukaran *move* menjadi 2-6-3-5-7-4-8 yang memiliki indeks dari 1 sampai dengan 7. Pertukaran indeks *move* yang menghasilkan *move value* (jarak) dapat dilihat pada Tabel 4, Tabel 5, dan Tabel 6.

Tabel 4. *Tabu contour* iterasi 1.

Move		***Best_nbr2***								***Move value***
1	2	6	2	3	5	7	4	8	1	39.5992
1	3	3	6	2	5	7	4	8	1	41.0271
1	4	5	6	3	2	7	4	8	1	47.7097
1	5	7	6	3	5	2	4	8	1	51.1618
1	6	4	6	3	5	7	2	8	1	52.5496
1	7	8	6	3	5	7	4	2	1	46.0237
2	3	2	3	6	5	7	4	8	1	42.6498
2	4	2	5	3	6	7	4	8	1	44.3969
2	5	2	7	3	5	6	4	8	1	48.3407
2	6	2	4	3	5	7	6	8	1	50.2663
2	7	2	8	3	5	7	4	6	1	44.6355
3	4	2	6	5	3	7	4	8	1	38.2672
3	5	2	6	7	5	3	4	8	1	42.5098
3	6	2	6	4	5	7	3	8	1	45.0948

Move		***Best_nbr2***								***Move value***
3	7	2	6	8	5	7	4	3	1	37.6618
4	5	2	6	3	7	5	4	8	1	41.2803
4	6	2	6	3	4	7	5	8	1	39.4976
4	7	2	6	3	8	7	4	5	1	39.2780
5	6	2	6	3	5	4	7	8	1	39.1191
5	7	2	6	3	5	8	4	7	1	**35.7974**
6	7	2	6	3	5	7	8	4	1	37.4265

Tabel 5. *Tabu contour* iterasi 2.

Move		***Best_nbr2***								***Move value***
1	2	6	2	3	5	8	4	7	1	37.1294
1	3	3	6	2	5	8	4	7	1	38.5573
1	4	5	6	3	2	8	4	7	1	45.4828
1	5	8	6	3	5	2	4	7	1	48.8788
1	6	4	6	3	5	8	2	7	1	50.0798
1	7	7	6	3	5	8	4	2	1	45.8369
2	3	2	3	6	5	8	4	7	1	40.1800
2	4	2	5	3	6	8	4	7	1	41.4268
2	5	2	8	3	5	6	4	7	1	45.6410
2	6	2	4	3	5	8	6	7	1	47.7965
2	7	2	7	3	5	8	4	6	1	44.8654
3	4	2	6	5	3	8	4	7	1	37.1073
3	5	2	6	8	5	3	4	7	1	37.7570
3	6	2	6	4	5	8	3	7	1	42.6250
3	7	2	6	7	5	8	4	3	1	39.9448
4	5	2	6	3	8	5	4	7	1	38.3377
4	6	2	6	3	4	8	5	7	1	38.8105
4	7	2	6	3	7	8	4	5	1	39.7508
5	6	2	6	3	5	4	8	7	1	38.4320
5	7	2	6	3	5	7	4	8	1	38.2672
6	7	2	6	3	5	8	7	4	1	**35.6438**

Tabel 6. *Tabu contour* iterasi 3.

Move		***Best_nbr2***								***Move value***
1	2	6	2	3	5	8	7	4	1	36.9758
1	3	3	6	2	5	8	7	4	1	38.4037
1	4	5	6	3	2	8	7	4	1	45.3292
1	5	8	6	3	5	2	7	4	1	47.8626
1	6	7	6	3	5	8	2	4	1	50.0458
1	7	4	6	3	5	8	7	2	1	45.7173
2	3	2	3	6	5	8	7	4	1	40.0264
2	4	2	5	3	6	8	7	4	1	41.2732
2	5	2	8	3	5	6	7	4	1	44.5908
2	6	2	7	3	5	8	6	4	1	46.8250
2	7	2	4	3	5	8	7	6	1	45.6833
3	4	2	6	5	3	8	7	4	1	36.9537
3	5	2	6	8	5	3	7	4	1	36.7515
3	6	2	6	7	5	8	3	4	1	41.5748
3	7	2	6	4	5	8	7	3	1	40.8414
4	5	2	6	3	8	5	7	4	1	37.3322
4	6	2	6	3	7	8	5	4	1	38.6569
4	7	2	6	3	4	8	7	5	1	39.7508
5	6	2	6	3	5	7	8	4	1	37.4265
5	7	2	6	3	5	4	7	8	1	39.1191
6	7	2	6	3	5	8	4	7	1	**35.7974**

Berdasarkan Tabel 4, Tabel 5, dan Tabel 6 banyaknya kombinasi pertukaran *move* yang dilakukan dalam setiap iterasi sebanyak 21 kombinasi. Setiap pertukaran *move* memiliki *move value* (nilai objektif) yang disimpan dalam *tabu contour*. Setelah itu, proses intensifikasi dilakukan dengan mencari nilai objektif terkecil dari setiap iterasi dan proses diversifikasi digunakan untuk melakukan perluasan daerah pencarian ke iterasi selanjutnya. Nilai objektif terkecil dari setiap iterasi tersebut disimpan dalam *tabu list* yang dapat dilihat pada Tabel 7.

Tabel 7. *Tabu list.*

Iterasi	***Move***		***Move value***
1	5	7	35.7974
2	6	7	**35.6438**
3	6	7	35.7974

Tabu list awalnya berupa himpunan kosong. Pada iterasi pertama, didapatkan *move* antara indeks 5 dan 7. Pertukaran indeks 5 dan 7 mempunyai nilai *tabu tenure* sebesar 3, yang artinya selama 3 iterasi selanjutnya *move* tersebut tidak boleh dilakukan kembali. Pada iterasi kedua, nilai objektif terkecil ada pada *move* antara indeks 6 dan 7. *Move* tersebut memiliki *tabu tenure* sebesar 3, sedangkan *move* 5 dan 7 saat ini memiliki *tabu tenure* sebesar 2, begitupun seterusnya nilai untuk menentukan *tabu tenure* pada *move* yang ada dalam tabu list.

Kesimpulan dari kasus SDVRP ini, solusi terbaik ada pada iterasi kedua dengan pertukaran *move* antara indeks 6 dan 7 yang menyebabkan proses penghentian terpenuhi

berdasarkan kriteria aspirasi, sehingga pencarian hanya dilakukan sampai iterasi ketiga, karena pada iterasi kedua telah ditemukan solusi yang lebih baik dibandingkan dengan solusi yang ada pada iterasi ketiga.

3.4 Perbandingan Hasil ILP dan Algoritme *Tabu Search*

Pada bagian ini, akan dibandingkan hasil eksekusi dari masing-masing metode berdasarkan waktu eksekusi dan total jarak yang diperoleh. Hasil perbandingannya dapat dilihat pada Tabel 8 berikut.

Tabel 8. Perbandingan waktu eksekusi dan total jarak setiap metode.

Perbandingan	Metode	
	ILP	***Tabu Search***
Waktu eksekusi	02 33’ 06’’	00 00’ 0.15’’
Total jarak	55.4416	64.0678

Berdasarkan Tabel 8 dapat diketahui bahwa waktu eksekusi pada algoritme *Tabu Search* jauh lebih cepat dibandingkan dengan waktu eksekusi menggunakan ILP. Pada Kasus SDVRP ini membutuhkan waktu eksekusi dengan algoritme *Tabu Search* hanya dalam satuan detik. Waktu eksekusi *Tabu Search* 61,240 kali lebih cepat dibandingkan dengan waktu eksekusi yang optimal dari ILP. Sedangkan untuk total jarak yang dihasilkan, metode ILP menghasilkan solusi yang optimal, namun pada *Tabu Search* menghasilkan solusi yang mendekati optimal. Pada kasus SDVRP ini, selisih jarak sebesar 15.55% dari hasil optimal. Presentase tersebut cukup besar karena terdapat jarak yang jauh saat melakukan *split* antara kendaraan 1 dan kendaraan 2.

4 Simpulan dan Saran

Dalam penyelesaian SDVRP, pencarian rute optimal menggunakan metode eksak memerlukan waktu yang lama. Banyaknya node sangat mempengaruhi dalam menyelesaikan kasus ini. Dengan adanya metode *metaheuristic*, didapatkan hasil yang mendekati optimal dengan waktu eksekusi jauh lebih cepat dibandingkan dengan metode eksak. Hasil implementasi pada suatu kasus pendistribusian barang diperoleh perbandingan jarak antara metode eksak dan *metaheuristic* sebesar17.68%.

Saran untuk penelitianselanjutnya, dapat menggunakan tiga tipe *move* (*relocated, relocated split, dan exchange*) sehingga hasil yang didapatkan mungkin akan lebih mendekati optimal.

Daftar Pustaka

[1] Archetti C, Mansini R, Speranza MG. 2005. Complexity and reducibility of the skip delivery problem. *Transportation Science*. 39(2):182-187. https://doi.org/10.1287/trsc.1030.0084.

[2] Archetti C, Speranza MG. 2012. Vehicle routing problems with split deliveries. *International Transactions in Operational Research*. 19(1–2):3-22. https://doi.org/10.1111/j.1475-3995.2011.00811.x.

[3] Bräysy O, Gendreau M. 2005. Vehicle Routing Problem with Time Windows, Part II: Metaheuristics. *Transportation Science*.39(1):119-139. https://doi.org/10.1287/trsc.1030.0057.

[4] Çetiner S. 2003. *An Iterative Hub Location and Routing Problem for Postal Delivery Systems* [thesis]. Turkey: The Middle East Technical University.

[5] Garside AK, Cahyanti DN.2018. Penyelesaian vehicle routing problem with simultaneous pick up and delivery dengan algoritma Tabu Search. *Jurnal Ilmiah Teknik Industri*. 17(2):125. https://doi.org/10.23917/jiti.v17i2.6703.
[6] Glover F, Laguna M. 1997. *Tabu Search*. Boston (US): Kluwer Academic.
[7] Ho SC, Haugland D. 2004. A tabu search heuristic for the vehicle routing problem with time windows and split deliveries. *Comput Oper Res.* 31(12):1947-1964. https://doi.org/10.1016/S0305-0548(03)00155-2.
[8] Salsabila SK, Mayyani H, Supriyo PT. 2023. Penyelesaian VRPSDP menggunakan Firefly Algorithm (studi kasus distribusi Aqua galon). *MILANG Journal of Mathematics and Its Applications.* 19(1):53-67. https://doi.org/10.29244/milang.19.1.53-67.
[9] Schneider U. 2011. A tabu search tutorial based on a real-world scheduling problem. *Cent Eur J Oper Res.* 19(4):467-493. https://doi.org/10.1007/s10100-010-0137-8.
[10] Toth P, Vigo D. 2002. *The Vehicle Routing Problem*. Philadhelphia: Society for Industrial and Applied Mathematics.