Terbit *online* pada laman : https://teknosi.fti.unand.ac.id/

Jurnal Nasional Teknologi dan Sistem Informasi

| ISSN (Print) 2460-3465 |ISSN (Online) 2476-8812|

Artikel Penelitian

Sistem Pendukung Keputusan dengan Algoritma Branch&Bound dan Naive Approach pada Beberapa Pemesanan Makanan Online

Bhustomy Hakim[1*], *Fendyanto*[2]

[1,2] *Universitas Bunda Mulia, Jl. Sutera Barat Kav 7-9 Alam Sutera, Tangerang, 15143, Indonesia*

INFORMASI ARTIKEL

Sejarah Artikel:
Diterima Redaksi: 12 Juli 2022
Revisi Akhir: 07 Agustus 2022
Diterbitkan *Online*: 31 Agustus 2022

KATA KUNCI

Traveling Salesman Problem,
Branch & Bound,
Naïve,
Rute,
Simple Additive Weighting

KORESPONDENSI

E-mail: bhustomy.hakim@gmail.com*

A B S T R A C T

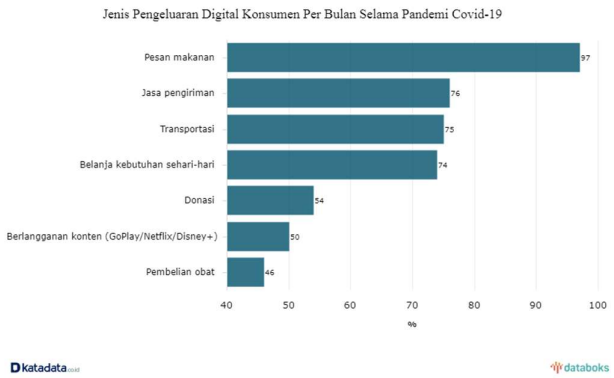
Dengan berkembangnya teknologi saat ini, cara manusia menjalani hidupnya juga mengalami pergeseran seperti dalam memenuhi kebutuhannya untuk makan. Pemesanan makanan online menjadi suatu aktivitas yang menjamur. Gofood dan Grabfood merupakan salah satu platform terbesar yang mendominasi pasar pemesanan *online* di Indonesia. Mereka mendapatkan kurang lebih 160 miliar rupiah dengan 3 juta *order*-an perbulan untuk hanya melayani pesanan makanan *online* saja. Namun dalam menangani pemesanan tersebut, mereka masih menggunakan *system one-o-one* dimana driver hanya melakukan pemesanan dan pengantara ekslusif ke satu pelanggan. Dengan waktu tunggu yang cukup panjang yaitu 30-60 menit, hal ini dirasa tidak efektif dari sisi kinerja. Oleh karena itu, *multiple* pesanan diajukkan di penelitian ini. *Traveling Salesman Problem* (TSP) digunakan untuk menggambarkan permasalahan *multiple* pesanan untuk menentukan rute terbaik yang memiliki nilai keefektifan terbaik. Dalam penelitian ini, dua algoritma untuk memecahkan masalah TSP telah dibandingkan satu sama lain. Dua algoritma tersebut yaitu *Brand&Bound Strategy* dan *Naïve Approach*. Jalur rute yang memiliki titik awal dan akhir yang sama dan memiliki berat minimal. Berat dari masing-masing rute akan didapatkan dari kriteria; jarak, tingkat kesibukkan, dan prioritas, lalu didapatkan berat ideal dengan menggunakan *Simple Additive Weighting* (SAW). Kriteria evaluasi dalam laporan ini adalah jalur/rute terbaik, kompleksitas waktu dan biaya masing-masing algoritma. Dan ada empat kondisi yang berbeda dalam ukuran yang telah digunakan sebagai masalah untuk evaluasi dua algoritma dalam penelitian ini. Dari penelitian ini, ditemukan hasil bahwa algoritma *Branch&Bound* menunjukan performa lebih efisien daripada *Naive Aproach* dimana hanya membutuhkan kurang dari satu detik untuk menemukan rute jalan terbaik dengan bobot tertinggi sebagai bahan pengambilan keputusan dengan biaya algoritma $O(n^3)$.

1. PENDAHULUAN

Dengan perkembangan teknologi yang semakin canggih, cara manusia untuk melakukan kesehariannya mengalami pergeseran. Di era pandemik seperti ini, banyak kegiatan mengandalkan teknologi untuk memudahkan mereka. Salah satunya dalam memenuhi kebutuhan makanan. Orang sekarang kerap menggunakan aplikasi untuk memesan makanan secara online. Menurut Lembaga Demografi FEB UI pada 2021, aktivitas paling tinggi yang dilakukan online adalah layanan makanan online (97%), disusul dengan jasa pengiriman (76%), dan transportasi (75%) seperti grafik di Gambar 1.

Di Indonesia, layanan makanan *online* memberikan kontribusi hingga 34% dari tingkat pendapatan perkapita yang mana pebisnis kuliner juga diuntungkan[1]. Aplikasi dengan layanan pemesanan makanan *online* sudah sangat beragam mulai dari GoFood, GrabFood, ShopeeFood, Travelokaeats, Digiresto, dan lain sebagainya. Dari sisi pengguna, Gojek merupakan aplikasi dengan pengguna terbanyak dibanding pesaingnya. 70,4% masyarakat Indonesia telah menggunakan Gojek lalu disusul Grab dengan 45,7% [2]. Namun dari segi layanan pemesanan makanan online, GrabFood unggul dengan menghasilkan pesanan terbanyak selama 2021. Sekitar 109,4 triliun rupiah telah diraup Grab. Sedangkan Gojek memperoleh 98 triliun rupiah dengan kurang lebih 750 juta pesanan per bulan [3].

https://doi.org/10.25077/TEKNOSI.v8i2.2022.044-051



Gambar 1. Grafik Jenis Pengeluaran Digital Konsumen Per Bulan

Dengan keadaan yang menguntungkan ini, banyak pihak dapat merasakan peningkatan pendapatan, baik itu dari pihak UMKM yang bergerak dibidang kuliner sampai dengan pihak *driver* yang mengandalkan pekerjaannya hanya sebagai ojek *online* khususnya pengantar makanan. Pihak pengguna juga diuntungkan dengan kemudahan mereka memperoleh makanan tanpa harus keluar rumah. Apabila sistem ini dipertahankan, maka semua sisi aspek manusia menjadi simbiosis mutualisme yang baik.

Tetapi fenomena ini disertai dengan kendala yang dirasa masyarakat masih belum maksimal dalam kepuasan menggunakan layanan makanan *online* ini. Keluhan datang dari tiga faktor utama yaitu *service quality*, *customization*, dan kecepatan layanan. Hal ini dikarenakan dalam menangani pemesanan tersebut, mereka masih menggunakan *one-o-one system* dimana satu *driver* hanya melakukan pengantaran ekslusif ke satu pelanggan. Dengan waktu tunggu yang cukup panjang yaitu 30-60 menit, hal ini dirasa tidak efektif dari kecepatan kinerja dan banyaknya permintaan pesanan [4].

Multiple pemesanan makanan *online* diajukan untuk mengatasi masalah ini. Dengan adanya *multiple* pemesanan makanan, satu *driver* dapat dengan efisien menggunakan satu *trip*-nya mengantarkan lebih dari satu pesanan, sehingga waktu tunggu dapat digunakan dengan baik dan perjalanan sekaligus menyelesaikan beberapa pesanan. *Decision support system* harus dirancang untuk menentukan rute jalur mana yang harus dipilih dalam satu *trip* agar waktu dan jarak yang ditempuh minimal. Dengan menggunakan *Traveling Salesman Problem* (TSP), masalah ini dapat diselesaikan. Menggunakan dua algoritma yang akan dibandingkan dan dievaluasi yaitu *Branch&Bound Strategy* dan *Naive Approach* untuk mengetahui rute yang lebih bagus performanya.

Penelitian ini dilakukan untuk mengembangkan proses bisnis yang terjadi pada studi kasus pemesanan makanan online. Gap analisis yang didapatkan bahwa penelitian sebelumnya hanya memfokuskan dan mengimplementasikan metode Travelling Salesman Problem (TSP) untuk fixed route seperti pengantaran barang oleh kurir sedangkan penelitian ini dilakukan dengan kasus pemesanan makanan online yang lebih fleksibel rutenya dimana tiap orderannya bisa bervariasi dan ketentuan untuk harus mengambil pesanan terlebih dahulu.

Selain itu, penelitian sebelumnya hanya memakai TSP dengan Naive Approach saja sedangkan penelitian membandingkannya dengan Branch&Bound Strategy. Serta penelitian ini menggabungkan konsep sistem pendukung keputusan lain yaitu Simple Additive Weighting (SAW) sebagai perhitungan pendukungnya.
Tujuan dari diadakannya penelitian ini antara lain mengimplementasikanTSP dalam studi kasus pengambilan rute, membandingkan pendekatan algoritma *Branch&BoundStrategy* dan *NaiveApproach* serta memberikan keputusan rute terbaik yang harus diambil dengan waktu yang tercepat

Adapun manfaat yang dapat diambil dari penelitian ini adalah mengetahui algoritma yang palingsesuai dalam pengambilan rute *tripdriver* dalam pemesanan makanan *online* serta mengukur kinerja dengan algoritma *cost TSP*,*Branch & Bound*, dan *Naive Approach* dalam sistem pengambilan keputusan.

2. METODE

2.1. Decision Support System (DSS)

Decision Support System (DSS) adalah sistem yang membantu dan mendukung pengambilan keputusan. DSS dibagi dan dikategorikan menjadi sistem pendukung keputusan berbasis data, sistem pendukung keputusan berbasis komunikasi, sistem pendukung keputusan berbasis dokumen, sistem pendukung keputusan berbasis model, dan sistem pendukung keputusan berbasis pengetahuan [5]. Banyak sistem pendukung keputusan dengan algoritma yang spesifik untuk permasalahan yang khusus seperti AHP untuk menyelesaikan masalah perangkingan [6], machine atau deep learning untuk menyelesaikan masalah klasifikasi [7], dan TSP untuk menunjukkan pengambilan rute terbaik [8]. Dengan adanya sistem pendukung keputusan ini, pengguna mendapatkan beberapa rekomendasi apa yang harus dipilih atau dilakukan dengan beberapa perhitungan kriteria sebagai bahan pertimbangan untuk mendukung memastikan rekomendasi yang ditawarkan itu sudah sangat baik untuk dipilih.

2.2. Travelling Salesman Problem (TSP)

The Traveling Salesman Problem (TSP) adalah algoritma yang dihadapi dalam mengoptimalkan masalah kombinasi jalur yang diambil dari studi kasus seorang *sales* yang ingin berkeliling banyak kota dengan waktu dan jarak tempuh serta medan yang bermacam-macam setiap jalurnya yang direpresentasikan dalam berat/bobot skor. TSP membantu mencari kombinasi rute jalur mana yang paling pendek jarak tempuhnya dan berat/bobot skor terendah sehingga keputusan yang diambil dapat membuat pekerjaan menjadi efisien [9]. Penerapan TSP ditemukan di banyak bidang seperti manufaktur semikonduktor, logistik, dan transportasi [10]

Untuk menyelesaikan masalah dengan model TSP, sebuah grafik perjalanan rute dengan berat/bobot skor didefinisikan dengan $G = (V, E)$, dengan V adalah banyaknya *nodes* dan E adalah berat/bobot skor dari masing-masing rute. Perulangan dilakukan untuk mencari total skor dari kombinasi rute jalur yang mungkin ditempuh dalam formula (1). Dan dilakukan pencarian mana rute jalur yang memiliki skor terendah pada formula (2). Sedangkan

formula (3) untuk memastikan bahwa setiap nodes hanya dilalui satu kali [11, 21].

$$minF = \sum_{i \neq j} d_{ij} \times x_{ij} \quad (1)$$

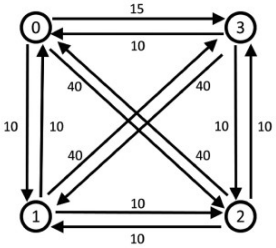
$$st, x_{ij} = \begin{cases} 1, e_{ij}\ pada\ rute\ yang\ optimal \\ 0, e_{ij}\ pada\ rute\ yang\ tidak\ optimal \end{cases} \quad (2)$$

$$\sum_{i \neq j} x_{ij} = 1, i \in V\ \&\ j \in V \quad (3)$$

Terdapat dua pendekatan algoritma TSP yaitu *Branch&Bound Strategy* dan *Naive Approach* yang memiliki sedikit perbedaan namun sama-sama mempunyai tujuan sama yaitu menghasilkan keputusan pemilihan rute jalur terbaik.

2.3. Naive Approach

Naïve Approach adalah suatu teknik pendekatan untuk menemukan perincian dalam hal parameter satu per satu [12]. Penyelesaian TSP dengan *Naive Approach* adalah algoritma yang standar seperti *brute force* untuk mencari semua kemungkinan kombinasi yang dapat terjadi dengan rumus (n-1)!, dimana n adalah banyak *nodes* yang kemudian dicari mana yang terbaik. Misalnya terdapat permasalahan seperti di Gambar 2, penyelesaian *Naive Approach* akan mendapatkan semua kombinasi dengan skor-skornya masing-masing yang kemudian akan dipilih mana yang memiliki skor terendah [13].



Gambar 2. Contoh Permasalahan TSP dengan *Naive Approach*

Tabel 1. Hasil *Naive Approach* dari permasalahan diatas

Rute yang diambil	Total skor
0-1-2-3	30
0-1-3-2	60
0-2-1-3	90
0-2-3-1	90
0-3-1-2	65
0-3-2-1	35

Dari Tabel 1, dapat dilihat bahwa rute dengan total skor yang terendah adalah rute dengan kombinasi jalur 0-1-2-3. Sehingga sistem akan merekomendasikan perjalanan 0-1-2-3 untuk menjadi rute agar perjalanan menjadi efisien.

2.4. Branch&Bound Strategy

Branch dan *Bound* umumnya digunakan untuk memecahkan masalah pemrograman integer. *Branch*dan*bound* adalah teknik yang tidak terbatas pada MILP (*Mixed Integer Linear Programming*) dan dapat digunakan untuk pencarian pohon secara umum [14].Langkah-langkahnya terutama dibagi menjadi tiga bagian: *branching*, *delimiting*, dan *pruning*[15]. Di antara mereka, *branching* mengacu pada membagi semua ruang solusi yang layak dan terus membaginya menjadi *subset* yang lebih kecil; *delimiting* mengacu pada menghitung batas atas dan batas bawah untuk setiap *node* anak setelah mencari *node* anak; *pruning* mengacu pada setelah setiap cabang, jika batas bawah yang dihitung lebih besar dari solusi optimal saat ini, rute cabang ini dihilangkan. Hal ini perlahan-lahan mempersempit ruang lingkup pencarian sampai solusi yang layak ditemukan. Solusi yang diperoleh dengan metode cabang dan terikat umumnya merupakan solusi optimal [16].

2.5. Simple Additive Weighting (SAW)

Simple Additive Weighting merupakan salah satu metode *MultiAttribute Decision Making* (MADM) [17]. *Simple Additive Weighting* (SAW) adalah metode untuk mencari bobot penjumlahan yang merepresentasikan keadaan atribut tertentu.sistem pendukung keputusan yang menggunakan *Simple Additive Weighting* mampu menampilkan hasil dari pembobotan dan perhitungan berdasarkan kriteria dengan sangat mudah dan efisien [18]. SAW memberikan nilai yang sesuai dengan mengkategorikan atribut sebagai biaya atau benefit seperti pada formula (4) di bawah ini [19].

$$r_{ij} = \begin{cases} \frac{x_{ij}}{Max(x_{ij})}, untuk\ atribut\ benefit \\ \frac{Min(x_{ij})}{x_{ij}}, untuk\ atribut\ biaya \end{cases} \quad (4)$$

Kemudian berat/bobot dari semua atribut direpresentasikan kedalam satu nilai dengan mengalikan masing-masing atribut (rij) dengan presentasi yang telah ditentukan (wj) [20].

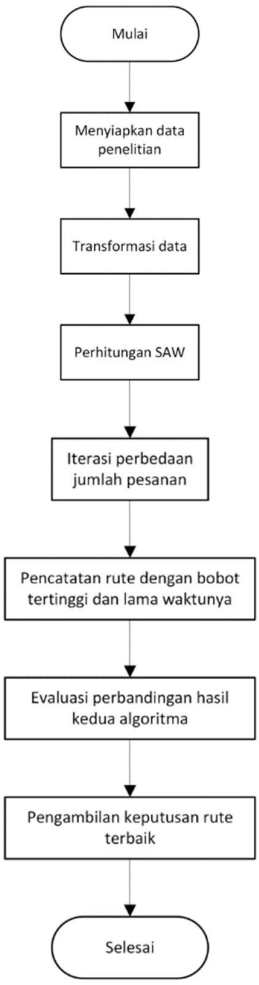
$$V_i = j \sum_{i-1}^{n} w_j \times r_{ij} \quad (5)$$

2.6. Metodologi Penelitian

Gambar 3 dibawah ini, menjelaskan terkait metode penelitian yang dilakukan dalam penelitian ini. Penelitian diawali dengan pengumpulan data berupa jarak, tingkat kesibukkan, dan skala prioritas yang diambil secara otomatis. Data yang telah dikumpulkan selanjutnya ditransformasi sesuai dengan kebutuhan dari perhitungan dari Simple Additive Weighting (SAW) dimana semua dikategorikan sebagai atribut benefit dan penerapan Travelling Salesman Problem (TSP). Kemudian penerapan Travelling Salesman Problem tersebut dilakukan dengan menggunakan dua algoritma berbeda, yakni: Branch &Bound Strategy dan Naive Approach.

Lalu dilakukan iterasi dengan perbedaan jumlah pesanan makanan yang setelahnya dilakukan pencatatan hasil dengan rute yang memiliki bobot tertinggi serta lama waktu prosesnya. Berikutnya dilakukan evaluasi untuk membandingkan hasil antara kedua algoritma tersebut. Terakhir, pengambilan keputusan pengambilan rutenya yang memiliki skot bobot/berat yang tinggi dan nantinya direkomendasikan ke driver untuk dijalankan sesuai rutenya.

https://doi.org/10.25077/TEKNOSI.v8i2.2022.044-051



Gambar 3. Metodologi Penelitian

2.7. Alat Penelitian

Penelitian ini menggunakan sistem operasi Windows 10 (64 bit), processor dengan *highest* GPU dengan RAM 8 GB, bahasa pemrograman Python dengan beberapa *packages library* pendukung, API maps dari googlemaps, dan beberapa aplikasi seperti Anaconda dan Jupyter Notebook.

3. HASIL

Tujuan dari penelitian ini adalah untuk memperoleh pemahaman mengenai algoritma pengambilan keputusan memilih rute terbaik untuk *multiple* pemesanan *online* dengan mengimplementasikan dua pendekatan algoritma berbeda yaitu *Branch& Bound Strategy* dan *Naive Approach*. Adapun tahapan dari pelaksanaan penelitian ini adalah sebagai berikut.

3.1. Persiapan dan Transformasi Data Penelitian

Persiapan dan transformasi data untuk penelitian ini sudah dilakukan. Terdapat atribut kriteria seperti jarak tempuh, tingkat kesibukkan dari pihak restoran, dan tingkat prioritas pelayanan. Jarak tempuh berisikan data *integer* yang merepresentasikan kilometer (km) antara satu *node* dengan *nodes* yang lain akan ditransformasikan kedalam skala 1-5 sesuai dengan tingkat dari terjauh ke terdekat. Tingkat kesibukkan pihak restoran berisikan informasi "Sangat Sibuk" (pesanan>10), "Sibuk" (6<pesanan<11), "Sedang" (1<pesanan<6), dan "Sepi" (Tidak ada pesanan yang berlangsung). Kemudian Tingkat kesibukkan restoran ini akan diubah menjadi kategori 1,2,3, dan 4.

Sedangkan Tingkat Kesibukkan antara rute ke pengguna diberikan berdasarkan kelancaran rute yang diambil dari API maps yag tersedia. Terdapat beberapa kriteria seperti "Macet", "Sedang", Cukup Lancar", "Lancar", tergantung dari status rute tersebut di API yang digunakan. Kemudian Tingkat tersebut dirubah menjadi kategori 1, 2, 3, dan 4.

Tingkat prioritas berisikan pesanan yang dilakukan sesuai urut yang muncul (diimplementasikan dengan metode FIFO).Yang terlihat pada Tabel 2 adalah situasi dimana kondisi Driver mendapat pesanan pertama dari User1 yang melakukan pemesanan di tempat makan A yang sibuk dan kemudian user2 melakukan pemesanan di tempat makan B yang sepi. Data tersebut akan ditransformasikan sebagai berikut.

Tabel 2. Dataset awal

Rute	Jarak (km)	Tingkat Kesibukkan	Prioritas
Driver-A	0.3 km	Sibuk	4
Driver-B	1.2 km	Sepi	2
A-B	4.2 km	Macet	1
A-User1	3.2 km	Cukup Lancar	5
A-User2	1.8 km	Lancar	2
B-User1	0.6 km	Cukup Lancar	3
B-User2	2.4 km	Lancar	3
User1-User2	3.8 km	Cukup Lancar	2

Informasi yang ada pada Tabel 2 diatas akan ditransformasikan ke dalam Tabel 3 dibawah dengan ketentuan yang telah disebutkan sehingga semuanya merupakan sebuah float ddengan range 1-5.

Tabel 3. Dataset awal setelah transformasi

Rute	Jarak (km)	Tingkat Kesibukkan	Prioritas
Driver-A	5	2	4
Driver-B	4	4	2
A-B	1	1	1
A-User1	2	2	5
A-User2	3	3	2
B-User1	4	2	3
B-User2	2	3	3
User1-User2	1.5	2	2

3.2. Pembobotan dengan Simple Additive Weighting

Setelah transformasi data dilakukan, kemudian semua atribut tersebut diisi dan ditransformasikan dengan *Simple Additive Weighting* (SAW) untuk mendapatkan berat/bobot skor (*weight*) masing-masing jalur. Nilai maksimum dicari dari tiap atributnya dan kemudian setiap data dibagi nilai maksimum setiap atributnya seperti pada Tabel 4.

Tabel 4. Dataset dengan SAW dengan nilai maksimum

Rute	Jarak	Tingkat Kesibukkan	Prioritas
Driver-A	5/5= 1	2/4= 0.5	4/5= 0.8
Driver-B	4/5= 0.8	4/4= 1	2/5= 0.4
A-B	4/5= 0.8	4/4= 1	1/5= 0.2
A-User1	2/5= 0.4	2/4= 0.5	5/5= 1
A-User2	3/5= 0.6	3/4= 0.75	2/5= 0.4
B-User1	4/5= 0.8	2/4= 0.5	3/5= 0.6
B-User2	2/5= 0.4	3/4= 0.75	3/5= 0.6
User1-User2	1,5/5= 0.3	2/4= 0.5	2/5= 0.4
Nilai Maksimum	**5**	**4**	**5**

Setelah didapatkan hasil tiap data dibagi dengan nilai maksimal, sesuai dengan Simple Additive Weighting, tiap hasilnya dikali dengan dengan bobot 50% untuk jarak, 30% dari tingkat kesibukkan, dan 20% dari prioritas seperti pada Tabel 5. Bobot tersebut didapatkan karena jarak merupakan atribut yang berpengaruh dalam pengantaran perjalanan driver ke masing-masing tujuan. Sedangkan tingkat kesibukkan disini diberikan 30% karena lama waktu antri dan kepadatan *traffic* jalanan merupakan hal yang krusial.

Tabel 5. Dataset SAW dengan bobot pengali

Rute	Jarak (50%)	Tingkat Kesibukkan (30%)	Prioritas (20%)
Driver-A	1*0.5= 0.5	0.5*0.3= 0.15	0.8*0.2= 0.16
Driver-B	0.8*0.5= 0.4	1*0.3= 0.3	0.4*0.2= 0.08
A-B	0.8*0.5= 0.4	1*0.3= 0.3	0.2*0.2= 0.04
A-User1	0.4*0.5= 0.2	0.5*0.3= 0.15	1*0.2= 0.2
A-User2	0.6*0.5= 0.3	0.75*0.3= 0.225	0.4*0.2= 0.08
B-User1	0.8*0.5= 0.4	0.5*0.3= 0.15	0.6*0.2= 0.12
B-User2	0.4*0.5= 0.2	0.75*0.3= 0.225	0.6*0.2= 0.12
User1-User2	0.3*0.5= 0.15	0.5*0.3= 0.15	0.4*0.2= 0.08

Data penelitian ini juga dibuat dan disimulasikan dengan empat kondisi berbeda dimana jumlah pesanan yang datang yang menjadi perbedaannya serta kondisi jarak, tingkat kesibukkan, dan prioritas.

4. PEMBAHASAN

Dari data simulasi yang sudah ditransformasikan menjadi matriks yang siap dihitung dengan metode Travelling Salesman Problem (TSP) dimana akan dicari kombinasi tiap jalur yang mungkin dari setiap nodes kemudian dicari total bobotnya. Selanjutnya didapatkan hasil beberapa kombinasi rute yang dapat direkomendasikan sebagai sebuah keputusan yang dapat dilihat dalam Tabel 8 berikut.

Nilai-nilai yang sudah didapatkan di Tabel 5 kemudian akan dijumlahkan untuk mendapatkan hasil bobot SAW di tiap-tiap nodes yang akan dijadikan nilai benefit jalur tersebut yang disajikan pada Tabel 6.

Tabel 6. Hasil perhitungan SAW

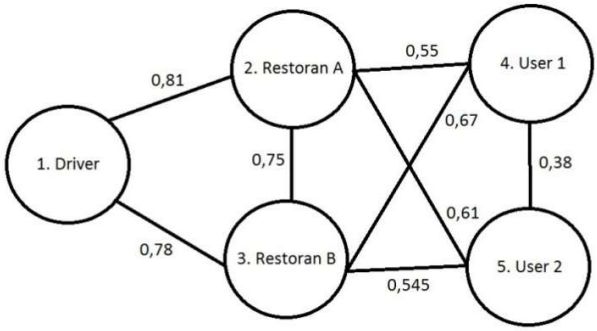
Rute	Jarak	Tingkat Kesibukkan	Prioritas	Hasil SAW
Driver-A	5	2	4	**0,81**
Driver-B	4	4	2	**0,78**
A-B	4	4	1	**0,75**
A-User1	2	2	5	**0,55**
A-User2	3	3	2	**0,61**
B-User1	4	2	3	**0,67**
B-User2	2	3	3	**0,545**
User1-User2	1,5	2	2	**0,38**

3.3. Penerapan Travelling Salesman Problem

Data-data dari hasil perhitungan beban/bobot dengan metode SAW tersebut disajikan dalam bentuk matriks yang memiliki jumlah baris dan kolom sesuai dengan jumlah kombinasi rute pesanan yang terjadi. Dan nantinya akan dapat diselesaikan dengan algoritma TSP baik dengan *Branch&Bound Strategy* maupun *Naive Approach*. Dengan data sesuai data transformasi, maka matriks TSP akan tersaji pada Tabel 7 dan digambarkan dalam bentuk grafik jaringannya pada Gambar 5.

Tabel 7. Traveling Salesman Problem dataset

Rute	Driver	A	B	User1	User2
Driver	0	0,81	0,78	0	0
A	0	0	0,75	0,55	0,61
B	0	0,75	0	0,67	0,545
User1	0	0	0	0	0,38
User2	0	0	0	0,38	0



Gambar 4. Grafik jaringan dari data matriks TSP

 https://doi.org/10.25077/TEKNOSI.v8i2.2022.044-051

Tabel 8. Hasil perhitungan metode TSP dengan Naive Bayes

Rute	Perhitungan	Total
1-2-3-4-5	0,81+0,75+0,67+0,38	**2,613**
1-2-3-5-4	0,81+0,75+0,545+0,38	2,488
1-2-4-3-5	0,81+0,55+0,67+0,545	2,575
1-3-2-4-5	0,78+0,75+0,55+0,38	2,463
1-3-2-5-4	0,78+0,75+0,61+0,38	2,523
1-3-5-2-4	0,78+0,545+0,61+0,55	2,485

Dengan menggunakan TSP Naive Bayes didapatkan hasil dengan rute 1-2-3-4-5 dimana driver memesan di restoran A kemudian restoran B lalu mengantarkan ke user 1 dan ke user 2 adalah rute dengan total skor/bobot maksimum yang merupakan keputusan yang terbaik yang harus diambil aplikasi. Sedangkan dengan *Branch and Bound Strategy*, rute hasilnya juga sama yaitu 1-2-3-4-5. Hal ini dikatakan bahwa hasil dari kedua algoritma tersebut sama untuk kasus dua pesanan sekaligus. Namun untuk lama waktu pemrosesan sampai kedua algoritma ternyata berbeda yaitu dengan Naive Bayes membutuhkan 0,07 detik, sedangkan dengan *Branch and Bound Strategy* membutuhkan hanya 0,03 detik.
Perbedaan juga terdapat apabila TSP dilakukan dengan kondisi berbeda seperti jumlah pesanan yang akan datang dengan simulasi random yang dijadikan dataset namun dengan perlakuan yang sama. Hal tersebut dapat dilihat dalam Tabel 9.

Hasilnya ditemukan bahwa, kedua algoritma tersebut berhasil merekomendasikan skor/bobot maksimum berat yang sama di setiap masalah dengan ukuran data yang berbeda. Pendekatan Naive Bayes menunjukkan semua kemungkinan kombinasi rute jika mereka semua memiliki skor maksimum tetapi dalam *Branch and Bound Strategy* yang telah dirancang, jika ada lebih dari satu kombinasi yang memiliki skor/bobot maksimal dalam hasil, rute yang diambil didasarkan pada urutan jumlah. Misalnya di data 3 kota, ada dua rute yang memiliki skor paling tinggi dari total berat,1-4-6-7-3-5-2 dan 1-4-6-7-3-2-5, semua jalur 5 pertama adalah sama tetapi *Branch and Bound Strategy* memilih 2-5-1 daripada 5-2-1 karena urutan nomor 2 dan 5. Begitu pula pada kondisi 4 pesanan terdapat dua rute dengan bobot yang sama dan berbeda di 3 edges tengah saja yaitu 5-6-4 dan 6-4-5. Sedangkan dengan jumlah 5 pesanan memiliki satu rute dengan bobot yang sama baik di algoritma Naive Approach dan Branch&Bound.

Tabel 9. Hasil empat kondisi dengan jumlah pesanan berbeda

Jumlah Pesanan	Hasil Bobot Maksimum		Rute
	Naive Bayes	Branch & Bound	
2 Pesanan	2,613	2,613	1-2-3-4-5
3 Pesanan	3,54	3,54	1-4-6-7-**3-5-2** 1-4-6-7-**3-2-5***
4 Pesanan	5,72	5,72	1-3-2-**5-6-4**-7-9-8 1-3-2-**6-4-5**-7-9-8*
5 Pesanan	7,891	7,891	1-5-3-4-2-7-9-6-8-10-11

Note: rute dengan simbol bintang adalah rute tambahan alternatif lain yang ditemukan ole algoritma Branch&Bound.

Rute yang diambil sebagai rekomendasi hasil keputusan adalah rute yang terdiscovered atau diketahui lebih dahulu. Sehingga untuk simulasi 3 pesanan sekaligus, rute 1-4-6-7-3-5-2 akan direkomendasikan oleh sistem kepada driver dan untuk simulasi 4 pesanan sekaligus, rute 1-3-2-5-6-4-7-9-8 yang akan direkomendasikan karena didapatkan terlebih dahulu.

Tabel 10. Perbandingan hasil waktu yang dibutuhkan

Jumlah Pesanan	Waktu (detik)	
	Naive Bayes	Branch & Bound
2 Pesanan	0,07	**0,003**
3 Pesanan	0,030	**0,011**
4 Pesanan	19,31	**0,238**
5 Pesanan	172,4	**0,412**

Dan untuk kompleksitas waktu, seperti yang tertulis dalam tabel bahwa *Branch and Bound Stratgey* membutuhkan lebih sedikit waktu untuk menemukan solusi dari TSP untuk setiap ukuran data. Pada data pertama, perbedaannya hanya 0,015 detik, tetapi pada data dengan 5 pesanan, perbedaan waktu yang dibutuhkan cukup lama. Jadi perbedaan waktu yang dibutuhkan antara kedua algoritma ini meningkat seiring dengan bertambahnya ukuran jumlah pesanan.

Tetapi kompleksitas waktu dari Naive Bayes yang terlalu jauh melalui jumlah nodes yang harus dikunjungi. Untuk dua dataset pertama, perbedaannya tidak terlalu tinggi, tetapi mulai untuk pesanan yang banyak kesenjangan kompleksitas waktu jauh lebih tinggi karena kompleksitas waktu Naive Bayes adalah n!. Dan untuk *Branch and Bound Strategy*, kompleksitas waktu kurang dari 1 detik dalam percobaan ini. *Branch and Bound Strategy* secara signifikan mengurangi lebih banyak kompleksitas waktu dari pendekatan Naive Bayes dengan beberapa jumlah pesanan yang baik untuk menemukan masalah TSP.Untuk *Branch and Bound Strategy*, biaya untuk menemukan solusi untuk masalah pengambilan keputusan untuk n nodes cukup besar dan secara asimptotik $O(n^3)$ dan di bawah beberapa asumsi lagi kompleksitas algoritma adalah $O(n^{3\ln(n)})$ yang berarti tingkat biaya n adalah 3 secara stabil. Dan untuk pendekatan Naive Bayes, biaya algoritma adalah O((n-1)!) dan untuk menetapkan masalah untuk n pesanan. Semakin banyak ukuran n, derajat n akan meningkat tinggi.

5. KESIMPULAN

Untuk menemukan rute terbaik yang harus diambil oleh driver secara efisien, pendekatan Naive Bayes memberikan semua kemungkinan rute permutasi yang memiliki skor/bobot maksimum. Tetapi dibutuhkan ruang besar dan waktu yang lama untuk menemukan hasil untuk sejumlah multiple pesanan. Dan berdasarkan kompleksitas waktu dan biaya algoritma yang diambil dalam masalah TSP ini untuk mengambil keputusan, *Branch and Bound Strategy* adalah algoritma terbaik dan efisien untuk digunakan untuk memecahkan masalah multiple pemesana

makanan online ini. Algoritma ini hanya membutuhkan biaya setidaknya $O(n^3)$ dan hanya membutuhkan waktu kurang dari 1 detik untuk menemukan solusi dalam kumpulan data besar yang kira-kira 99,5% efisien daripada menggunakan pendekatan Naive Bayes yang membutuhkan waktu paling lama tiga menit (180 detik) untuk dilakukan dan biayanya adalah O((n-1)!) dengan total jumlah 5 pesanan.

UCAPAN TERIMA KASIH

Terima kasih kepada Universitas Bunda Mulia khususnya tim Penelitian, Pengembangan, dan Pengabdian kepada Masyarakat (P3M) Universitas Bunda Mulia yang telah membiayai penelitian ini dalam program Hibah Internal UBM Kampus Serpong sampai selesai.

DAFTAR PUSTAKA

[1] A. Tumpuan, "Peranan Aplikasi Go Food Terhadap Perkembangan Bisnis Kuliner,"*TOURISM: Jurnal Travel, Hospitality, Culture, Destination, and MICE*, vol. 3, pp. 26-30, 2020

[2] V. B. Pambudi and K. Khuzaini, "Analisis Persepsi Nilai Terhadap Kepuasan Pelanggan Pada Layanan Go-Food Di Kalangan Mahasiswa Stiesia Surabaya," *Jurnal Ilmu Dan Riset Manajemen (JIRM)*, vol. 9, pp., 2020.

[3] H. Akhmadi, A. R. Alfatah, and Susanawati, "Generation Z consumer's preferences for online food ordering application: a study of Gofood and Grabfood", *E3S Web Conf*, vol. 316, pp. , 2021.

[4] T. S. Kumar, "Data Mining Based Marketing Decision Support System Using Hybrid Machine Learning Algorithm," In *Journal of Artificial Intelligence and Capsule Networks*, vol. 2, pp. 185-193, 2020.

[5] C. Winarto, "Analisis Faktor-Faktor Yang Mempengaruhi Keputusan Konsumen Dalam Menggunakan Jasa Grabfood Di Surabaya," in *Seminar Nasional Ilmu Terapan*, 2019, pp. E6-E6.

[6] N. K. A. P. Sari, "Implementation of the AHP-SAW Method in the Decision Support System for Selecting the Best Tourism Village," *Jurnal Teknik Informatika C.I.T Medicom*, vol. 13, pp. 23-32, 2021.

[7] S. Hosseini, "A decision support system based on machined learned Bayesian network for predicting successful direct sales marketing,"*Journal of Management Analytics*, vol. 8, pp. 295-315, 2021.

[8] D. E. Gomes, M. I. D.Iglésias, A. P. Proença, T. M. Lima, and P. D. Gaspar, "Applying a Genetic Algorithm to a m-TSP: Case Study of a Decision Support System for Optimizing a Beverage Logistics Vehicles Routing Problem," *Electronics (Switzerland)*, vol. 10, pp. 2298, 2021.

[9] K.A.F.A. Samah, N. Sabri, R. Hamzah, R. Roslan, N.A. Mangshor, and A.A.M. Asri, "Brute force algorithm implementation for traveljoy travelling recommendation system," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 16, pp. 1042-1049, 2019.

[10] I. K.Gupta, S.Shakil, and S.Shakil, "A hybrid GA-PSO algorithm to solve traveling salesman problem," In *Advances in Intelligent Systems and Computing*, vol. 798, pp. 453-362, 2019.

[11] J. C. Zhang, "Comparison of various algorithms based on TSP solving," *Journal of Physics: Conference Series*, vol. 2083, pp. 32007, 2021.

[12] H. Qi, Y. Shi, X. Mu, and M. Hou, "Knowledge Granularity for Continuous Parameters," In *IEEE Access*, vol. 9, pp. 89432-89438, 2021.

[13] Y. Begnio, A. Lodi, A. Prouvost, "Machine learning for combinatorial optimization: A methodological tour d'horizon," In *European Journal of Operational Research*, vol. 290, pp. 405-421, 2021.

[14] A. Ibrahim, R. A. Surya, "The Implementation of Simple Additive Weighting (SAW) Method in Decision Support System for the Best School Selection in Jambi," In *Journal of Physics: Conference Series*, vol. 1338, pp. 012054, 2019.

[15] S. Goyal, "A Survey on Travelling Salesman Problem," *Midwest Instruction and Computing Symposium*, vol. , pp. 1-9, 2018.

[16] A.Cahyapratama& R.Sarno, "Application of Analytic Hierarchy Process (AHP) and Simple Additive Weighting (SAW) methods in singer selection process," In *International Conference on Information and Communications Technology (ICOIACT)*, vol., pp. 234-239, 2018.

[17] S. Violina, "Analysis of Brute Force and Branch & Bound Algorithms to solve the Traveling Salesperson Problem (TSP)," *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, pp. 1226–1229, 2021.

[18] A. Arigliano, T. Calogiuri, G. Ghiani, E. Guerriero, "A branch-and-bound algorithm for the time-dependent travelling salesman problem," *Networks*, vol. 72, pp. 382-392, 2018.

[19] N. Nurmalini, R. Rahim, "Study Approach of Simple Additive Weighting For Decision Support System," *International Journal of Scientific Research in Science and Technology*, vol. 3, pp. 295–315, 2017.

[20] V. Sihombing, V. M. M. Siregar, W. S. Tampubolon, M. Jannah,R. Risdalina, and A. Hakim, "Implementation of simple additive weighting algorithm in decision support system," *IOP Conference Series: Materials Science and Engineering*, vol. 1088, pp. 12014, 2021.

[21] Muhammad Irfan. Penyelesaian Travelling Salesman Problem (TSP) menggunakan Algoritma Hill Climbing dan Matlab". Matematika, 17(1), 2018.

[22] Chunhua Fu, Lijun Zhang, Xiaojing Wang, and Liying Qiao. "Solving tsp problem with improved genetic algorithm". In AIP Conference Proceedings, volume 1967, page 040057. AIP Publishing LLC, 2018.

[23] Jan Scholz. "Genetic algorithms and the traveling salesman problem a historical review". arXiv preprint arXiv:1901.05737, 2019.

[24] Cheikhrouhou, O. Khoufi, I. "A comprehensive survey on the Multiple Traveling Salesman Problem: Applications, approaches and taxonomy". Comput. Sci Journal. Rev, 40, 100369, 2021.

[25] Yagamishi, K., Gantalao, C., Tiu, A.M., Tanaid, R.A, Medalla, M.E., Abellana, D.P., Selerio, E.,dan Ocampo, L., "Evaluating the destination management performance of small islands with the fuzzy best-worst method and fuzzy

simple additive weighting", Current Issues in Method and Practice Tourism Journal, 2022.
[26] Taillard, E.D., "A linearithmic heuristic for the travelling salesman problem". European Journal of Operational Research, 2022.

BIODATA PENULIS

Bhustomy Hakim
Seorang dosen Universitas Bunda Mulia yang fokus mengajar dan melakukan penelitian di bidang *Computer Science*. Menerima gelar Master of Engineering pada 2021 di Harbin Institute of Technology jurusan Computer Science.

Fendyanto
Seorang mahasiswa Universitas Bunda Mulia semester 7 jurusan Sistem Informasi yang aktif melakukan penelitian di bidang *Computer Science*. Menerima gelar Bachelor estimasi pada tahun 2023.