Evaluasi Rute Pelayaran Kapal dengan Pendekatan *Modified Minimum Spanning Tree*

Felix Julio Tjoea[1], Siana Halim[2]

Abstract: This research aims to optimize the shipping routes of the largest container shipping liner company in Indonesia. Route optimization is essential for improving the efficiency of the company in the face of increasingly intense business and industry competition. In this study, a multi-objective approach is used, considering both demand and distance factors. The proposed evaluation model utilizes a modified minimum spanning tree (MST) with probabilistic calculations based on historical data. The modification takes into account probabilistic factors derived from past data, enabling better decision-making.Additionally, for scenarios where the route needs to return to the depot, the Christofides algorithm is employed. The Christofides algorithm is used to solve the Traveling Salesman Problem (TSP), where the objective is to find the shortest path that visits all points and returns to the starting point. This algorithm is based on the MST. The results of the conducted case studies using this model show an improvement in the performance of shipping routes, with a 1.13% increase in ship utility and a 2.83% decrease in total distance. Therefore, it can be concluded that the evaluation model for shipping routes, employing a multi-objective approach, modified MST with probabilistic considerations, and the Christofides algorithm, can provide more efficient solutions and opportunities for operational improvements in the company.

Keywords: transportation optimization, traveling salesman problem, minimum spanning tree, christofides algorithm

Pendahuluan

Pada tahun 2021, UNCTAD melaporkan bahwa perdagangan maritim internasional mengalami pertumbuhan sebesar 3,2 persen setelah mengalami penurunan pada tahun sebelumnya. Namun, pandemi yang masih berlangsung dan masalah rantai pasok global masih menjadi hambatan bagi perdagangan, meskipun permintaan kargo kontainer meningkat pada tahun 2022 yang diproyeksikan sebesar 1,2 persen, dan diperkirakan akan meningkat menjadi 1,9 persen pada tahun 2023. Proyeksi UNCTAD untuk tahun 2022 menunjukkan bahwa pertumbuhan perdagangan maritim hanya akan tumbuh moderat sebesar 1,4 persen, dan proyeksi untuk periode 2023-2027 rata-rata sebesar 2,1 persen per tahun, yang lebih lambat dibandingkan dengan rata-rata tiga dekade sebelumnya. Perlambatan ini dipengaruhi oleh tantangan makroekonomi, melemahnya ekonomi China, kenaikan inflasi dan biaya hidup, dan adopsi model bisnis "*just-in-case*" dan "*just-enough*" oleh perusahaan untuk membangun ketahanan, dan kontinuitas rantai pasok (UNCTAD [1]).

[1,2] Fakultas Teknologi Industri, Program Studi Teknik Industri, Universitas Kristen Petra. Jl. Siwalankerto 121-131, Surabaya 60236. Email: c13190111@john.petra.ac.id, halim@petra.ac.id

PT. X merupakan perusahaan pelayaran peti kemas terbesar di Indonesia dan terdaftar dalam 20 besar perusahaan pengiriman peti kemas terbesar di dunia. Perusahaan ini memiliki 37 kantor cabang yang tersebar di seluruh Indonesia dan kapal kontainer dengan total kapasitas 60.000 TEUs, serta 2 terminal peti kemas di Tanjung Perak, Surabaya dan Tanjung Priok, Jakarta, serta 5 kapal curah dengan kapasitas 50.000 DWT. Salah satu tujuan dari semua aktivitas industri adalah menekan biaya operasional seminimal mungkin untuk mendapatkan keuntungan yang maksimal dan meningkatkan daya saing. Semakin ketatnya persaingan usaha mengharuskan perusahaan agar terus beradaptasi dan melakukan efisiensi serta optimalisasi proses operasional untuk menjawab tantangan global yang ada di sektor perdagangan maritim saat ini, seperti pelemahan pertumbuhan industri. Untuk menjawab kebutuhan untuk optimasi serta tantangan global, manajemen operasional dalam perusahaan pelayaran menjadi semakin penting (Salleh *et al.* [2]). salah satu komponen biaya utama dalam bisnis proses sebuah perusahaan pelayaran adalah *ship cost*, yang didalamnya kemudian ada beberapa komponen, terkhususnya *fuel cost*. Salah satu cara untuk melakukan efisiensi penggunaan bahan bakar dan menjaga kehandalan operasional pelayaran adalah dengan mengoptimalkan rute pelayaran. Hal ini

dapat mengurangi biaya operasional, meningkatkan produktivitas, serta mengurangi dampak lingkungan. Ini diharapkan dapat membantu perusahaan mengatasi tantangan global dan memperkuat keberlangsungan bisnisnya (Meng *et al.* [3]). Berdasarkan wawancara awal dengan perusahaan dan temuan dari data masa lalu, optimasi rute tidak hanya terbatas pada biaya terendah, tetapi juga dapat dinilai dari segi utilitas atau *ship occupancy ratio*, yaitu rasio antara keterisian kapal dengan kapasitas optimalnya. Selain itu, harus memperhatikan tingkat pemenuhan *demand* dari konsumen sehingga rute yang dibuat dapat memenuhi permintaan pengantaran barang dengan jarak dan *utility* terbaik. Saat ini, proses pembuatan rute oleh perusahaan masih dilakukan secara manual dan tidak memiliki langkah yang sistematis dalam penentuannya. Oleh karena itu, sangat mungkin terjadi bahwa rute yang dipilih oleh perusahaan bukanlah rute yang paling optimal. Ada beberapa rute pelayaran yang memiliki tingkat *utility* di bawah 50%. Tentunya, dalam segi operasional, hal ini menunjukan sistem operasional yang tidak efisien. Sehingga, perlu adanya perbaikan yang terstruktur dalam hal penentuan rute pelayaran tersebut.

Permasalahan yang dihadapi oleh perusahaan adalah permasalahan klasik dalam bidang *travelling salesman problem* (TSP), yang menggambarkan seorang pedagang yang harus menjual ke beberapa rute (*node*) dengan batasan bahwa pedagang tersebut harus kembali ke tempat asal (*depot*). Tantangan dalam menyelesaikan permasalahan ini adalah mencari rute atau lintasan terpendek. Cara termudah untuk menyelesaikannya adalah dengan membandingkan keseluruhan kemungkinan rute yang bisa dibentuk, yang dikenal sebagai metode *bruteforce*. Namun, metode ini menjadi sangat berat secara komputasi ketika jumlah *node* meningkat. Oleh karena itu, diperlukan cara lain untuk mendapatkan solusi optimal dari TSP, yaitu menggunakan *approximation* atau algoritma pendekatan yang memiliki solusi yang cukup dekat dengan solusi asli, namun dengan waktu penyelesaian lebih cepat. Salah satu algoritma *approximation* ini adalah algoritma Christofides yang mampu memberikan solusi optimal dengan waktu yang relatif lebih singkat daripada metode *bruteforce*, dengan maksimal rentang galat sebesar 50% (Christofides [4]).

Algoritma Christofides diciptakan oleh Nicos Christofides [4] dan merupakan metode pemecahan TSP yang berdasarkan *minimum spanning tree*. Algoritma ini mencari nilai atau bobot minimum dari sebuah lintasan sehingga akan menghasilkan irisan dari suatu *graph* yang memiliki bobot optimal. Kemudian, hasil dari algoritma MST tersebut akan digunakan untuk membentuk sirkuit Euler, sehingga dapat menghasilkan sebuah rute perjalanan yang dimulai dari suatu depot dan akan kembali pada akhir rute. Penelitian ini akan fokus pada pemecahan masalah TSP serta melihat dan mengevaluasi efisiensi penggunaan algoritma Christofides pada kasus di perusahaan, berdasarkan parameter *utility*, jarak, dan pemenuhan *demand* pada tiap rute yang digunakan. Fleksibilitas dari algortima yang berdasarkan MST ini, kemudian dapat menjadi nilai tambah, mengingat bahwa ada beberapa rute di perusahaan, yang bukan merupakan sebuah rute TSP, namun juga perlu dilakukan evaluasi.

Metode Penelitian

Penelitian ini menggunakan metode penyelesaian utama yaitu algoritma Christofides, yang berbasiskan MST pada langkah pertamanya, dimana dilakukan modifikasi terhadap fungsi tujuan dari MST untuk dapat mengakomodir konteks dan kebutuhan dari masalah perusahaan.

Graf

Menurut Munir [5], sebuah graf terdiri dari simpul-simpul atau *node* yang tidak kosong dan sisi-sisi atau *edge* yang menghubungkan simpul-simpul tersebut. Graf G dapat direpresentasikan dalam notasi matematika sebagai tupel (V, E), di mana V adalah himpunan simpul yang tidak kosong dan E adalah himpunan sisi yang menghubungkan dua simpul dalam V. Terdapat dua jenis graf berdasarkan sifat sisi-sisinya:

- Graf tak berarah (*undirected graph*)
  Sisi-sisi dalam graf ini tidak memiliki arah tertentu, sehingga urutan atau arah tidak diperhatikan dalam penghubungannya.
- Graf berarah (*directed graph*)
  Sisi-sisi dalam graf ini memiliki arah tertentu, sehingga urutan atau arah menjadi penting dalam penghubungannya.

Terminologi

Berikut adalah beberapa istilah yang umum digunakan dalam penelitian ini (Munir [5]):

- Derajat simpul
  Jumlah sisi yang terhubung dengan simpul tersebut dan ditulis sebagai $deg(v)$. Teorema jabat tangan (*handshaking theorem*) menyatakan bahwa dalam graf tak berarah $G = (V, E)$ dengan n sisi, berlaku $2n = \Sigma deg(v)$ untuk setiap simpul v di V.

- Lintasan
  Terjadi ketika ada serangkaian sisi yang dapat menghubungkan dua simpul, mulai dari simpul awal hingga simpul akhir.
- Sirkuit
  Jenis lintasan yang dimulai dan berakhir di simpul yang sama.
- Graf terhubung
  Terhubung jika terdapat setidaknya satu lintasan yang menghubungkan setiap pasang simpul dalam graf tersebut. Jika ada setidaknya satu pasangan simpul yang tidak memiliki lintasan yang menghubungkannya, graf tersebut disebut graf tak terhubung (*disconnected graph*).

Weighted Graph

Graf yang memiliki bobot pada sisinya disebut graf berbobot. Dalam jenis graf ini, bobot dapat memiliki berbagai makna tergantung pada konteks penggunaan graf. Dalam konteks permasalahan *travelling salesman problem* yang akan dibahas, graf berbobot digunakan untuk menggambarkan lokasi yang ingin dikunjungi. Setiap simpul dalam graf mewakili sebuah lokasi, sedangkan setiap sisi dalam graf mewakili jarak antara dua lokasi yang berbeda (Munir [5]).

Travelling Salesman Problem

Travelling salesman problem atau bisa disingkat TSP adalah masalah optimasi kombinatorial klasik dalam bidang *computer science* dan matematika (Gohil *et al.* [6]; Ahmed *et al.* [7]). TSP melibatkan penentuan lintasan terpendek yang melibatkan kunjungan pada sejumlah titik tertentu, dengan memperhitungkan jarak atau biaya perjalanan antar titik-titik tersebut (Genova dan Williamson [8]). Dalam TSP, terdapat sejumlah lokasi yang harus dikunjungi, dan setiap lokasi harus dikunjungi tepat satu kali. Tujuan utama TSP adalah mencari urutan kunjungan yang meminimalkan total jarak atau biaya yang harus ditempuh. Keunikannya terletak pada fakta bahwa setiap lokasi harus dikunjungi tepat satu kali dan lintasan harus membentuk siklus tertutup, sedemikian hingga awal dan akhir dari lintasan adalah lokasi yang sama (Ahmed dan Dayel [9]).

Permasalahan TSP dapat didefinisikan sebagai sebuah grafik berbobot lengkap $G = (V, E)$, di mana V merupakan himpunan titik atau kota yang harus dikunjungi, dan E adalah himpunan semua kemungkinan jalur atau sisi yang menghubungkan setiap pasangan kota. Setiap sisi $[\,i\,,j\,]$ memiliki bobot atau jarak yang terkait, dan tujuan adalah mencari tur (lintasan) yang melalui semua titik dengan biaya minimum (Genova dan Williamson [8]). TSP memiliki variasi tergantung pada sifat tambahan yang diberikan pada masalah, seperti TSP simetris (jarak dari lokasi A ke lokasi B sama dengan jarak dari lokasi A ke lokasi B) atau TSP asimetris (di mana jarak antara dua lokasi bisa berbeda). TSP juga dapat memiliki batasan tambahan, seperti batasan waktu, kapasitas kendaraan, atau larangan kunjungan antara beberapa pasangan titik. Karena TSP diklasifikasikan sebagai masalah NP-*hard*, tidak ada algoritma efisien yang dapat menyelesaikan TSP dalam waktu polinomial untuk semua kasus. Oleh karena itu, dalam mencari solusinya, digunakan berbagai pendekatan algoritmik dan heuristik untuk mencari solusi yang memadai atau mendekati solusi optimal dalam waktu yang wajar.

Model Matematis

Model matematis dalam *travelling salesman problem*, dapat didefinisikan seperti pada Persamaan (1) hingga Persamaan (4).

$$Minimize\ Z = \sum_{i=1}^{n}\sum_{j=1}^{n} d\,[\,i\,,j\,] * c\,[\,i\,,j\,] \tag{1}$$

$$\sum_{j=1}^{n} x\,[\,i\,,j\,] = 1, \qquad i = 1,..n \tag{2}$$

$$\sum_{i=1}^{n} x\,[\,i\,,j\,] = 1, \qquad j = 1,..n \tag{3}$$

$$x\,[\,i\,,j\,] \in \{\,0\,,1\,\}, \qquad i,j = 1,..n \tag{4}$$

Permasalahan TSP akan berusaha mencari rute terpendek, dimana rute yang ditempuh akan mulai dan berakhir pada nodes yang sama, dan mengunjungi nodes lain tepat satu kali. Pada Persamaan (1) merupakan fungsi tujuan dari masalah TSP, dimana akan meminimalkan nilai perkalian antara jarak dengan *cost*, untuk setiap kombinasi *nodes* $[\,i\,,j\,]$, yang sejumlah n. Persamaan (2) dan (3) menyatakan fungsi pembatas, sedemikian hingga setiap kombinasi nodes $[\,i\,,j\,]$ hanya boleh dikunjungi satu kali dalam rute, dan untuk Persamaan (4) sebagai bentuk biner dari keputusan untuk mengunjungi *nodes* (bernilai 1) atau tidak mengunjungi *nodes* (bernilai 0).

Minimum Spanning Tree

Minimum spanning tree pada sebuah graf berbobot adalah pohon yang terdiri dari subgraf graf berbobot tersebut dengan bobot total yang minimal (Kafaji dan Shiker [10]). Ada dua metode umum untuk mencari MST, yaitu dengan menggunakan algoritma Prim dan algoritma Kruskal. Pada penelitian ini akan

memfokuskan pada algoritma Prim. Jika dimisalkan T adalah pohon yang ingin dibentuk, G adalah graf berbobot, dan n adalah jumlah simpul dalam G. Berikut adalah langkah-langkah dalam algoritma Prim (Hussein dan Klein [11]):

- Mengambil sisi dengan bobot minimum dari G dan tambahkan ke T.
- Memilih sisi S dengan bobot minimum yang terhubung dengan simpul di T, sehingga tidak membentuk sirkuit di T.
- Memasukkan sisi S pada langkah 2 ke graf T.
- Mengulangi langkah 2 sebanyak n - 2 kali.

Modified Minimum Spanning Tree

Pada Persamaan (1), fungsi tujuan permasalahan TSP konvensional merupakan minimasi dari penjumlahan matriks jarak dengan *cost*, yang merupakan *single objective function*. Sedangkan, dalam studi kasus penelitian perusahaan pelayaran ini, konteks *cost* diubah menjadi jumlah permintaan atau *demand*, dan dilakukan pembobotan untuk jarak dan *demand* sehingga akan dihasilkan *weight* yang dapat mengakomodasi *multiple objective* pada permasalahan TSP. Perhitungan *weight* akan mengambil referensi perhitungan dari penelitian yang dilakukan oleh Mandi *et al.* [12]. Dimana, penelitian tersebut menggunakan pendekatan data *driven* dan probabilistik untuk menyelesaikan permasalahan TSP, sehingga apabila dikombinasikan, *weight* yang akan digunakan pada penelitian ini akan menggunakan kombinasi yang seimbang antara jarak dan perhitungan total *demand* pada tiap rangkaian pelabuhan. Untuk perhitungan *demand*, menurut Mandi *et al.* [12], dapat menggunakan konsep *conditional probability* dengan pendekatan markov *counting*, dimana secara matematis dapat didefinisikan dalam Persamaan (5).

$$\Pr(r|s) = \frac{D_{su}}{\sum_u D_{su}} \tag{5}$$

Pada persamaan (5), notasi D melambangkan *demand* atau permintaan pada setiap kombinasi pelabuhan yang akan menghitung *conditional probability* untuk *demand*, dimana pada konteksnya dan praktiknya, akan dilakukan perhitungan *demand* pada suatu pelabuhan s ke u dan membandingkannya dengan total *demand* dari s. Sehingga, akan didapatkan pembobotan yang adil bagi rute-rute dari s. Secara mudah, formula ini dapat menjelaskan dan menggambarkan tingkat "kepentingan" untuk memenuhi suatu permintaan dari suatu pelabuhan asal ke tujuan, dengan berdasarkan seluruh data permintaan dari pelabuhan asal. Kemudian, untuk parameter jarak akan menggunakan Persamaan (6) yang merupakan *distance-based probability* yang digunakan untuk menghitung kemungkinan berdasarkan pangkat eksponen dari jarak. Secara konsepual, semakin jauh jarak antara kedua pelabuhan, maka kemungkinan tingkat "kepentingan" akan berkurang secara eksponsial, seperti terlihat pada persamaan (6). Kemudian, dilakukan pembagian jarak dengan nilai 1000, dimana angka 1000 ini dimaksudkan agar hasil hasil formula dapat mendekati standar *probability demand*.

$$\Pr(r|s) = \frac{e^{-\frac{d_{sr}}{1000}}}{\sum_u e^{-\frac{d_{su}}{1000}}} \tag{6}$$

Secara kenyataan, pelabuhan yang lebih jauh akan memiliki kemungkinan yang kecil untuk dikunjungi terlebih dahulu. Sehingga, pada formula tersebut juga akan membandingkan invers eksponen dari jarak antara s ke r dengan invers eksponen dari keseluruhan total jarak untuk rute-rute dari s.
Setelah mendapatkan angka *probability* berdasarkan *demand* pada Persamaan (5) dan berdasarkan jarak pada Persamaan (6), akan dihitung *weight* akhir dengan bobot α setara, yaitu 0.5. Sehingga, untuk perhitungan *weight* akhir akan mengikuti formula pada persamaan (7).

$$w(r|s) = \begin{cases} \alpha \frac{D_{su}}{\sum_u D_{su}} + (1-\alpha) \frac{e^{-d_{sr}}}{\sum_u e^{-d_{su}}}, & D > 0 \\ 100, & D = 0 \end{cases} \tag{7}$$

Sebagai solusi praktis, untuk kombinasi rute yang tidak memiliki permintaan atau *demand*, hasil dari *weight* akan diberikan angka 100, sesuai dengan konsep *Big* M untuk menghindari terpilihnya rute tersebut (Cococcioni dan Faschi [13]). Untuk rute yang memiliki permintaan, *weight* merupakan setengah dari nilai penjumlahan *probability* berdasarkan *demand* dan jarak. Kemudian, hasil dari perhitungan tersebut akan dipangkatkan -1, karena pada konteksnys, kita menginginkan bahwa semakin besar *demand*, dan semakin kecil *distance*, maka kita menginginkan bahwa *weight* nya akan semakin kecil (berkebalikan). Sehingga, dari Persamaan (5), (6), (7), akan dilakukan modifikasi terhadap fungsi tujuan TSP pada Persamaan (1), seperti pada Persamaan (8), yang dimaksudkan untuk melakukan minimasi terhadap *weight*.

$$\begin{aligned} &Minimize\ Z \\ &= \sum_{r=1}^{u} \sum_{s=1}^{u} \begin{cases} (\alpha \frac{D_{su}}{\sum_u D_{su}} + (1-\alpha) \frac{e^{-d_{sr}}}{\sum_u e^{-d_{su}}})^{-1}, & D > 0 \\ 100, & D = 0 \end{cases} \end{aligned} \tag{8}$$

Algoritma *minimum spanning tree* dapat sebagai penyelesaian fungsi tujuan tersebut. Dalam konteks perusahaan pelayaran, dimana ada dua kondisi yang terjadi, yaitu rute harus kembali ke *depot* dan tidak kembali, maka modifikasi penyelesaian TSP menggunakan MST ini akan menjadi lebih fleksibel, dikarenakan dapat menyelesaikan dan menyesuaikan kedua kondisi tersebut.

Algoritma Christofides

Algoritma Christofides digunakan untuk menghasilkan solusi aproksimasi dalam TSP pada graf tak berarah berbobot lengkap. Namun, graf harus memenuhi syarat ketidaksamaan segitiga. Ketidaksamaan segitiga menyatakan bahwa panjang sisi dalam segitiga harus lebih pendek daripada jumlah panjang dua sisi lainnya. Pada graf berbobot, bobot suatu sisi harus lebih kecil daripada jumlah bobot dua sisi lainnya, di mana simpul yang bersisian dengan sisi pertama adalah simpul awal dan simpul akhir dari dua sisi yang membentuk lintasan (Karlin *et al.* [14]). Berikut adalah langkah-langkah umum dalam algoritma Christofides (Nallaperuma *et al.* [15]):

- Suatu graf $G = (V, E)$ yang merupakan *complete graph*.
- Membangun *minimum spanning tree*, T, dari graf G.
- Menentukan himpunan dari simpul O dalam graf G yang memiliki derajat ganjil.
- Membuat graf H yang adalah graf dengan simpul-simpul dari O dan memiliki sisi-sisi yang menghubungkan simpul-simpul dalam O.
- Mencari *minimum cost perfect matching* M, dari graf H.
- Mengabungkan graf T dan graf M menjadi G'.
- Mencari sirkuit Euler S dalam graf G'.
- Mengubah sirkuit Euler S menjadi sirkuit Hamiltonian π dengan menghapus sisi yang akan membuat simpul dikunjungi dua kali.

Minimum Cost Perfect Matching

Dari himpunan simpul-simpul dengan derajat ganjil tersebut, kita dapat membuat *minimum-cost perfect matching* yang merupakan subgraf dari graf tak berarah berbobot G yang memiliki semua simpul dari G, namun memiliki sisi-sisi sedemikian rupa sehingga tidak ada dua sisi dalam subgraf M yang terhubung dengan simpul yang sama, dan memiliki bobot minimum (Edmonds [16]; Kolmogorov [17]). Untuk mencari *minimum-cost perfect matching*, jumlah simpul dalam graf G haruslah berjumlah genap. Selanjutnya, graf dengan derajat ganjil akan digabungkan menjadi satu graf yang sama dan jika ada sisi ganda, sisi tersebut tidak akan digabung menjadi satu sisi tunggal, melainkan akan tetap dibiarkan untuk dieliminasi pada langkah terakhir.

Sirkuit Euler

Sirkuit Euler merupakan suatu lintasan yang melalui semua sisi-sisi yang ada dalam graf, dimana sirkuit Euler ini bertujuan untuk memastikan bahwa setiap sisi dalam graf termasuk dalam lintasan yang akan dibentuk (Kumar [18]).
Dalam langkah ini, graf hasil penggabungan mungkin menjadi graf ganda (*multigraph*), di mana beberapa sisinya dapat muncul lebih dari satu kali. Dengan menemukan sirkuit Euler, kita dapat memastikan bahwa setiap sisi dalam graf telah tercakup dalam lintasan yang dihasilkan. Namun, perlu diingat bahwa sirkuit Euler itu sendiri mungkin mengunjungi beberapa simpul lebih dari sekali, sehingga masalah ini kemudian akan diselesaikan pada langkah terakhir untuk melakukan eliminasi pada nodes yang dikunjungi lebih dari satu kali.

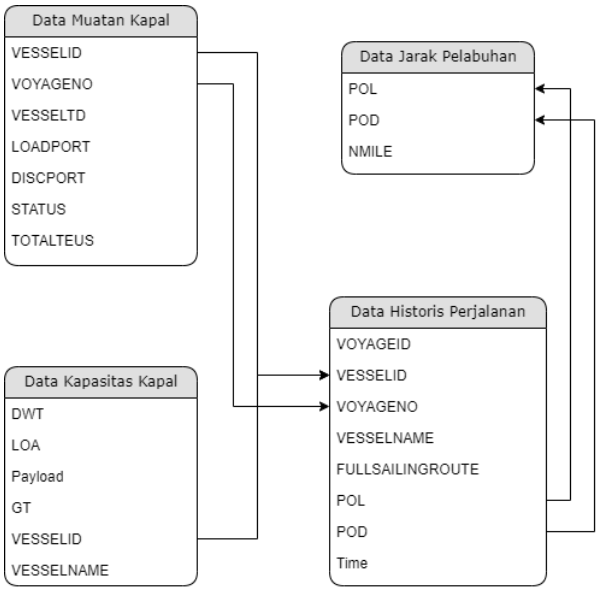
Lintasan Hamilton dan Sirkuit Hamilton

Langkah terakhir dalam algoritma Christofides melibatkan penghapusan dan penambahan sisi-sisi sehingga tidak ada simpul yang dikunjungi dua kali. Lintasan Hamilton dalam suatu graf melintasi semua simpul tepat sekali. Sirkuit Hamilton dalam suatu graf melintasi semua simpul, kecuali simpul awal yang juga merupakan simpul akhir (Rubin [19]; Babar *et al.* [20]).
Teorema Dirac menyatakan bahwa jika suatu graf sederhana memiliki jumlah simpul $n \geq 3$ dan setiap simpul memiliki derajat setidaknya $n / 2$, maka graf tersebut memiliki sirkuit Hamilton (Wolfram [21]). Dalam konteks ini, jika graf yang digunakan dalam algoritma Christofides adalah graf lengkap, di mana setiap simpul terhubung langsung dengan semua simpul lainnya, maka pasti terdapat sirkuit Hamilton. Hal ini karena setiap simpul dalam graf lengkap memiliki derajat n - 1, yang lebih besar dari $n / 2$. Oleh karena itu, penghapusan dan penambahan sisi-sisi dalam langkah terakhir bertujuan untuk memastikan bahwa lintasan yang dihasilkan merupakan sirkuit Hamilton, di mana setiap simpul dikunjungi tepat sekali tanpa ada simpul yang dikunjungi dua kali (Wang [22]).

Hasil dan Pembahasan

Tahap awal dalam penelitian ini adalah mengumpulkan data-data yang dapat digunakan untuk mendukung penelitian ini. Data relevan dan sudah dikumpulkan, meliputi data yang berkaitan dengan data *demand* atau muatan kapal, data jarak antar *port*, data kapasitas optimal kapal, dan data

historis perjalanan kapal. Setelah mengumpulkan data, dilakukan tahapan pengolahan data awal sebagai *input* dalam model yang akan digunakan pada penelitian ini. Untuk langkah awal, akan dilakukan penggabungan 4 *dataset* yang telah dikumpulkan, seperti Gambar 1.



Gambar 1. Data *relationship diagram*

Data muatan kapal dan data historis perjalanan kapal dihubungkan dengan *primary key* berupa VESSELID dan VOYAGENO, sedangkan data kapasitas kapal dihubungkan dengan variabel VESSELID, dan untuk data jarak antar pelabuhan dihubungkan dengan variabel POL dan POD, sehingga akan menghasilkan *dataset* baru, hasil dari penggabungan keempat *dataset* tersebut. Setelah dilakukan penggabungan *dataset*, dilakukan perhitungan untuk variabel LOAD dan UNLOADING, yang merupakan jumlah kontainer (*full* & *empty*) yang dimuat pada kapal di pelabuhan, dan jumlah kontainer (*full* & *empty*) yang dibongkar dari kapal di pelabuhan. Selanjutnya, menghitung *utility* kapal dengan membagi muatan dengan *payload*, dimana untuk *utility* ini dapat melebihi 100%, dikarenakan *payload* itu sendiri sebagai definisi nya, bukanlah kapasitas maksimal dari kapal, namun hanyalah kapasitas optimal yang diinginkan oleh perusahaan.

Proses evaluasi kemudian, dilakukan dengan mengambil data permintaan kapal, dan jarak antar pelabuhan, dan dilakukan perhitungan *weight*, ilustrasi proses ini dapat dilihat seperti pada Gambar 2. Kemudian, dilakukan pengambilan sampel pada beberapa rute perusahaan, yang merupakan rute TSP (kembali ke *depot*) dan non-TSP (tidak kembali ke *depot*), untuk melihat performa dari algorima ini bekerja dalam penentuan rute kapal.

	1	2	3	4	5	6	7	8	9	10
	POL	POD	DEMAND	JARAK	TOTAL DEMAND	TOTAL JARAK	P(B|A)	exp^(-JARAK)	Pd(B|A)	WEIGHT
i	C	Y	136	275	136	510	1.00	0.76	0.49	1.34228
ii	C	T	0	235	136	510	0.00	0.79	0.51	100
iii	Y	C	0	275	125	504	0.00	0.76	0.49	100
iv	Y	T	125	229	125	504	1.00	0.80	0.51	1.32319
v	T	C	38	235	43	464	0.88	0.79	0.50	1.44695
vi	T	Y	5	229	43	464	0.12	0.80	0.50	3.2374

Gambar 2. Ilustrasi perhitungan *weight*

Model yang dibangun akan memakai bantuan bahasa Python, dengan platform Google Colaboratory. Perlu diingat, bahwa *weight* yang di input pada model adalah weight untuk *directed graph*, dimana *weight* antar pasangan *nodes* adalah berbeda, maka dari itu akan dilakukan transformasi pada model, sehingga dapat diubah menjadi *undirected weight*, dimana weight r ke s dan s ke r adalah sama. Transformasi dilakukan dengan mengambil nilai *weight* minimum yang dimiliki oleh pasangan *nodes*, seperti pada Persamaan (9).

$$w(r|s) = \mathrm{w(s|r)} = \min(w(r|s), w(s|r)) \qquad (9)$$

Output akhir yang sudah didapatkan, kemudian akan dilakukan perbandingan terhadap 3 indikator, yang meliputi tingkat pemenuhan *demand* (dalam 1 putaran), jarak, dan utilitas kapal. Tabel 1 memperlihatkan perbandingan rute kapal XXX 12/2022 berdasarkan data historis dan *output* dari model yang sudah dijalankan. Dapat dilihat, bahwa ada 2 perbedaan rute yang nampak pada kedua rute tersebut, yaitu pada nodes RRR dan III.

Tabel 1. Perbandingan rute kapal XXX 12/2022

Rute	1	2	3	4	5	6
Output	YYY	SSS	MMM	III	BBB	YYY
Existing	YYY	SSS	MMM	RRR	BBB	YYY

Setelah mengetahui rute awal dan hasil dari model, dilakukan perhitungan untuk LOAD, UNLOAD, UTILITAS, serta JARAK dari kedua rute tersebut, contohnya pada Tabel 2. Perhitungan ini dilakukan untuk mendapatkan jumlah total jarak, rata-rata utilitas, dan persentase kegagalan dalam mengantarkan *demand* pada rute kapal XXX 12/2022. Selanjutnya, untuk menganalisa performa dan melakukan komparasi pada kedua rute awal dan rute yang dihasilkan adalah dengan melakukan penjumlahan jarak untuk keseluruhan rute, menghitung rata-rata untuk utilitas pada tiap pelayaran kapal, dan melihat apakah ada *demand* yang belum diambil atau diantarkan. Hasil dari komparasi ini dapat dilihat pada Tabel 3. Pada bagian *average utility* didapatkan dengan mencari nilai rata-rata utilitas, untuk mendapatkan perbandingan dari segi *utility* atau keterisian kapal saat berlayar.

Tabel 2. Contoh perhitungan performa rute

Existing	YYY	SSS	MMM	RRR	AAA	III	BBB	YYY
Payload	1122	1122	1122	1122	1122	1122	1122	1122
Load	598	99	283	112	106	101	209	0
Unload		258	200	94	85	60	0	811
Muatan		598	439	522	540	561	602	811
Utilitas		0,533	0,3913	0,4652	0,4813	0,5	0,5365	0,7228
Jarak		416	1051	165	140	113	601	986

Tabel 3. Ringkasan perbandingan performa rute

TSP (Algoritma Christofides)			
Rute Kapal XXX 12/2022 (7 *Nodes*)	*Average Utility*	Total Jarak	*Failed Demand*
Existing	51,86%	3472	0
Output	52,44%	3374	0
Perubahan	1,13%	-2,82%	0
Rute Kapal YYY 10/2022 (5 *Nodes*)	*Average Utility*	Total Jarak	*Failed Demand*
Existing	48,78%	3775	5
Output	62,04%	4154	100
Perubahan	27,18%	10,03%	95
Rute Kapal ZZZ 05/2022 (4 Nodes)	*Average Utility*	Total Jarak	*Failed Demand*
Existing	44,27%	1493	0
Output	54,59%	1668	36
Perubahan	23,31%	11,72%	36
NON-TSP (*Minimum Spanning Tree*)			
Rute Kapal XXX 12/2022 (7 *Nodes*)	*Average Utility*	Total Jarak	*Failed Demand*
Existing	51,86%	3472	0
Output	52,44%	3374	0
Perubahan	1,13%	-2,82%	0

Kemudian, untuk total jarak didapatkan dengan menjumlahkan seluruh jarak pada rute awal dan *output* model. Terakhir, untuk menghitung kegagalan pengangkutan dan pengantaran *demand*, dilakukan dengan mencocokan daftar *demand* dengan kedua rute. Dari hasil evaluasi lebih lanjut pada sampel, seperti pada Tabel 4, terlihat bahwa untuk rute yang dihasilkan dari model, performanya konsisten meningkat apabila hanya dilihat melalui parameter *utility*, namun terdapat penurunan terhadap total *distance* dan demand *fullfilment*, dengan adanya hasil ini, dapat dilihat bahwa model algoritma Christofides yang digunakan ini sudah dapat dengan konsisten (pada 3 sampel), meningkatkan performa *utility* kapal, namun untuk parameter dari total *distance* dan *demand fullfilment*, masih tidak konsisten dimana pada sampel pertama, mengalami perbaikan, sementara pada kedua sampel berikutnya, tidak berhasil dalam meningkatkan performanya. Hal ini dapat terjadi akibat asumsi dari nilai weight α = 0,5 antara *probability demand* dan *distance based*. Dikarenakan variasi rute non-TSP pada data historis yang minim, dan juga kebanyakan dari rute non-TSP yang ada, hanya memiliki *nodes* sejumlah 3 ataupun 4, sehingga untuk sampel hanya akan diambil pada 1 kapal YYY 13/2019 yang memiliki 5 *nodes*. Terlihat bahwa untuk MST memiliki performa yang sama dari segi *utility* dan *demand fullfilment*, namun mengalami penurunan dari performa *distance*.

Simpulan

Algoritma Christofides adalah algoritma aproksimasi berbasis *minimum spanning tree* untuk *travelling salesman problem* (TSP). Keunggulan algoritma ini adalah waktu penyelesaian yang lebih singkat dan rentang galat maksimum 50% dari solusi optimal *brute-force*. Algoritma ini cocok untuk perusahaan yang memiliki kapal dengan opsi kembali atau tidak kembali ke depot. *Minimum spanning tree* dapat menyelesaikan permasalahan kapal yang tidak kembali ke *depot*, sedangkan algoritma Christofides dapat menyelesaikan permasalahan TSP dengan kapal yang harus kembali ke *depot*. Kombinasi kedua algoritma ini dapat menjadi *baseline* penelitian untuk pemilihan rute dalam perusahaan. Meskipun

masih terdapat beberapa *constrain* yang tidak terjangkau seperti *capacitated vehicle*, *simultaneous pickup and delivery*, *time windows*, dan *heterogeneous vehicle*, model yang diciptakan dapat menjawab kebutuhan riset dan menjadi dasar untuk perbaikan sistem pemilihan rute di perusahaan. Pada kondisi perusahaan saat ini, pembuatan rute secara manual memiliki risiko menghasilkan rute yang tidak optimal. *Minimum spanning tree* dan algoritma Christofides dapat memberikan evaluasi pada rute-rute historis yang sudah dilalui oleh kapal. Model ini berhasil meningkatkan efisiensi kapal dalam beberapa sampel, namun performanya belum konsisten. Meskipun perbaikan yang sedikit dapat berdampak besar jika diakumulasikan pada frekuensi rute, perlu dilakukan penelitian lebih lanjut terkait pengaruh konfigurasi bobot pada performa model yang lebih konsisten. Model yang dibuat membuka peluang perbaikan dalam penentuan rute di perusahaan. Model saat ini hanya dapat menyelesaikan permasalahan TSP sederhana, namun memiliki *multi objective* dengan memperhatikan jarak dan *demand* berdasarkan probabilitas pada *weight* model. Meskipun model ini cocok untuk permintaan yang telah dikategorikan, seperti rute-rute pendulum atau *fixed route*, masih belum dapat menangani permasalahan *multiple* dan *capacitated vehicle*. Sehingga saran untuk penelitian kedepannya agar fokus menyelesaikan *generalized multiple depot multiple travelling salesman problem* serta penggunaan *neural network* untuk *multiple* TSP.

Daftar Pustaka

1. United Nation Division on Technology and Logistics, *Review of Maritime Transport 2022: Navigating Stormy Waters*, United Nation, 2022.
2. Salleh, N. H., Riahi, R., Yang, Z., and Wang, J., Predicting A Containership's Arrival Punctuality In Liner Operations By Using A Fuzzy Rule-Based Bayesian Network. *The Asian Journal of Shipping and Logistics*, 33(2), 2017, pp. 95–104.
3. Meng, Q., Wang, S., Andersson, H., and Thun, K., Containership Routing And Scheduling In Liner Shipping: Overview And Future Research Directions. *Transportation Science*, 48(2), 2004, pp. 265–280.
4. Christofides, N., Worst-Case Analysis Of A New Heuristic For The Travelling Salesman Problem. *Operations Research Forum*, 3(1), 1976.
5. Munir, R., *Buku Teks Ilmu Komputer: Matematika Diskrit*, Informatika, 2003.
6. Gohil, A., Tayal, M., Sahu, T., & Sawalpurkar, V., Travelling Salesman Problem: Parallel Implementations & Analysis. *arXiv*, 2022.
7. Ahmed, Z. H., Yousefikhoshbakht, M., Saudagar, A. K.., and Khan, S., Solving Travelling Salesman Problem Using An Ant Colony System Algorithm. *International Journal of Computer Science and Network Security*, 23(2), 2023, pp. 55-64.
8. Genova, K., and Williamson, D. P., An Experimental Evaluation Of The Best-Of-Many Christofides' Algorithm For The Traveling Salesman Problem. Algorithmica, 78(4), 2017, pp. 1109–1130.
9. Ahmed, Z. H., & Al-Dayel, I., An Exact Algorithm For The Single-Depot Multiple Travelling Salesman Problem. *International Journal of Computer Science and Network Security*, 20(9), 2020, pp. 65-74.
10. Kafaji, S. M. H. A., and Shiker, M. A. K., Optimizing The Minimum Spanning Tree (MST) And Its Relationship With The Minimum Cut. *International Journal of Health Sciences* (IJHS), 2020, pp. 9347–9360.
11. Hussein, A., and Klein, A., Modelling and Validation of District Heating Networks Using an Urban Simulation Platform. *Applied Thermal Engineering*, 2021, p. 187.
12. Mandi, J., Canoy, R., Bucarey, V., and Pardalos, P. M., Data Driven VRP: A Neural Network Model to Learn Hidden Preferences for VRP. *Principles and Practice of Constraint Programming*, 2021, p. 17.
13. Cococcioni, M., and Fiaschi, L., The Big-M Method With The Numerical Infinite M. *Optimization Letters*, 15(7), 2021, pp. 2455–2468.
14. Karlin, A. R., Klein, N. D., and Gharan, S. O., A (Slightly) Improved Approximation Algorithm For Metric Tsp, 2021.
15. Nallaperuma, S., Wagner, M., Neumann, F., Bischl, B., Mersmann, O., and Trautmann, H., A Feature-Based Comparison Of Local Search And The Christofides Algorithm For The Travelling Salesperson Problem, 2013.
16. Edmonds, J., Maximum Matching And A Polyhedron With 0,1-Vertices. *Journal of Research of the National Bureau of Standards*, Section B: Mathematical Sciences, 69B(1 and 2), 1965, p. 125.
17. Kolmogorov, V., Blossom V: A New Implementation Of A Minimum Cost Perfect Matching Algorithm. *Mathematical Programming Computation*, 1(1), 2009, pp. 43–67.
18. Kumar, A., A Study On Euler Graph And It's Applications. *International Journal of Mathematics Trends and Technology*, 43(1), 2017, pp. 9–15.
19. Rubin, F., A Search Procedure For Hamilton Paths And Circuits. *Journal of the ACM*, 21(4), 1974, pp. 576–580.
20. Babar, G. M., Khiyal, M. S. H., and Saeed, S. A., Finding Hamilton Circuit In A Graph. *In Conference on Scientific Computing*, 2006, pp. 238–244.
21. Wolfram Research, Inc. *Dirac's Theorem*. Wolfram MathWorld, n.d, retrieved from mathworld.wolfram.com/DiracsTheorem on 1 June 2023.
22. Wang, Y., A Nearest Neighbor Method with a Frequency Graph for Traveling Salesman Problem, 2014.