BAREKENG: Journal of Mathematics and Its Applications
December 2025 Volume 19 Issue 4 Page 2507–2520
P-ISSN: 1978-7227 E-ISSN: 2615-3017
https://doi.org/10.30598/barekengvol19no4pp2507-2520

OPTIMIZING BI-OBJECTIVE MULTIPLE TRAVELING SALESMEN ROUTES FOR DISASTER RELIEF LOGISTICS USING GENETIC ALGORITHM

Amos Hatoguan Sihombing[1*], **Ratna Herdiana**[2*], **Jovian Dian Pratama**[3]

[1,2,3]*Department of Mathematics, Faculty of Science and Mathematics, Universitas Diponegoro*
Jln. Prof. Jacub Rais, Semarang, 50275, Indonesia

*Corresponding author's e-mail: * **ratnaherdiana@lecturer.undip.ac.id***

Article History:

Received: 15th January 2025
Revised: 7th May 2025
Accepted: 10th June 2025
Available online: 1st September 2025

Keywords:

Disaster Relief Logistics;
Genetic Algorithm;
Multiple Traveling Salesman Problem;
Route Optimization.

ABSTRACT

Handling natural disasters such as floods requires efficient logistics distribution to minimize the negative impact on victims. Distribution route optimization becomes very important in this process. This paper applies a metaheuristic method using Genetic Algorithm to the Bi-objective Multiple Traveling Salesman Problem (BMTSP) to obtain a solution that minimizes the distance and time to deliver disaster relief logistics. Multiple vehicles are used in this study to represent delivery agents with two main objectives, namely minimizing total distance and travel time. Genetic Algorithm is applied by considering these two main objectives through the process of selection, crossover, mutation, and produces an effective Pareto solution. The results indicate that applying the Genetic Algorithm to the Bi-Objective Multiple Traveling Salesman Problem yields more efficient delivery routes—reducing both distance and time—compared to the Nearest Neighbor Algorithm. The simulation and testing in this study utilize data on distances and travel times among Central Java Regional Disaster Management Agency offices in 19 regencies—including a central depot—located in flood-prone areas of Central Java Province. The scenario involves two vehicles with identical load capacities.

How to cite this article:

A. H. Sihombing, R. Herdiana and J. D. Pratama., "OPTIMIZING BI-OBJECTIVE MULTIPLE TRAVELING SALESMEN ROUTES FOR DISASTER RELIEF LOGISTICS USING GENETIC ALGORITHM," *BAREKENG: J. Math. & App.*, vol. 19, no. 4, pp. 2507-2520, December, 2025.

1. INTRODUCTION

The Traveling Salesman Problem (TSP) is a prominent issue in combinatorial optimization, frequently addressed in mathematics and computer science. Its exploration began in the 18th century, with contributions from Irish mathematician Sir William Rowan Hamilton and English mathematician Thomas Penyngton Kirkman [1]. The aim of TSP is to determine the most efficient route that visits each node (city) exactly once and returns to the origin, factoring in the distance or cost between each pair of cities. TSP is recognized as an NP-hard problem, indicating that no universally optimal solution algorithm has been found for all instances [1]. As the number of cities increases, identifying the optimal solution to the TSP becomes progressively more complex due to its combinatorial nature. The computational effort grows exponentially with the problem size, making exact methods impractical for large-scale instances and necessitating the use of heuristic or metaheuristic approaches.

Over time, various extensions of TSP have emerged, each defined by specific characteristics. A proper understanding of these variants is essential for choosing suitable solution methods. The primary variations of the Traveling Salesman Problem include the symmetric TSP (sTSP), the asymmetric TSP (aTSP), and the Multiple TSP (MTSP). In sTSP, the distance between two cities is considered the same regardless of the travel direction, whereas in aTSP, the distances may vary depending on the direction traveled. MTSP involves multiple salesmen, each following their own route, with the goal of minimizing the total travel distance for all salesmen combined [1]. MTSP is often applicable to practical problems in logistics, supply chain management, e-commerce distribution, vehicle routing, and scheduling [2].

In real-world applications, MTSP has been adapted with additional constraints to better reflect actual conditions. One emerging problem is the Bi-objective Multiple Traveling Salesman Problem (BMTSP), which arises when there are two objectives to be optimized simultaneously. For example, in addition to minimizing the total travel distance, BMTSP may also consider minimizing travel time, total travel cost, or carbon emissions. Solving BMTSP typically involves multi-objective programming, with the goal of finding Pareto-optimal solutions, where no solution is better in all objectives [2] [3].

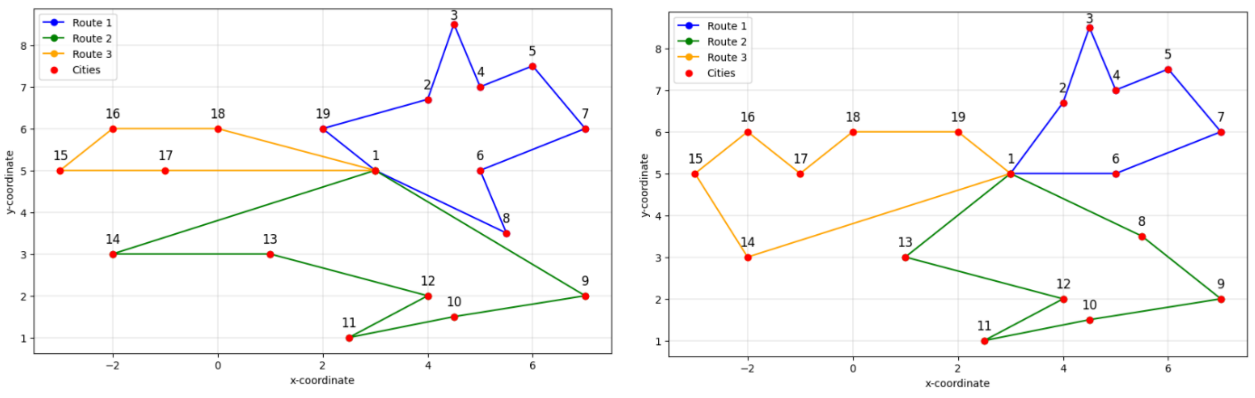
Due to the combinatorial complexity of BMTSP, studies on BMTSP remain limited making it challenging to find optimal solutions using exact algorithms. Consequently, considerable research efforts have been directed toward the development of heuristic and metaheuristic algorithms aimed at generating near-optimal solutions in a computationally efficient manner [4]. One such approach is the Genetic Algorithm (GA), which has been widely applied to various formulations of the Traveling Salesman Problem (TSP), examples found in [4] – [8]. Study by Ha et al. [6] introduced a hybrid GA approach incorporating drones to assist in package delivery, thereby addressing complex TSP scenarios. Additionally, a modified GA to solve MTSP, employing local search operators to explore the search space efficiently and identify optimal solutions was proposed in [9]. In another study [10], a macroscopic multi-criteria optimization model was developed to address large-scale evacuation planning, employing the NSGA-II variant of GA, which demonstrated notable effectiveness as a heuristic solution method. In [11], a hybrid GA was proposed for solving bi-objective TSP, incorporating two satisfactory degree indices to guide the evolution of individuals during iterations. Computational results show the algorithm is efficient and robust. Authors in [12] study a global perspective on the optimization of distribution on logistics and transportation network based on complex network theory, furthermore GA is used in solving the model and were relatively accurate and effective. In [13], a fuzzy-based GA approach was introduced to address uncertainties in evacuation demand, where crowd sizes at pickup points were modeled using triangular fuzzy numbers, aiming to minimize total travel time in the evacuation model.

Finding near-optimal solution is crucial in emergency logistics, as efficient resource distribution can significantly reduce casualties and damage during natural disaster. Emergency preparedness includes activities before, during, and after a disaster, with logistics distribution playing a key role. Here, vehicles or delivery agents represent the salesmen. The TSP method is useful for minimizing the total travel distance needed to reach all affected areas [9]. This paper focuses on solving a Bi-objective Multiple Traveling Salesman Problem model, aiming to optimize two main objectives: minimizing travel distance and minimizing travel time, in line with real-world constraints. GA is used as an optimization approach, with mutation techniques such as swap integrated into the mutation process to efficiently obtain Pareto solutions for MTSP with multiple objectives. The proposed algorithm is applied to simulated flood relief distribution data involving selected cities and districts in Central Java Province.

2. RESEARCH METHODS

2.1 Model Formulation

The Multiple Traveling Salesman Problem (MTSP) typically finds m optimal routes for m vehicles, which represent salesmen, where each vehicle starts and finishes their journey at the same depot city. Each city must be visited by exactly one vehicle, with no overlapping routes. Each vehicle starts and ends at the depot city, aiming to minimize the total distance traveled by all vehicles. However, optimizing the total distance traveled alone often results in an unbalanced distribution of cities. In some cases, a vehicle may have to visit most cities, while other vehicles only visit one or a few cities as in Figure 1 **(a)**. Whereas Figure 1 **(b)** show a more balanced solution through the application of the concept of load balancing, thus each vehicle gets almost the same number of cities.



(a) **(b)**

Figure 1. **(a) Route Plans of Unbalanced Visit Load, (b) Route Plans for MTSP with Balanced Load Distribution**.

This paper addresses MTSP with two objectives, which incorporates load balancing constraints. In the Bi-objective Multiple Travelling Salesman Problem (BMTSP), load balancing constraints are applied, with the first objective to minimize the total distance travelled and the second objective is to minimize the total travel time across all vehicles. In this BMTSP, each vehicle must visit maximum q cities, excluding the depot city, where $q = \lfloor (n-1)/m \rfloor$. Travel time between cities is not directly proportional to distance due to factors such as traffic conditions, road quality, and others. Achieving an optimal solution to this problem is quite challenging, as it involves trade-offs between two objectives. Therefore, finding the best Pareto solution is necessary for two-objective or multi-objective optimization. The Pareto optimal solution is defined as:

Definition 1. [14] *A solution $x^* \in X$ is said to be Pareto optimal if there is no other solution $x \in X$ that makes at least one objective function better without making the other objective function worse, which is expressed by the notation* $f(x) \prec pareto\, f(x^*)$.

The mathematical formulation of the BMTSP is presented as follows [2]:

$$\text{Minimize } Z = (z_1, z_2) \tag{1}$$

where,

$$\begin{cases} z_1 = \sum_{k=1}^{m} \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij}\delta_{ijk} \\ z_2 = \sum_{k=1}^{m} \sum_{i=1}^{n} \sum_{j=1}^{n} t_{ij}\delta_{ijk} \end{cases} \tag{2}$$

with constraints:

$$\sum_{k=1}^{m} \sum_{j=1}^{n} \delta_{\alpha jk} = m \tag{3}$$

$$\sum_{k=1}^{m} \sum_{j=1}^{n} \delta_{j\alpha k} = m \tag{4}$$

$$\sum_{k=1}^{m} \sum_{j=1}^{n} \delta_{ijk} = 1, \ \ \forall i \in V | \{\alpha\} \tag{5}$$

$$\sum_{k=1}^{m}\sum_{i=1}^{n}\delta_{ijk} = 1,\ \ \forall j \in V|\{\alpha\} \tag{6}$$

$$\sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{n}\delta_{ijk} = m+n-1 \tag{7}$$

$$\sum_{i\in S_k}\delta_{ipk} - \sum_{j\in S_k}\delta_{pjk} = 0,\ \ \forall p \in V,\ \ k = 1,2,\dots,m \tag{8}$$

$$\sum_{j=1}^{n} y_{ik} \le q,\ \ k = 1,2,\dots,m \tag{9}$$

$$\delta_{ijk}, y_{ik} \in 0,1 \tag{10}$$

The following is the list of notations used to explain the BMTSP model:

z_1	= The overall distance covered by the vehicles.
z_2	= The overall time taken by the vehicle.
n	= Number of cities, this parameter indicates that there are n cities that will be used in the problem.
m	= Number of vehicles, this parameter indicates the total number of vehicles that will be used to visit all cities in the problem.
α	= Depot city, this parameter indicates the start and end cities of each vehicle's journey.
V	= Vertex set, this set represents all the cities to be visited in the problem, including the depot city.
d_{ij}	= The distance between two cities i and j, this parameter indicates the distance between two specifics cities i and j in kilometers or miles.
t_{ij}	= Travel time between cities i and j, this parameter indicates the total time taken to travel between cities i and j in minutes or hours.
q	= Load balancer, this parameter indicates the maximum number of cities that must be visited by each vehicle, except the depot city.
S_k	= The subset of cities that vehicle k visits.
δ_{ijk}, y_{ik}	= Binary decision variables, variable δ_{ijk} indicates whether city j is visited from city i by vehicle k and variable y_{ik} indicates whether city i is visited by vehicle k.

Equation (2) defines the two objectives of minimizing total travel distance (z_1) and total travel time (z_2) across all vehicles. **Equation (3)** defines the starting condition (each vehicle departs from the depot). **Equation (4)** defines the ending condition (each vehicle returns to the depot). **Equation (5)** guarantees that every non-depot city ($i \in V|\{\alpha\}$) is visited exactly once by one vehicle. **Equation (6)** ensure thar each non-depot city ($j \in V|\{\alpha\}$) is entered exactly once, complementing **Equation (5)**. **Equation (7)** states that every feasible solution for BMTSP must have $m+n-1$ edges. **Equation (8)** ensures continuity of travel for each subset of cities S_k assigned to vehicles k, preventing the formation of sub-tours. **Equation (9)** addresses load balancing among vehicles, stating that the number of cities visited by a vehicle k should not exceed q. The two binary decision variables defined in **Equation (10)** are as follows:

$$\delta_{ijk} = \begin{cases} 1, & \text{if vehicle } k \text{ travels from city } i \text{ to city } j \\ 0, & \text{otherwise} \end{cases}$$

and

$$y_{ik} = \begin{cases} 1, & \text{if vehicle } k \text{ visits city } i \\ 0, & \text{otherwise.} \end{cases}$$

The variable δ_{ijk} helps in finding the path taken by each vehicle, ensuring that each city is only visited once by one vehicle and helps in calculating the total distance and travel time. The variable y_{ik} is utilized to determine the workload or the number of cities each vehicle is required to visit, ensuring that no vehicle visits more than q cities in addition to the depot city. Mutation steps, such as the swap operation, are incorporated into the mutation process to obtain efficient Pareto-optimal solutions for the MTSP, which involves the two objective functions defined in **Equation (1)**.

2.2 Solution Approach

The techniques of GA follow the genetic mechanisms of biological organisms that have adapted and evolved in a highly competitive and changing environment, hence the terms used in GA are widely adopted from these sciences [15]. The algorithm begins with a set of solutions known as a population. Solutions from this population are selected to create a new population, with the expectation that the new population will outperform the previous one. Solutions are chosen based on their fitness values to generate new solutions, and this process is repeated until specific conditions are fulfilled [16]. The key components of the algorithm include natural reproduction, selection, crossover, and mutation.

2.2.1 Population Initialization

Population initialization is a critical first step in GA, involving the creation of a set of valid solution candidates (individuals) that are randomly generated. Each individual is represented by chromosomes, meaning that the initial population consists of a collection of these chromosomes. These chromosomes must conform to the specific format selected for the problem, which could include a binary string, an integer array, or another data structure of a defined length [17]. The purpose of this population initialization is to ensure the genetic diversity in the population that is fundamental to the exploration of a wide solution space, and provide the algorithm with a good opportunity to correlate optimal solutions through the evolutionary process.

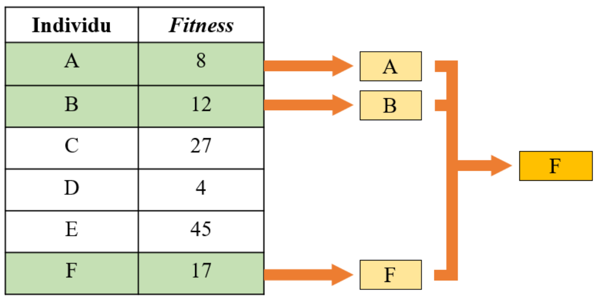
2.2.2 Fitness Evaluation

Evaluation is the process by which chromosomes in the population are judged to determine how well they solve the given problem. This stage is very important because the results of this evaluation will be used to guide the selection process in the GA, and lead to the formation of the next generation [16]. In maximization problems, the fitness value f is typically set equal to the objective function z, i.e., $f = z$. However, in minimization problems, this direct approach is unsuitable since GA favor individuals with higher fitness values. A common solution is to use $f = \frac{1}{z}$, allowing smaller objective values to correspond to higher fitness [18]. To prevent division by zero, a small positive constant θ is added to the denominator, yielding a stable fitness function:

$$fitness = f = \frac{1}{z + \theta} \tag{11}$$

2.2.3 Selection

Selection is the initial stage in each cycle of the GA process, where individuals (chromosomes) from the existing population are chosen to serve as parents for the next generation. This selection process is based on probability, with the likelihood of an individual being chosen linked to its fitness value, thereby favoring those with higher fitness levels. There are several selection methods in GA, one of which is tournament selection. In this method, two or more individuals are randomly selected from the population, and then the individual with the best fitness value will win and be selected to the next stage [17]. For example, given in Figure 2, where three individuals are randomly selected and individual F with the best fitness value wins.



Individu	*Fitness*
A	8
B	12
C	27
D	4
E	45
F	17

Figure 2. **Tournament Selection Process**

2.2.4 Crossover

Crossover is the main genetic operator in GA, which involves a process during the reproduction phase in which the genetic information of two parent individuals (chromosomes) is combined to create one or more offspring that may exhibit better qualities [16]. Various methods or techniques exist for crossover, allowing the selection of a problem-specific crossover method that best suits a particular situation. The choice of crossover method depends on the problem structure and desired exploitation [19].

2.2.5 Mutations

In the context of BMTSP, mutations play a crucial role in generating diverse solutions that can effectively balance the two objective functions, minimizing total travel distance and balancing the workload among vehicles. Mutation operators like swap, flip, and scramble are applied to the chromosome representations of candidate solutions, which encode the routes assigned to each vehicle. By introducing small, random changes to these routes, mutations help the algorithm escape local optima and explore a broader range of possible solutions. This is particularly important in BMTSP, where the search space is complex due to the presence of multiple vehicles and dual objectives. The use of mutation ensures that the population maintains sufficient diversity, increasing the likelihood of discovering Pareto-optimal solutions that offer a good trade-off between the competing objectives [20]. Figure 3 shows example of the three types of mutation process. Figure 3 (a) illustrates the process of flip mutation where the segment sequence of gene$_{[3],[4],[5]}$ i.e. | 4 7 2 | is flipped became |2 7 4|; Figure 3(b) is the process of swap mutation where gene$_{[3]}$ and gene$_{[6]}$ are swapped; and Figure 3 (c) illustrates the process of scramble mutation [2].

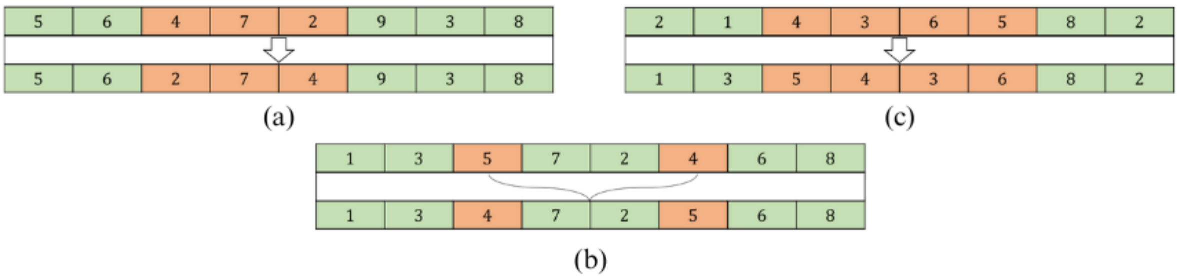
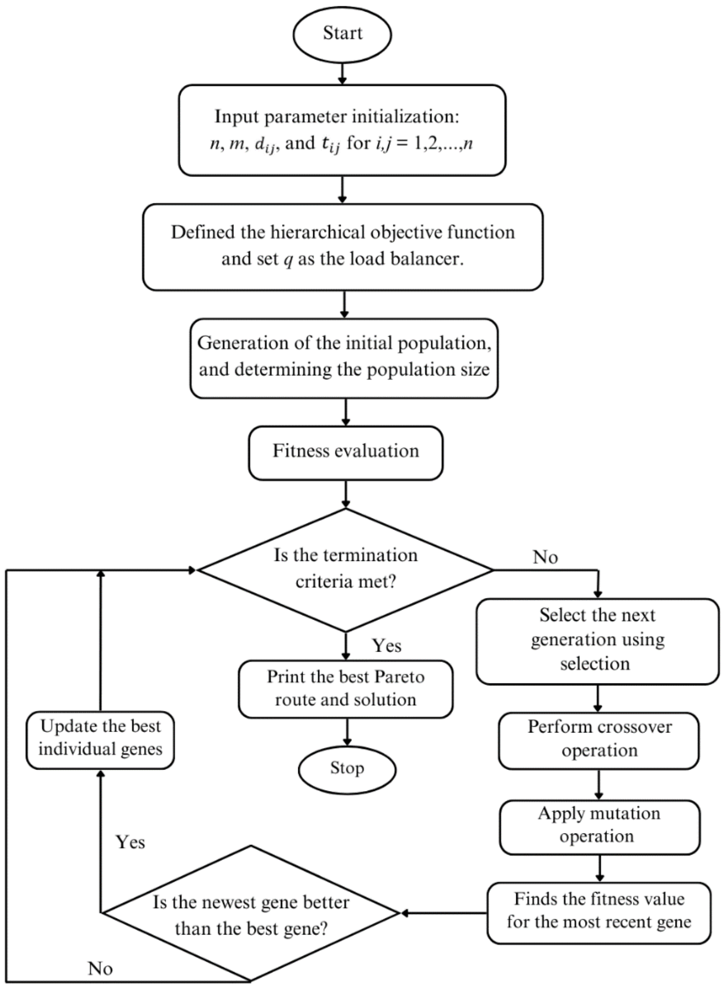


Figure 3. **(a) Flip Mutation, (b) Swap Mutation, (c) Scramble Mutation**



Start
Input parameter initialization: n, m, d_{ij}, and t_{ij} for $i,j = 1,2,\ldots,n$
Defined the hierarchical objective function and set q as the load balancer.
Generation of the initial population, and determining the population size
Fitness evaluation
Is the termination criteria met?
No
Yes
Select the next generation using selection
Print the best Pareto route and solution
Stop
Perform crossover operation
Apply mutation operation
Finds the fitness value for the most recent gene
Is the newest gene better than the best gene?
Yes
No
Update the best individual genes

Figure 4. **Flowchart of Genetic Algorithm on BMTSP**

2.2.6 Flowchart

To identify the Pareto optimal solution for BMTSP, the code structure of the proposed Genetic Algorithm, implemented in Python, is depicted in the flow chart in Figure 4.

3. RESULTS AND DISCUSSION

To better comprehend the BMTSP and the concepts involved in the proposed algorithm, the following implementation is given relating to the delivery of logistics assistance to areas categorized as flood-prone areas in several regencies in Central Java Province during the period January 1 – April 30, 2024. The logistics assistance is intended for $n = 19$ regencies, namely Semarang, Demak, Jepara, Kudus, Pati, Grobogan, Blora, Sragen, Karanganyar, Sukoharjo, Klaten, Boyolali, Kebumen, Cilacap, Brebes, Tegal, Pemalang, Pekalongan, and Kendal. Henceforth the regencies are represented respectively as nodes 1, 2, …, 19. The delivery route starts from node 1.

The Central Java Regional Disaster Management Agency (BPBD), which functions as the depot (aid storage warehouse) located in Semarang City. It is assumed that BPBD Central Java will deliver the logistics assistance to BPBD offices in the 18 regencies by land and utilize two vehicles, $m = 2$, with the same load capacity. The distance between two nodes i and j (d_{ij}) and time travel (t_{ij}) data were obtained based on coordinates taken from Google Maps. During the journey, the vehicles should not visit any other regencies other than those that have been allocated, except for the depot city as the starting and ending point. This problem includes two important aspects, the selection of which BPBD offices each vehicle should visit, and the order in which the vehicles should visit. Since this problem has two different objectives a single optimal solution cannot be obtained. Therefore, the Pareto solution approach is used, where the decision maker can select the most suitable solution based on the trade-off between distance and time, or based on the priority assigned to either objective.

Table 1. Distance Matrix (Kilometer)

d_{ij}	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	0	30.4	76.1	46.1	25.3	51.6	81.2	52.8	115.4	108.9	120.3	118.1	101.4	76.4	165.9	249.3	197.2	196.7	179.2
2	30.4	0	30.4	55.5	55.5	76.1	55.5	53.5	124.3	96.1	115.5	120.7	117.6	92.6	194	239.8	177.9	214.8	139.7
3	76.1	30.4	0	37.1	25.3	63	63	47	136.3	138.9	153.8	163.5	160.4	135.4	237.8	272.3	214.8	221.8	217.8
4	46.1	55.5	37.1	0	26.4	26.4	37	47	72.5	104.5	123.9	129.1	132	107	218.8	241.2	232.9	221.8	217.1
5	25.3	55.5	25.3	26.4	0	47	47	64.8	64.8	58.5	77.9	83.1	87.9	78.7	288.2	319.3	273.7	236.9	193
6	51.6	76.1	63	26.4	47	0	58.5	72.5	87	87	130	152	58.1	44.7	201.3	287.7	273.6	280.3	245.7
7	81.2	55.5	63	37	47	58.5	0	79	108.8	118.8	190.8	217	45.7	31.1	166.6	239	207.6	238	181.5
8	52.8	53.5	47	47	64.8	72.5	79	0	44.4	33.3	44.4	65.8	25.9	0	140.2	207.6	135.3	140.2	141.4
9	115.4	124.3	136.3	72.5	64.8	87	108.8	44.4	0	58.1	152	251.9	0	25.9	283.5	90.6	23.9	141.2	0
10	108.9	96.1	138.9	104.5	58.5	87	118.8	33.3	58.1	0	66.6	188.3	188.1	141.2	0	114.2	172.2	129.8	65.2
11	120.3	115.5	153.8	123.9	77.9	130	190.8	44.4	152	66.6	0	299	270.6	180.8	172.2	129.8	114.2	75	43.2
12	118.1	120.7	163.5	129.1	83.1	152	217	65.8	251.9	188.3	299	0	252.4	227.4	131.7	137.5	160.4	0	34.2
13	101.4	117.6	160.4	132	87.9	58.1	45.7	25.9	0	188.1	270.6	252.4	0	227.4	131.7	137.5	160.4	0	34.2
14	76.4	92.6	135.4	107	78.7	44.7	31.1	0	25.9	141.2	180.8	227.4	227.4	0	172.2	129.8	114.2	75	43.2
15	165.9	194	237.8	218.8	288.2	201.3	166.6	140.2	283.5	0	172.2	131.7	131.7	172.2	0	75	43.2	194.2	225.9
16	249.3	239.8	272.3	241.2	319.3	287.7	239	207.6	90.6	114.2	129.8	160.4	160.4	129.8	75	0	0	137.5	194.2
17	197.2	177.9	214.8	232.9	273.7	273.6	207.6	135.3	23.9	172.2	114.2	75	160.4	114.2	43.2	0	0	34.2	103.1
18	196.7	214.8	221.8	221.8	236.9	280.3	238	140.2	141.2	129.8	75	0	0	75	194.2	137.5	34.2	0	68.5
19	179.2	139.7	217.8	217.1	193	245.7	181.5	141.4	0	65.2	43.2	34.2	43.2	43.2	225.9	194.2	103.1	68.5	0

The mileage data used for logistics aid delivery route planning was obtained from the official source of BPBD Central Java. The data forms a symmetrical distance matrix, which means that the distance between two BPBD offices remains the same even if the direction of travel is reversed as presented in Table 1. The travel time for each delivery route is presented in Table 2. Unlike the distance matrix, the entries in Table 2 are asymmetric, meaning the travel time between two regencies can differ depending on the direction of travel. This variation can be caused by road conditions, congestion, or infrastructure differences between the outbound and return routes.

Table 2. Travel-Time Matrix (minutes)

t_{ij}	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	0	45	112	80	106	86	195	112	114	131	116	75	246	314	129	131	115	91	56
2	43	0	70	44	67	68	175	130	131	148	135	91	264	334	147	148	133	112	72
3	112	74	0	70	93	123	207	200	208	217	205	163	338	424	225	226	209	187	146
4	77	44	71	0	29	78	136	163	173	183	169	136	270	359	197	194	179	156	115
5	100	64	94	28	0	80	111	166	193	205	191	158	290	375	218	214	200	177	135
6	84	64	121	82	79	0	106	82	109	124	140	109	280	369	213	210	194	171	130
7	192	169	207	136	107	101	0	150	175	189	210	180	368	464	305	302	287	262	221
8	95	129	204	162	160	80	141	0	44	62	82	46	254	359	190	189	176	154	118
9	108	133	215	175	188	113	164	44	0	33	80	48	255	360	192	191	177	155	119
10	128	155	232	190	209	131	182	63	34	0	56	61	238	335	208	207	194	172	135
11	115	137	222	177	198	145	209	80	77	55	0	48	207	302	207	208	193	170	135
12	73	86	167	138	157	108	171	47	52	60	43	0	232	342	161	163	148	125	86
13	243	273	361	290	313	293	363	250	253	236	196	225	0	151	279	274	237	256	264
14	319	349	420	305	387	373	465	368	373	342	299	343	149	0	208	213	231	258	296
15	131	156	233	197	220	207	301	200	202	221	204	165	274	202	0	24	54	66	110
16	130	155	233	193	215	202	297	195	198	217	204	164	268	212	21	0	41	65	108
17	116	140	215	180	203	190	284	182	184	203	189	150	228	226	54	41	0	52	89
18	92	115	191	158	180	168	263	162	163	180	169	130	252	256	68	69	53	0	73
19	57	75	148	113	136	128	220	123	125	140	130	86	267	293	110	111	90	70	0

3.1 Setting Initial Population

In this study, the initial population for the genetic algorithm is set to consist of 10 chromosomes. Each chromosome represents a candidate solution that satisfies the problem constraints outlined in **Equation (3)** to **Equation (10)**. Specifically, each chromosome encodes two routes, represented by two consecutives square brackets, corresponding to the two vehicles involved in the BMTSP. For example, Chromosome$_{[1]}$ is represented as two sequences of cities visited by each vehicle, starting and ending at the depot city (node 1).

The choice of 10 chromosomes as the initial population size is grounded in a systematic approach based on the problem scale. This parameter is derived from the rounded value of the ratio $\frac{n}{m}$, where n is the total number of cities to be visited, and m is the number of vehicles available. This ratio provides an estimate of the average number of cities each vehicle is expected to serve, which in turn informs the population size to ensure sufficient diversity and coverage of the solution space. By selecting the initial population size in this manner, the algorithm balances computational efficiency with the need for genetic diversity. A population size that is too small may lead to premature convergence and suboptimal solutions, while an excessively large population increases computational overhead without proportional gains in solution quality. The chosen size of 10 chromosomes thus represents a practical compromise, enabling the genetic algorithm to explore a wide range of feasible solutions while maintaining manageable computational demands:

Chromosome$_{[1]}$ = [[1, 3, 2, 4, 6, 5, 7, 8, 10, 9, 1], [1, 12, 11, 13, 14, 16, 15, 17, 19, 18, 1]]
Chromosome$_{[2]}$ = [[1, 18, 19, 17, 16, 15, 14, 12, 13, 11, 1], [1, 9, 10, 8, 7, 6, 5, 4, 2, 3, 1]]
Chromosome$_{[3]}$ = [[1, 5, 7, 9, 11, 13, 15, 17, 19, 2, 1], [1, 4, 6, 8, 10, 12, 14, 16, 18, 3, 1]]
Chromosome$_{[4]}$ = [[1, 3, 6, 9, 12, 15, 18, 2, 5, 8, 1], [1, 11, 14, 17, 4, 7, 10, 13, 16, 19, 1]]
Chromosome$_{[5]}$ = [[1, 14, 12, 10, 8, 6, 4, 2, 19, 17, 1], [1, 15, 13, 11, 9, 7, 5 , 3, 18, 16, 1]]
Chromosome$_{[6]}$ = [[1, 18, 16, 14, 12, 10, 8, 6, 4, 2, 1], [1, 3. 5, 7, 9, 11, 13, 15, 17, 19, 1]]
Chromosome$_{[7]}$ = [[1, 15, 13, 11, 9, 7, 5, 3, 2, 4, 1], [1, 6, 8, 10, 12, 14, 16, 18, 17, 19, 1]]
Chromosome$_{[8]}$ = [[1, 17, 15, 13, 11, 9, 7, 5, 3, 2, 1], [1, 4, 6, 8, 10, 12, 14, 16, 18, 19, 1]]
Chromosome$_{[9]}$ = [[1, 2, 3, 4, 6, 8, 10, 12, 14, 16, 1], [1, 5, 7, 9, 11, 13, 15, 17, 19, 18, 1]]
Chromosome$_{[10]}$ = [[1, 19, 17, 15, 13, 11, 9, 7, 5, 2, 1], [1, 4, 6, 8, 10, 12, 14, 16, 18, 3, 1]]

3.2 Tournament Selection Process

For each chromosome, the value of the first objective function (z_1) which is the objective function of first priority is calculated to determine the fitness value using **Equation (11)**. The next step is the selection process based on the tournament selection method. The number of tournaments is according to the number of chromosomes, which is 10, In each tournament take 3 chromosomes randomly, then the best is selected based on the greatest fitness value. The ten tournaments obtained are given in **Table 3**.

Table 3. Selection Tournament Process Scenario

Tournament	Randomly Selected chromosomes	Fitness $[i]$	Best Chromosome	$Best[i_{win}]$
1	1, 2, 5	0.000728, 0.000612, 0.000452	1	0.000728
2	2, 3, 6	0.000612, 0.000555, 0.000599	2	0.000612
3	4, 5, 9	0.000422, 0.000452, 0.000554	9	0.000554
4	7, 8, 9	0.000575, 0.000599, 0.000554	8	0.000554
5	2, 4, 10	0.000612, 0.000422, 0.000571	2	0.000612
6	5, 6, 8	0.000452, 0.000599, 0.000599	6	0.000599
7	1, 3, 9	0.000728, 0.000555, 0.000554	1	0.000728
8	3, 6, 10	0.000555, 0.000599, 0.000571	6	0.000599
9	1, 2, 7	0.000728, 0.000612, 0.000575	1	0.000728
10	7, 8, 10	0.000575, 0.000599, 0.000571	8	0.000599

3.3 Crossover and Mutation

The next process is the crossover using the sequential crossover method and mutation process using the swap mutation method. Several combination tests between crossover probability and mutation probability are carried out and based on **Figure 5**, the highest fitness value achieved is 0.000876 obtained by combining $\rho_c = 0.8$ (probability of crossover is 80%) and $\rho_m = 0.05$ (probability mutation is 5%).

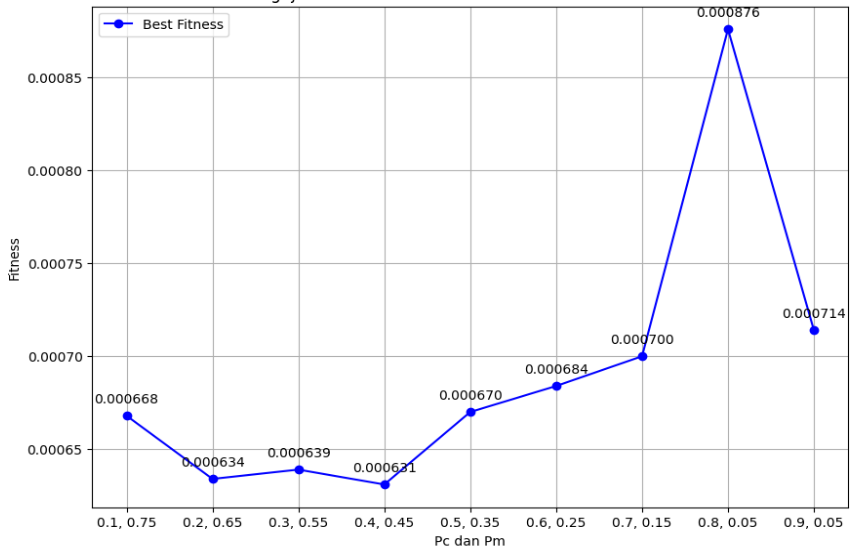


Figure 5. Results of Crossover and Mutation Probability Combination Testing

3.3.1 Sequential Crossover process

There are ten chromosomes in the population and the probability of crossover (ρ_c) is 0.8 or 80%. Before obtaining a pair of chromosomes, a random number $R_{[i]}$ between 0 and 1 is generated for

Chromosome$_{[i]}$. If the $R_{[i]}$ is less than ρ_c then the Chromosome$_{[i]}$ is chosen as the *parent.* Suppose random numbers $R_{[i]}$, i =1, …, 10 are obtained in order, namely (0.25, 0.55, 0.42, 0.17, 0.75, 0.66, 0.70, 0.33, 0.82, 0.90). Thus, the chromosomes that will undergo crossover are Chromosome$_{[1]}$, …, Chromosome$_{[8]}$. The scenario for determining chromosome pairs is determined randomly. Suppose the following pair is selected

Chromosome$_{[1]}$ >< Chromosome$_{[3]}$.

Chromosome$_{[1]}$ is called *Parent*$_{[1]}$ dan Chromosome$_{[3]}$ is called *Parent*$_{[2]}$. The location of the crossover point of a gene in the chromosome is done randomly. In this problem, the crossover point is symbolized by "|". For the above pair, supposed the crossover points is selected between the position of the 5th and 6th genes from Chromosome$_{[1]}$ and Chromosome$_{[3]}$. So the parental chromosome pairs ($P_{[1]}$, $P_{[2]}$) are as follows:

$P_{[1]}$ = [[1 3 2 4 6 | 5 7 8 10 9 1], [1 12 11 13 14 16 15 17 19 18 1]]

$P_{[2]}$ = [[1 2 3 4 6 | 8 10 12 14 16 1], [1 5 7 9 11 13 15 17 19 18 1]].

A new chromosome is called *Offspring*$_{[1]}$ or $O_{[1]}$ is formed by taking the first part before Gene$_{[6]}$ of *Parent*$_{[1]}$, and taking the second part of *Parent*$_{[2]}$. While the second new chromosomes called *Offspring*$_{[2]}$ or $O_{[2]}$ is formed by taking the first part before Gene$_{[6]}$ of *Parent*$_{[2]}$, and taking the second part of the *Parent*$_{[1]}$. So the new Chromosome$_{[1]}$ and Chromosome$_{[2]}$ are the 2 offsprings obtained from the *crossover* process:

$O_{[1]}$ = [[1 3 2 4 6 8 10 12 14 16 1], [1 5 7 9 11 13 15 17 19 18 1]]

$O_{[2]}$ = [[1 2 3 4 6 5 7 8 10 9 1], [1 12 11 13 14 16 15 17 19 18 1]].

If these chromosomes do not meet the **Equation (3)** to **Equation (10)**, then the new chromosomes (*offsprings*) cannot proceed to the next stage and the result of the crossover will be returned to the parent chromosome.

3.3.2 Swap Mutation Process

The total number of genes from 10 chromosomes is 220, the probability of mutation is 5%, which means 11 genes will undergo mutations. The sequencing of gene numbers starts from the first gene in Chromosome$_{[1]}$ to the last gene in Chromosome$_{[10]}$. The selection of genes that will undergo mutations is randomly selected, but still satisfy to the constraint conditions **Equation (3)** to **Equation (10)**. The randomly selected 11 genes out of 220 genes from the population of 10 chromosomes, and the corresponding chromosome that will undergo mutations as follows:

Chromosome$_{[1]}$: 2nd Gene and 20th Gene
Chromosome$_{[2]}$: 25th Gene
Chromosome$_{[3]}$: 53rd Gene
Chromosome$_{[4]}$: 68th Gene
Chromosome$_{[5]}$: 92nd Gene
Chromosome$_{[6]}$: 113rd Gene
Chromosome$_{[8]}$: 156th Gene and 168th Gene
Chromosome$_{[9]}$: 186th Gene and 189th Gene

The mutation process in Chromosome$_{[1]}$ occurs as follows: the 2nd Gene is swapped with the 3rd Gene; and the 20th Gene is swapped with the 21st Gene. Thus, a new gene arrangement in Chromosome$_{[1]}$ is obtained, namely: Chromosome$_{[1]}$ = [[1,2,3,4,6,8,10,12,14,16,1], [1,5,7,9,11,13,15,17,18,19,1]].

Similar mutation procedure is conducted for the other 7 chromosomes. The result from one iteration (generation) is a new generation of the following new population is shown in **Table 4**.

The GA process continue to repeat until the termination criteria is met. The criterion used is maximum number of generations, in this case it is set 1000 iterations The final resulting chromosome with the best fitness value of 0.000876 is:

Chromosome$_{z_1}$ = [[1,2,3,4,5,7,8,9,10,6,1],[1,12,11,13,14,16,15,17,19,18,1]].

Table 4. New Population after Renewal

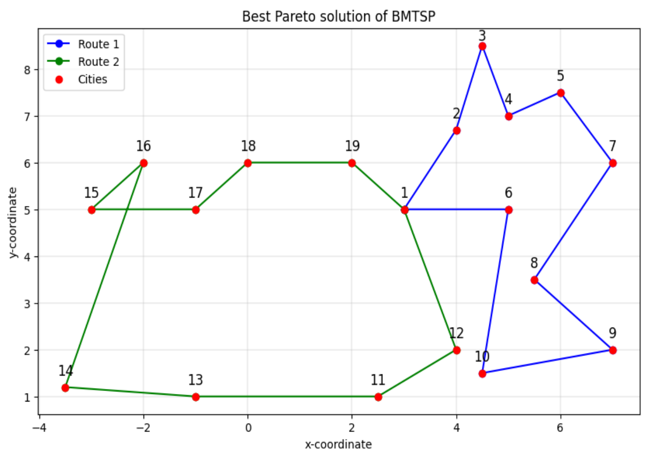
Chromosome$_{[i]}$	Genes	*Fitness*
1	[[1,2,3,4,6,8,10,12,14,16,1], [1,5,7,9,11,13,15,17,18,19,1]]	0.000599
2	[[1,19,18,17,16,15,14,12,13,11,1], [1,9,10,8,7,6,5,4,2,3,1]]	0.000668
3	[[1,2,3,4,6,5,7,8,9,10,1], [1,12,11,13,14,16,15,17,19,18,1]]	0.000754
4	[[1,15,17,13,11,9,7,5,3,2,1], [1,4,6,8,10,12,14,16,18,19,1]]	0.000603
5	[[1,17,19,18,16,15,14,12,13,11,1], [1,9,10,8,7,6,5,4,2,3,1]]	0.000587
6	[[1,3,2,4,6,5,7,8,10,9,1], [1,12,11,13,14,16,15,17,19,18,1]]	0.000728
7	[[1,3,2,4,6,5,7,8,10,9,1], [1,12,11,13,14,16,15,17,19,18,1]]	0.000728
8	[[1,18,19,17,16,15,14,12,13,11,1], [1,9,10,8,7,6,5,4,2,3,1]]	0.000618
9	[[1,3,2,4,6,5,7,8,9,10,1], [1,11,12,13,14,16,15,17,19,18,1]]	0.000719
10	[[1,17,15,13,11,9,7,5,3,2,1], [1,4,6,8,10,12,14,16,18,19,1]]	0.000599

The first objective function, z_1, is treated as the top priority, hence focusing on determining the *2*- route with the minimum total distance for the BMTSP while ensuring load balance.

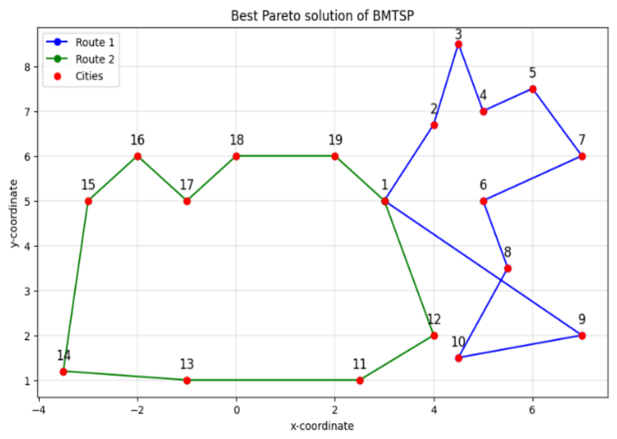
On the other hand, if the decision-maker chooses the objective function z_2 (total travel time) as the main priority, then the steps of the completion procedure are carried out similarly to those discussed for z_1, but using z_2 to calculate the fitness value. In this case, the end result from the GA simulation gives the best chromosome solution, with fitness value 0.0006086, is:

Chromosome_{z_2}= [[1,2,3,4,5,7,6,8,10,9,1], [1,12,11,13,14,15,16,17,18,19,1]].

The two route plans obtained for the BMTSP with n = 19, m = 2, and q = 9 of the two scenarios are given in Figure 6 (a) and Figure 6 (b), respectively, where the x and y coordinates indicate the location of cities in Central Java Province. The nodes are labeled with indices from 1 to 19. There are two route plans, one for each vehicle. Both vehicles start the journey from the depot city 1, visit 9 other BPBD offices separately, and return to the depot. If z_1 is given top priority, the optimum total distance traveled is 1141.08 kilometers with a travel time of 1713 minutes. Conversely, if z_2 is given top priority, the total distance traveled is 1173.85 kilometers with minimum travel time of 1643 minutes. The optimal solution cannot be achieved for both objectives simultaneously, so the final decision depends on the priority set by the decision maker.



(a)



(b)

Figure 6. (a) Solution with z_1 as First Priority, (b) Solution with z_2 as First Priority

The first objective function z_1 is considered as the top priority based on the fact that route determination with minimum distance is easier to verify and less influenced by external factors, such as traffic conditions, compared to travel time. Therefore, prioritizing z_1 provides a clear and reliable basis for determining logistics delivery routes, especially in emergency situations such as aid delivery to flood-prone areas.

We next implemented the Nearest Neighbor Algorithm (NNA) [21] to evaluate and compare the results regarding distance and travel time in logistics support delivery. NNA serves as a benchmark for addressing the same problem, allowing identification of performance differences between the two algorithms. The comparison of results generated by the two algorithms presented in **Table 5** depicts the effectiveness of the two approaches in determining the optimal route for logistics aid delivery in Central Java Province. It can be seen that GA is superior in minimizing distance and time, as well as providing a more efficient solution in the context of logistics delivery to flood-prone areas.

Table 5. Comparison of Optimal Routes from Two Algorithms

Methods	The Two Optimal Route	Total Distance of two routes (Kilometers)	Total Time (Minutes)
Genetic Algorithm	Vehicle 1: 1→2→3→4→5→7→8→9→10→6→1 Vehicle 2: 1→12→11→13→14→16→15→17→18→19→1	1141.08	1713
Nearest Neighbor Algorithm	Vehicle 1: 1→2→4→5→6→8→9→10→11→12→1 Vehicle 2: 1→19→18→17→16→15→14→13→3→7→1	1281.40	1886

4. CONCLUSION

The Bi-objective Multiple Traveling Salesman Problem (BMTSP) is a special variant of the Traveling Salesman Problem (TSP) that involves multiple salesmen with more than one objective being optimized simultaneously. This study presents a novel application of the BMTSP to the context of disaster relief logistics, specifically targeting the efficient delivery of aid to flood-prone areas in Central Java. The primary objective is to simultaneously minimize total delivery distance and travel time. A Pareto-based multi-objective optimization approach is employed to capture the trade-off between these conflicting goals. As shown in the experimental results, the GA approach produced a solution with a minimum total distance of 1141.08 km and a travel time of 1713 minutes, demonstrating its effectiveness of 10.95% and 9.17%, respectively over the Nearest Neighbor Algorithm (NNA) method. These findings underscore the potential of GA-based optimization in enhancing the responsiveness and effectiveness of logistics planning in disaster-prone regions.

AUTHOR CONTRIBUTIONS

Amos Hatoguan Sihombing: Conceptualization, Formal Analysis, Investigation, Methodology, Software, Visualization, Writing-Original Draft. Ratna Herdiana: Conceptualization, Funding Acquisition, Project Administration, Resources, Supervision, Visualization, Writing-original draft, Writing- Review and Editing. Jovian Dian Pratama: Supervision, Validation, Visualization, Writing-original draft, Writing- Review and Editing. All authors discussed the results and contributed to the final manuscript.

FUNDING STATEMENT

This work is financially supported by research grant of Dana Selain APBN Faculty of Science and Mathematics, Universitas Diponegoro 2025, grant number: 26.G/UN7.F8/PP/II/2025.

ACKNOWLEDGMENT

The authors would like to express their gratitude to the Faculty of Science and Mathematics, Universitas Diponegoro for their assistance. The authors also thank the editors and reviewers for their valuable suggestions and comments.

CONFLICT OF INTEREST

The authors declare no conflicts of interest to report study.

REFERENCES

[1] R. Matai, S. Singh, and M. Lal, "TRAVELING SALESMAN PROBLEM: AN OVERVIEW OF APPLICATIONS, FORMULATIONS, AND SOLUTION APPROACHES," in *Traveling Salesman Problem, Theory and Applications*, InTech, 2010. doi: https://doi.org/10.5772/12909.

[2] S. Linganathan and P. Singamsetty, "GENETIC ALGORITHM TO THE BI-OBJECTIVE MULTIPLE TRAVELLING SALESMAN PROBLEM," *Alexandria Engineering Journal*, vol. 90, pp. 98–111, Mar. 2024, doi: https://doi.org/10.1016/j.aej.2024.01.048.

[3] X. Zan, Z. Wu, C. Guo, and Z. Yu, "A PARETO-BASED GENETIC ALGORITHM FOR MULTI-OBJECTIVE SCHEDULING OF AUTOMATED MANUFACTURING SYSTEMS," *Advances in Mechanical Engineering*, vol. 12, no. 1, Jan. 2020, doi: https://doi.org/10.1177/1687814019885294.

[4] S. Geetha, P. T. Vanathi, and G. Poonthalir, "METAHEURISTIC APPROACH FOR THE MULTI-DEPOT VEHICLE ROUTING PROBLEM," *Applied Artificial Intelligence*, vol. 26, no. 9, pp. 878–901, Oct. 2012, doi: https://doi.org/10.1080/08839514.2012.727344.

[5] F. Mone and J. E. Simarmata, "APLIKASI ALGORITMA GENETIKA DALAM PENJADWALAN MATA KULIAH," *BAREKENG: Jurnal Ilmu Matematika dan Terapan*, vol. 15, no. 4, pp. 615–628, Dec. 2021, doi: https://doi.org/10.30598/barekengvol15iss4pp615-628.

[6] Q. M. Ha, Y. Deville, Q. D. Pham, and M. H. Hà, "A HYBRID GENETIC ALGORITHM FOR THE TRAVELING SALESMAN PROBLEM WITH DRONE," *Journal of Heuristics*, vol. 26, no. 2, pp. 219–247, Apr. 2020, doi: https://doi.org/10.1007/s10732-019-09431-y.

[7] J. Xu, L. Pei, and R. Zhu, "APPLICATION OF A GENETIC ALGORITHM WITH RANDOM CROSSOVER AND DYNAMIC MUTATION ON THE TRAVELLING SALESMAN PROBLEM," *Procedia Comput Sci*, vol. 131, pp. 937–945, 2018, doi: https://doi.org/10.1016/j.procs.2018.04.230.

[8] F. Rafique and H. Cui, "AN EFFICIENT GENETIC ALGORITHM FOR SOLVING THE TRAVEL SALESMAN PROBLEM," Dec. 14, 2023. doi: https://doi.org/10.21203/rs.3.rs-3739487/v1.

[9] R. I. Bolaños, E. M. Toro O, and M. Granada E, "A POPULATION-BASED ALGORITHM FOR THE MULTI TRAVELLING SALESMAN PROBLEM," *International Journal of Industrial Engineering Computations*, pp. 245–256, 2016, doi: https://doi.org/10.5267/j.ijiec.2015.10.005.

[10] M. Goerigk, K. Deghdak, and P. Heßler, "A COMPREHENSIVE EVACUATION PLANNING MODEL AND GENETIC SOLUTION ALGORITHM," *Transp Res E Logist Transp Rev*, vol. 71, pp. 82–97, Nov. 2014, doi: https://doi.org/10.1016/j.tre.2014.08.007

[11] M. Ma and H. Li, "A HYBRID GENETIC ALGORITHM FOR SOLVING BI-OBJECTIVE TRAVELING SALESMAN PROBLEMS," *J Phys Conf Ser*, vol. 887, p. 012065, Aug. 2017, doi: https://doi.org/10.1088/1742-6596/887/1/012065.

[12] H. Liu, P. Zhan, and M. Zhou, "OPTIMIZATION OF A LOGISTICS TRANSPORTATION NETWORK BASED ON A GENETIC ALGORITHM," *Mobile Information Systems*, vol. 2022, pp. 1–8, Jun. 2022, doi: https://doi.org/10.1155/2022/1271488.

[13] D. Zhang and W. Y. Zhang, "AN OPTIMIZATION DESIGN FOR EVACUATION PLANNING BASED ON FUZZY CREDIBILITY THEORY AND GENETIC ALGORITHM," *IOP Conf Ser Earth Environ Sci*, vol. 81, p. 012194, Aug. 2017, doi: https://doi.org/10.1088/1755-1315/81/1/012194.

[14] A. Roy, G. So, and Y. A. Ma, "OPTIMIZATION ON PARETO SETS: ON A THEORY OF MULTI-OBJECTIVE OPTIMIZATION," 2023, Accessed: Jun. 03, 2025. [Online]. Available: https://arxiv.org/abs/2308.02145

[15] S. D. Immanuel and U. Kr. Chakraborty, "GENETIC ALGORITHM: AN APPROACH ON OPTIMIZATION," in *2019 International Conference on Communication and Electronics Systems (ICCES)*, IEEE, Jul. 2019, pp. 701–708. doi: https://doi.org/10.1109/ICCES45898.2019.9002372.

[16] S.N. Sivanandar and S.N. Deepa, *INTRODUCTION TO GENETIC ALGORITHMS*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. doi: 10.1007/978-3-540-73190-0.

[17] Eyal Wirsansky, *HANDS-ON GENETIC ALGORITHMS WITH PYTHON*. Birmingham: Packt Publishing Ltd, 2020.

[18] A. P. Wibowo, D. Avianto, and I. Imantoko, "PENGEMBANGAN ALGORITMA GENETIKA DENGAN PENDEKATAN REPETITIVE RANDOM UNTUK PENJADWALAN UJIAN PENDADARAN PROYEK TUGAS AKHIR," *Jurnal Nasional Teknologi dan Sistem Informasi*, vol. 7, no. 1, pp. 35–43, Jun. 2021, doi: https://doi.org/10.25077/TEKNOSI.v7i1.2021.35-43.

[19] A. Saxena and J. Panday, "REVIEW OF CROSSOVER TECHNIQUES FOR GENETIC ALGORITHMS," *International Journal of Trend in Research and Development,* vol. 3, no. 5, 2019.

[20] A. Hassanat, K. Almohammadi, E. Alkafaween, E. Abunawas, A. Hammouri, and V. B. S. Prasath, "CHOOSING MUTATION AND CROSSOVER RATIOS FOR GENETIC ALGORITHMS—A REVIEW WITH A NEW DYNAMIC APPROACH," *Information*, vol. 10, no. 12, p. 390, Dec. 2019, doi: https://doi.org/10.3390/info10120390.

[21] R. Ferdiani Harahap and Sawaluddin, "STUDY VEHICLE ROUTING PROBLEM USING NEAREST NEIGHBOR ALGORITHM," *J Phys Conf Ser*, vol. 2421, no. 1, p. 012027, Jan. 2023, doi: https://doi.org/10.1088/1742-6596/2421/1/012027.