Received: September 4, 2022. Revised: December 4, 2022.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Hybrid Metaheuristic for Solving Maritime Inventory Routing Problem in Bulk Product Transportation

Antono Adhi[1,2]* **Nurhadi Siswanto**[1]* **Budi Santosa**[1]

[1] *Industrial and Systems Engineering Department, Institut Teknologi Sepuluh Nopember, Indonesia*
[2] *Industrial Engineering, Universitas Stikubank, Indonesia*
* Corresponding author's Email: siswanto@ie.its.ac.id

Abstract: Maritime Inventory Routing Problem (MIRP) is an important issue in the optimization of maritime distribution and transportation. This problem is related to planned ship's routing and scheduling in delivery of goods from the depot to some demand points by minimizing associated costs such as transportation and inventory management costs. This paper discusses how to solve MIRP with multi ships and limited undedicated compartments used to deliver bulk products from the depot port to the several consumption ports. The new hybrid metaheuristics are used to find the optimal assignment routes and schedule of ships along time horizon. This research modifies several metaheuristics algorithms called Modified Hybrid Particle Swarm Optimization (MHPSO) to find the best solution for MIRP. The algorithm of this method is developed from the combination of Particle Swarm Optimization (PSO), Nahwaz-Enscore-Ham (NEH), and 3-Opt. Some metaheuristic methods such as Genetic Algorithm (GA), Tabu Search (TS), Particle Swarm Optimization (PSO), Hybrid Genetic Algorithm (HGA), and Hybrid Tabu Search (HTS) were also developed in the same way to test and compare with the proposed method. Based on the ten test data instances, it can be concluded that MHPSO provides 0.64% effectiveness better results than other metaheuristic methods.

Keywords: Maritime inventory routing problem, Transportation, Hybrid metaheuristics, Particle swarm optimization, Genetic algorithm, Tabu search.

1. Introduction

In supply chain management, all parties in the chain manage to achieve profitable value. It challenges management to deliver superior customer value at less cost to the whole supply chain [1]. Cost is the most important factor in decision-making [2]. Logistic management becomes part of supply chain management when it focuses to plan the flow of products and information through a business [1]. The process of logistics strategically manages the procurement, movement, and storage of products and their parts in such a way that profitability is maximized by cost efficiency to fulfill orders. The efficiency objective becomes important since it provides a competitive advantage. In other words, a company that cannot obtain efficiency in any of its business activities will not survive in competition with other companies. The company cannot push down the cost of business activity and cannot produce low price products. It is hard for the company to compete with other companies. Therefore, decision-making and optimization in vehicle routing to distribute and allocate transportation are very important [3].

Transportation and inventory management are the most important issues in supply chain management [4]. The integration of these two issues in the efficiency achievement process provide significant savings instead of treating them separately [4]. Product distribution strategies in the transportation process should consider the status of every inventory edge. These edges can be depots or customers to whom the products send to. The strategy considers any costs that appear along with the distribution. This problem is called the Inventory Routing Problem (IRP). IRP focuses mainly on the logical aim of cost reduction [5]. Cost calculation involved any activities both in transportation and inventory management. These costs have relation to each other. Transportation cost is the main cost in distribution activity. It has relation to fuel use. Therefore, any strategies should consider optimization of this cost, such as achieving a short distance. The main cost in inventory management is stock-out cost. The objective of IRP is to minimize all costs that appear in a route of delivery.

IRP usually appears in a routine delivery type, such as the delivery of bulk products such as oil or cement. The most used transportation mode to deliver bulk products from suppliers to the customers is through sea by ships. This IRP is called Maritime Inventory Routing Problem (MIRP). According to the IRP, MIRP has a problem planning ships' routing and scheduling to optimize costs in maritime transportation and inventory management [6]. Most suppliers and customers in a routine MIRP are in the same management or company. Supplier is depot port, and the customers are distributors of the company. In this paper, distributors were called consumption ports. In this type of organization, the depot has full information control on the inventory status of consumption ports. In supply chain management, this condition can be implemented in the different companies between depot and consumption ports if the depot have full control to obtain information on the consumption port needs. The number of products delivered in the distribution activity does not depend on the demand of consumption ports. Depot ports check the status of product inventory and determine which kind of product and how many products are brought.

MIRP has special conditions compared with other kinds of transportation. The transportation in MIRP uses a huge capacity ship to deliver bulk products [7], although it can be implemented in a small capacity ship. Usually, ships have one or more compartments to bring the products. There are two types of compartments, namely dedicated and undedicated compartments. A dedicated compartment cannot be used to bring different product types. It only can be used to bring one product type, while an undedicated compartment can be used to bring different product types in other traveling. The number of products types is usually more than the number of compartments, that is why an undedicated compartment is proper to bring bulk products. A compartment can be filled with a different type of product after it has been emptied [8]. The calculation process to plan the route of a ship is more complex if a ship with dedicated compartments is used to bring the bulk product. It is not efficient since one or some compartments are set to empty because the product type needed by a distributor is not proper with the compartment.

The main planning of the route of a ship is processed when the ship arrives in a depot. The ship is assigned to the next route of delivery. In the assignment management, the duty of an officer is searching for the optimum route by selecting distributors and products that are brought. This selection needs double optimum searching since the process against a problem of searching optimum route and what kinds of products that are brought. Not all products demanded by the consumption port can be brought by a ship, since the number of product types is more than the number of compartments. The objective of MRIP is to obtain effective minimum costs. Therefore, every effort to search optimum result must consider the minimum cost. The problem of the calculation for every routing case is an NP-Hard problem. The consequence of the problem is that large-scale MIRP cannot be solved by exact methods to obtain optimum solution [9]. It is hard for exact methods to solve complex problem such as MIRP and must calculate double optimum searching in feasible time [10]. The use of exact method only provides limited size instances of MIRP and cannot solve large-scale cases efficiently [11]. Metaheuristic methods are able to solve MIRP that are difficult to be solved by exact method [12]. This method provides a feasible and high-quality solution.

Metaheuristic methods do not guarantee optimal solutions but they are able to solve in reasonable computing time. Some research uses this method to solve MIRP. Some metaheuristic methods are used to solve MIRP with time window constraints [13]. The metaheuristic method is Particle Swarm Optimization (PSO) with Composite Particle (PSO-CP) to solve the problem and a combination of other metaheuristics to validate the result. A hybrid metaheuristic was proposed to solve multiple time windows MIRP [8]. This is multi-heuristics based on Genetic Algorithm (GA) as a strategy to select ship, route, product type, and quantity of products. This method is concluded as flexible to implement for other variants of MIRP. Tabu Search (TS) method is designed to solve pickup and delivery vehicle routing problem for multi-visit that closely relate to MIRP [14]. This method can produce optimal or near optimal solution for the proposed problem in a short time.

PSO works as a representation movement of organisms in a bird flock or fish school. The particles in PSO move to the combination of best particle and the best position of it. GA is inspired by the process of natural selection such as mutation, crossover, and selection. Tabu Search is local search method. It searches potential solution and check its immediate neighbors. These metaheuristic methods have the same initialization process. All of the data is generalized randomly in initialization process. Hybrid technique that is implemented in PSO (MHPSO) also can be implemented in GA and TS. According to this condition, GA and TS are also treated with the same hybrid process as PSO to compare the result.

The objective of this paper is to solve MIRP with multi ships and limited undedicated compartments used to deliver bulk products from the depot port to the consumption port. The assignment of every ship is obtained when the ship arrived at the depot port. Ships arrive at the depot port one by one at different times. The process is executed within a specific time horizon. Every planning and assignment schedule is processed with the metaheuristics method to obtain optimum transportation and inventory management costs. Referring to previous research that had succeeded in solving problems using the metaheuristic method, it was used to calculate optimum route of the ships and product types brought by the ships in this paper. The metaheuristic method uses hybrid Particle Swarm Optimization (HPSO) to solve Vehicle Routing Problem (VRP) in bulk product transportation [15]. This method is modified (Modified HPSO – MHPSO) to obtain better and proper solution to solve MIRP. The other metaheuristic methods namely GA and Tabu Search (TS) are used to compare with the result of MHPSO. The two methods are also treated in the same way as MHPSO to become Hybrid GA (HGA) and Hybrid Tabu Search (HTS).

The rest of the paper is organized as follows. Section 2 describes problem description. Section 3 presents a proposed method to solve MIRP. Section 4 describes the experiments and results. The last section concludes the results of the paper.

2. Problem description

This paper studies IRP in the delivery of bulk product by ships. This problem is called MIRP. MIRP is a complex problem. This problem is observed in a predetermined time horizon. Along that time horizon, ships take turns delivering bulk products from depot to the port cities. Each ship is assigned after the ship docked at the depot. The main objective of the assignment is to deliver bulk products at the lowest cost. This activity requires very careful calculations. Otherwise, the ship assignment will cost a lot of money.

Fig. 1 shows routes of the ships. They deliver products from depot to consumption ports. S_iR_j denotes ship i travels in route j. This ship serves port P (1, 2, ..., n). $n \leq MaxPort$. $MaxPort$ is a maximum number of ports can be served by a ship. The condition $n < MaxPort$ is happened when the number of products brought by the ship cannot fulfill all the needs of consumption port demands. Otherwise, when $n = MaxPort$, the ship should wait until the rest of the products are unloaded in the last port.

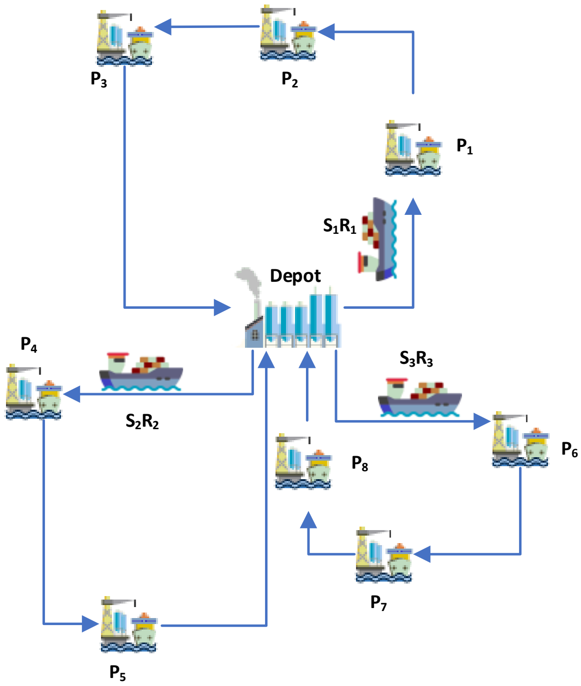

Figure. 1 Routes of the ships

MIRP activity process is repeated every time a ship returns to the depot. It is assigned to the next task to deliver products to other ports. In this assignment, the costs of delivery include travel cost, stockout cost, and demurrage cost. The assignment should choose the better route that is calculated by hybrid metaheuristic method. Therefore, this route provides optimum cost. This process is terminated after the processing time has reached the time horizon.

Limitation of factors in real condition needs precise calculation to obtain optimum delivery. The conditions considered in this problem are defined in mathematical model as follows:

- The number of ship compartments is less or equal than product types. This causes a ship may not be able to serve all consumption port demands in one shipment. This condition can be defined as follows:

$$cos \leq poc \tag{1}$$

cos is compartment number of ship s and poc is product type number of demands from consumption port c.

- The ship's compartment in one carriage must be filled fully to maintain transport efficiency. In undedicated compartment, a product can be put in any compartments. Capacity of ship's compartment is huge. The number of products in a compartment can be more than the number of same products on the consumption ports' demands. This condition can be defined as follows:

$$cap_{co,p} \geq \sum_{c=1}^{n} dmc_{c,p} \tag{2}$$

$cap_{co,p}$ is the capacity of compartment *co* and $dmc_{,p}$ is total demands of product *p* from *n* consumption ports served by the ship. This condition causes the rest of the product stored in one compartment to be sent to other consumption port s until empty. Ships deliver products to more than one consumption ports until all the products in their compartments are emptied. However, the number of ships visit to consumption ports is limited to a certain number. If there is still product left in the compartment, the ship wait at the last consumption port until the rest of the product can be moved. This cause demurrage cost that can be defined as follows:

$$DMc = tw \times dmt \tag{3}$$

- This delivery uses multi ships to service demand of port cities. Ships move from island to islands. Travel cost becomes an important issue that has to be considered. Most of the costs are due to fuel costs. This cost can be defined as follows:

$$TRc = dis_{i,j} \times trm \tag{4}$$

TRc is travel cost. $dis_{i,j}$ is distance between port *i* and *j* in nautical mile. *trm* is travel cost per mile.

- Stock out of product in every consumption port is not allowed. A penalty cost is given when there is a stock out of products. This cost is defined as follows:

$$SOc = (T_i - T_{i\text{-}1}) \times sot \tag{5}$$

SOc is stockout cost. $T_i - T_{i\text{-}1}$ is time difference between observation time and last stock out. *sot* is stockout cost per unit of time.

- A ship needs a long-time delivery since the distance among islands is far and ships cannot move fast. Ships cannot send products simultaneously since the arrival of the ships for assignment are not at the same time. This process needs tracing of ships traveling when deliver products. The observation of the process is done in a time horizon. The objective of this research is to achieve optimum cost of ship deliveries due to consumption port demands in a time horizon. This cost is calculated from travel costs and demurrage costs of ships appear in time horizon. This total cost is defined as follows:

$$TC = \min \begin{pmatrix} \sum_{i=1}^{s} \sum_{j=1}^{r} (\sum_{k=1}^{q-1} TRc_{i,j,k,k+1} \\ +TRc_{i,j,q,1} + DMc_{i,j}) \\ + \sum_{l=1}^{z} \sum_{m=1}^{c} \sum_{n=1}^{p} SOc_{lmn} \end{pmatrix} \tag{6}$$

TC is total cost of bulk product delivery in a horizon of time. *TC* is calculated from costs appear from all ships (1, 2, 3, ..., *s*) and stock out of inventory. Every ship has routes (1, 2, 3, ..., *r*) and every route has travel costs *TRc* between port and port (1, 2, 3, ..., *q*) and return to port 1 (depot), and demurrage cost *DMc*. *DMc* appear once in last port. Inventory cost appear when there is stockout cost of products (*SOc*) for every (1, 2, 3, …., *c*) of cities and (1, 2, 3, …., *p*) of product in (1, 2, 3, …, *z*) of time horizon.

3. Proposed method

3.1 Metaheuristics

In this research, Hybrid Particle Swarm Optimization (HPSO) developed and proposed to solve IRP with bulk products [15]. HPSO is a new hybrid metaheuristic combining PSO, Nahwaz-Enscore-Ham (NEH), and 3-Opt. PSO is metaheuristic method inspired by social behavior of animal, namely flocking birds or fish in finding food [16, 17]. This metaheuristic generates some particles that represent solutions. Each particle has position (*x*) and velocity (*v*) attributes. *x* is calculated based on *v*. They are updated iteratively using the following formulae.

$$v_t = w \times v_{t\text{-}1} + c1 \times rand \times (pBest - x_{t\text{-}1}) \\ + c2 \times rand \times (gBest - x_{t\text{-}1}) \tag{7}$$

$$x_t = x_{t\text{-}1} + v_t \tag{8}$$

where v_t denotes velocity at iteration t, x_t denotes position at iteration t. Both v and x work in d dimension. Each particle keeps its best value called *pBest*. The best of all *pBest* is called group best or *gBest*. At any iteration, *gBest* is the best solution for PSO. It always looks at the last position of the particles compared

Received: September 8, 2022. Revised: December 4, 2022.

with the position of *pBest* and *gBest*. The position of each particle is updated using Eq. (8). This process is repeated until termination criteria are reached.

NEH is a heuristic method that was used for the first time to solve the problem of production scheduling [18]. NEH can find a solution fast. In the initialization of its process, every fitness of the particle is calculated partially, and the result is sorted from worst to the best fitness. This result creates sequence *S*. Every particle is picked up from the sequence *S* and develops a new sequence *N*. The new particle is placed at the best position of sequence *N*. This process is repeated until the last particle of the sequence *S*. Sequence *N* is the solution of this method. Although NEH gives a good solution, for the complex data, NEH is not able to reach an optimum solution. It always provides the same solution every time this method is repeated [15].

3-Opt is a local search algorithm widely used to solve the Traveling Salesman Problem [15]. This algorithm searches three points of the circular route to specify three sequences. The sequences are shown in Fig. 2.

These sequences are being observed to make sure whether there is a possibility of improving the value if the route from the sequence is moved to a new sequence. Seven new possibility sequences can be transferred according to 3-Opt rule as shown in Fig. 2 [15].

GA is a metaheuristic algorithm inspired by natural selection. This metaheuristic method is part of the evolutionary algorithms (EA) class. GA provides solutions to search problems and optimization. The inspiration for biological changes in genetic inheritance such as mutation, crossover, and selection are adopted by GA [19]. A population of candidate solutions called individuals is developed towards a better solution. Each candidate solution called properties can be mutated and changed. Traditionally, solutions are represented in binary as 0 and 1, but other encodings are also possible [20]. The evolutionary process in GA starts from a population of individuals. The values in the individual population are obtained from random numbers. In this paper, some generation process from random numbers is replaced by the NEH algorithm. The next process is an iterative process. In this process, the population is changed by several techniques in each iteration, namely mutation, crossover, and selection. These changes in population are called generations. In each generation, the fitness value of every individual in the population is evaluated. Fitness is the value of the objective function in the optimization problem being solved. Individuals who are more fit are selected stochastically from the current population. The genome of every individual is modified and recombined. The genome can mutate randomly to form a new generation. The new generation of candidate solutions is used in the next iteration of the algorithm. The algorithm terminates when the maximum number of generations has been reached or a satisfactory fitness level has been reached for the population.

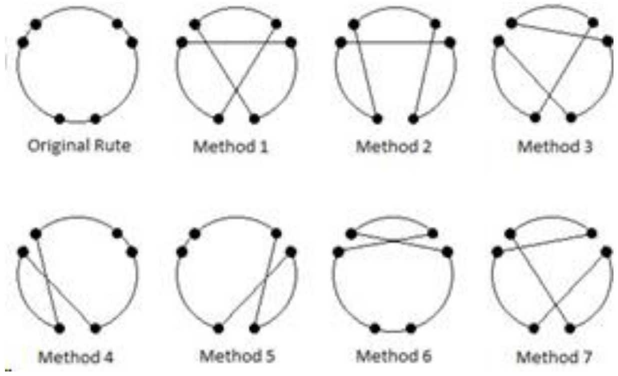

Figure. 2 Seven new sequences in the 3-Opt algorithm transferred from the original connection

TS was first introduced by Glover [21]. Glover stated that TS is a high-level metaheuristic procedure for solving combinatorial optimization problems. This TS is designed to direct other methods (or components of the TS process itself) to exit or avoid entering the local optimal solution. TS has the ability to produce near-optimal solutions that have been utilized in a variety of classical and practical problems. It has been used to solve various fields ranging from scheduling to telecommunications. TS is a metaheuristic algorithm that relies on finding neighborhood solutions and local memory. TS tries to find a neighborhood solution that is better than the current solution. In addition, local memory is used to record the searched steps that have been encountered. If the searched step has been encountered, it is marked as “tabu” or forbidden, then the TS ignore the searched step.

HPSO is developed by combining three metaheuristics algorithms. It uses some techniques to achieve better solutions, namely [15]:

1. Initialization
   Each particle represents the sequence of the city to be visited (1, 2, ..., *c*), *c* denotes the number of consumption ports or cities. In HPSO, initialization of the population in PSO method is replaced by NEH methods to obtain a better result. NEH always provides one solution and in HPSO, NEH is modified (MNEH) to provide more than one result. Therefore, more than one particle can be replaced by MNEH. In this process, PSO and NEH interact each other. NEH, which cannot move after the end of its process, can be moved by

Received: September 8, 2022. Revised: December 4, 2022.

PSO. On the other hand, PSO obtains a better initial value of particles compared to the random initialization. Improving initialization by NEH can be done by 3-Opt method.

2. Quantum technique
   The best value of every population, namely *pBest*. The best value of global population, namely *gBest*. *gBest* is obtained from the best *pBest*. It becomes the result value of PSO along its iterated searching process. In the searching process, *gBest* can be stuck in optimum local as depicted in Fig. 3.
   Quantum technique is used to move *gBest* jump over the local optimum. The purpose of this technique is to move *gBest* to the global optimum. This technique usually uses a random number to replace the value of the particles. HPSO uses NEH to replace particle values for every *pBest* of the populations.
   The value of particles in a *pBest* becomes initialization input data for NEH. NEH process it to search for better value. If this value is better than the *pBest*, it replaces *pBest*. In this research, the quantum technique is not used.
3. Serial improvement
   This process algorithm improves *gBest* value by 3-Opt method. *gBest* becomes initialization input data for 3-Opt. 3-Opt process it and try to search a better result. The output of 3-Opt become the result of HPSO.

NEH is a good method to initialize other methods that usually use random number. It runs fast and provides sharp fitness increasing value. However, NEH stop at a certain value and cannot change anymore. 3-Opt method usually needs a long-time procession. This method uses three-loop processes to find three edges to divide the route to become three sequences. These sequences are exchanged according to the rule of 3-Opt method. In HPSO, 3-Opt is placed once at the end of the process. However, this does not interfere with the efficiency of processing time if it was also placed at the beginning of the process. This updating process is expected to improve the performance of PSO initialization since 3-Opt method can improve NEH method as shown in Table 1.

Table 1. Improvement of 3-Opt to NEH method

No.	City	NEH	NEH + 3-Opt	Ef (%)
1	eil76	596	565	5.12
2	st70	778	684	11.98
3	berlin52	8274	8118	1.89
4	eil51	466	455	2.46
5	att48	35570	34214	3.81
6	ulysses22	76	76	0.25
7	ulysses16	74	74	0.00
8	burma14	31	31	0.00

Table 1 shows Vehicle Routing Problem (VRP) instances in a different number of cities. 3-Opt addition is shown in column NEH + 3-Opt. Column *Ef* denotes the effectiveness of adding 3-Opt. The effectiveness of this modification is calculated based on the following rule as shown in Eq. (9) [15]:

$$Ef = \frac{D_2 - D_1}{D_2} \times 100\% \tag{9}$$

$D1$ denotes the average route distance of NEH + 3-Opt calculation result, while $D2$ denotes the NEH calculation result. The effectiveness is calculated from the deviation between the NEH and NEH + 3-Opt calculation. The effectiveness shows how much the reduction in the route distance obtained by NEH + 3-Opt calculation is compared to the NEH calculation result.

The effectiveness of NEH + 3-Opt calculation improve VRP result value in a big number of cities. 3-Opt does not provide improvement in a little number of cities. The optimal fitness value has been obtained from NEH process. However, adding the process of 3-Opt to improve NEH results provides a better solution for solving combinatorial optimization problems such as IRP. In this research, HPSO is modified to improve the performance and

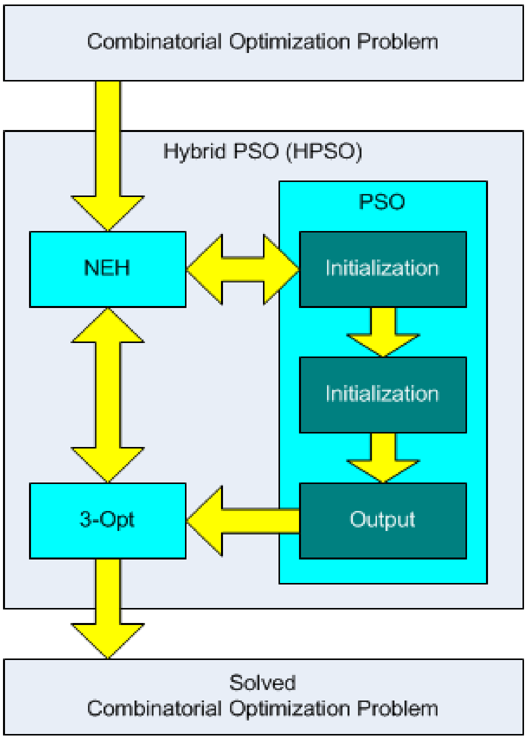

Figure. 4 Modified hybrid particle swarm optimization (MHPSO)

 DOI: 10.22266/ijies2023.0228.39

called Modified HPSO (MHPSO). The model of MHPSO is shown in Fig. 4. The modification of HPSO is performed in PSO initialization carried out by NEH. This process is improved by 3-Opt operation. Some of the result values of this process are used to initialize the populations of PSO. The other populations are set by a random number. Thenext MHPSO processes are carried out similarly to HPSO. The quantum process is not carried out since the PSO iteration process is not carried out in large numbers.

The pseudo-code of the procedure is written in Algorithm 1 as follows:

Algorithm 1

1. For each particle of PSO
2. Initialize particle with random number
3. End For
4. Generate sequences with NEH
5. For each sequence obtained from NEH
6. **Improve sequence with 3-Opt**
7. Replace particle with sequence
8. End For
9. Do
10. For each particle
11. Calculate fitness value
12. If the fitness value is better than the best fitness value (*pBest*) in history
13. set current value as the new *pBest*
14. End If
15. End For
16. Choose the particle with the best fitness value Of all the particles as the *gBest*
17. For each particle
18. For each dimension
19. Calculate particle velocity according to Eq. (4)
20. Update particle position according to Eq. (5)
21. End For
22. End For
23. While maximum iterations or minimum error criteria is not attained
24. Improve particle of *gBest* with 3-Opt

The bold algorithm is the modification of HPSO called MHPSO. The result of initialization obtained from NEH is improved by 3-Opt before it is used to replace the particles of PSO.

The result of MHPSO is compared with the algorithm GA and TS. The GA and TS are also treated in the same way as PSO. Both of them are initialized by NEH and 3-Opt, and the results of GA and TS are improved by 3-Opt. This modification of GA and TS are called Hybrid GA (HGA) and Hybrid TS (HTS).

3.2 Maritime inventory routing problem (MIRP)

MHPSO metaheuristic method is used to solve the MIRP. The purpose of using MHPSO is that the calculation process can be carried out in an acceptable time and the results are close to optimum. Assignment activities are carried out repeatedly every time the ship docks to the depot until the time horizon is reached. The total costs incurred are obtained from the sum of each assignment cost until the time horizon is reached. These costs include travel and inventory costs. The inventory of the depot is assumed unlimited. Therefore, there is no waiting time for the ship at the depot. The algorithms of MIRP processes are shown in Algorithm 2 as follows:

Algorithm 2

1. Set initialization of ships position when arrived to depot
2. Set schedule *LShip*() with ships arrival to depot order by time
3. Set initialization of inventory of every product of port *P* for all of port
4. *i* = 1
5. While LShip(*i*).Time in time horizon
6. Get ship *S* from *LShip*(*i*)
7. If *S* arrive in depot port Then
8. Set *gBest* = best *pBest* for *n* cities obtained from MHPSO
9. Generate *gBest* to *LShip* for every *n* port cities order by time
10. Set status of the port cities as served
11. Else If *S* arrive in consumption ports Then
12. Update inventory status of port for every product
13. Set status of the port city as free to serve
14. End If
15. *i* = *i* + 1
16. End While
17. Calculate total cost

LShip denotes the sequence (1, 2, ..., s) of ships attributes arrived in a port whether it is depot or consumption port. All ships' positions obtained from the initialization process are placed in *LShip* order by arrival time. *P* denotes the attribute of port (1, 2, ..., *c*). The attributes contain inventory position of products in port *p*. Line 5 in Algorithm 2 describes the process of ships' arrivals. This process is executed until the time has exceeded the specified time horizon. The algorithm processes every ship's arrival *S* in *LShip*. The arrival of the ship *S is* processed according to the type of the port, whether it is a depot or a consumption port. If the port is a depot, the ship is assigned to deliver products to some consumption ports.

If assignment of a ship *S* that is arrived in a depot port as in Algorithm 2 line 7, records of every ship arrival in consumption ports obtained from *gBest* are generated and added in *LShip*. *gBest* is the best route of the ship *S* produced by MHPSO. It contains the sequence of cities serviced by the ship, product types brought by the ship, and the number of products send to the cities. *gBest* generates another *LShip* of visited cities. These new *LShip*(s) contain information of arrival time, the city, product type and its amount. When it processes *LShip* of the arrival of the ship in this consumption port as in Algorithm 2 line 11, the status of every product received by consumption city is updated. The process continues and move to the next ship arrival *S* as shown in Algorithm 2 line 15.

If a ship *S* arrived in the depot as written in Algorithm 2 Line 7, this ship *S* is assigned to deliver products to *n* number of consumption ports. $n \leq MaxPort$. In this research, *MaxPort* is limited to only 3 (three) consumption ports. This means the ship receives an assignment to send the bulk product to 1, 2 or 3 consumption cities depending on the carrying capacity of the compartments or several cities that are free to be served. If the total carrying capacity of the compartments exceeds the total demands of consumption ports, the ship wait at the last port until all products are unloaded. This incident has an impact on the ship that will be charged a demurrage cost.

MHPSO provide g*Best*. *gBest* denotes the particle that represents the best route with the optimum cost. This cost includes travel, product stock out, and demurrage cost. *gBest* is obtained from the comparison of the *pBest* with the previous *gBest*. If *pBest* is better than *gBest*, then set the *gBest* with *pBest*. To calculate the fitness value of *pBest*, Algorithm 3 was developed as follows:

Algorithm 3

1. Get sequence of port *SeqPort* from MHPSO
2. Get sequence of port *Seq* from *SeqPort* contain city can be served if ≤ *MaxPort*
3. Set sequence *SProd* = Sum (product) of port *Seq* group by product type
4. Fill ship compartments with product with the type are determined by *SProd* sort by product number descending
5. Set *city1* = depot
6. Set *TimeStart* = time out from depot
7. For *s* = 1 to number of *Seq*
8. If ship compartments are empty Then exit for
9. Set *city2* = *Seq*(*s*)
10. Add *pBest*(s)
11. Set *pBest*(s).*city* = *Seq*(*s*)
12. Set *pBest*(s).*TimeIn* = *TimeStart* + distance time from *city1* to *pBt*(*s*).*city*
13. Add *pBest*.*TravelCost* according to distance of *city1* and *city2*
14. If *city2* is last city Then
15. Set *pBest*(s).*TimeOut* = *pBest*(s).*TimeIn* + unloading time of product + demurrage time
16. Set *pBest*(s).*ProductDeliver*() = all unloaded product type carried by ship
17. Add *pBest.DemmurrageCost* according to demurrage time
18. Add *pBest*.*StockOutCost* from empty product in consumption port according to delivered product
19. Else
20. Set *pBest*(s).*TimeOut* = *pBest*(s).*TimeIn* + unloading time of product
21. Set *pBest*(s).*Deliver*() = all unloaded product type carried by ship
22. Add *pBest*.*StockOutCost* from empty product in consumption port according to delivered product
23. Endif
24. Set *city1* = *pBest*(s).*city*
25. Set *TimeStart* = *pBest*(s).*TimeOut*
26. Next *s*
27. Add *pBest*(s)
28. Set *pBest*(s).*city* = depot
29. Set *pBest*(s).*TimeIn* = *TimeStart* + distance time from *city1* to *pBest*(*s*).*city*
30. Add *pBest*.*TravelCost* according to distance of *city1* and depot
31. Set *pBest*.*Fitness* = *pBest*.*TravelCost* + *pBest*.*DemmurrageCost* + *pBest*.*StockOutCost*

SeqPort in Algorithm 3 line 1 denotes the sequence (1, 2, …, *c*) of cities or consumption ports served by the depot. *Seq* in Algorithm 3 line 2 denotes the sequence (1, 2, …, *c*) of the chosen city from *SeqPort* served by the ship. *SProd* in Algorithm 3 line 3 denotes the sequence (1, 2, ..., *p*) of an aggregate number of every product type from cities in *Seq*. It contains the sum of products grouped by product type. The next process in Algorithm 3 line 4, *SProd* is sorted descending. The first product types in *SProd* are used to determine the product type brought by the ship in compartments. Not all product types are brought because of the limited number of compartments. The next process in Algorithm 3 is from lines 7 to 26. In this process, *pBest* is used to record the traveling of the ship for every city *s*. *pBest* contains information of the consumption cities in the route, what kind of product types and the number of them are delivered by the ship, and the arrival time to the cities. *pBest* also records the costs globally for all cities, namely travel cost, demurrage cost, and stockout cost. The last record of *pBest* is used to save information of the ship that returns to the depot city. It contains arrival time and travel cost. The fitness value of *pBest* that is recorded in *pBest.fitness* is accumulated from the costs.

Total cost is calculated after all processes are done as in Algorithm 2 line 17. According to Eq. 6, total cost (*TC*) is sum of travel cost (*TRc*), demurrage cost (*DMc*), and stockout cost (*SOc*). Total cost can be used to measure the effectiveness of heuristics method to calculate MIRP. Different metaheuristic methods can be compared on their total cost value. Travel cost and demurrage cost can be calculated from every record of *LShip*(). Stockout cost has to be traced from inventory status from (hour) zero and incremented for one hour until the time horizon because *LShip*() only records the number of product delivered to the cities. Stockout cost has to calculate stock out of other products not delivered. The algorithm of the total cost is shown in Algorithm 4 as follows:

Algorithm 4

```
1.  Set TC, TRc, DMc, SOc = 0
2.  For i = 1 to number of records in LShip()
3.      TRc = TRc + LShip(i).TravelCost
4.      DMc = DMc + LShip(i).DemmurrageCost
5.  Next i
6.  idx = 1
7.  For i = 1 to time horizon
8.      If LShip(idx).Time < i Then
9.        If LShip(idx).City <> depot Then
10.         For j = 1 to number of product
11.           If LShip(idx).Product(j) <> 0 Then
12.             Set CityLShip(idx).Inventory =
                LShip(idx).ProductDeliver
13.           End If
14.         Next j
15.       End If
16.       idx = idx + 1
17.     End If
18.     For j = 1 to number of consumption city
19.       For k = 1 to number of product
20.         Set City(j).Product(k) = City(j).Product(k)
            – City(j).DecSpeed(k)
21.         If City(j).Product(k) ≤ 0 Then
22.           City(j).Product(k) = 0
23.           SOc = SOc + City(j).SOcUnit
24.         End If
25.       Next k
26.     Next j
27. Next i
28. Set TC = TRc + DMc + SOc
```

In the first line, all cost is set to zero. Travel cost (*TRc*) and demurrage cost (*DMc*) are calculated from *LShip*().*TravelCost* and *LShip*(*i*).*DemmurrageCost* in Algorithm 4 line 3 and 4 from every record of *LShip*(). Stockout cost (*SOc*) is traced from (hour) zero until time horizon as shown in algorithm 4 line 7 to 27. For every record in *LShip*() that matches with the tracing time, the inventory status of city is updated in accordance with the number of products delivered to the city as shown in algorithm 4 lines 9 to 17. Every tracing time, the inventory status of every product in all of the cities is reduced by the decrement speed of the product (*City*(*j*).*DecSpeed*(*k*)). Decrement speed is product decrement that is delivered from consumption city to the retailers per hour. If inventory status of the products lower or equal to zero, then it is charged a stockout cost per hour (*City*(*j*).*SOcUnit*) as shown in algorithm 4 lines 21 to 24. Finally, total cost is calculated as shown in Algorithm 4 line 28.

4. Experimental results

The problem of this paper is the transportation of bulk cement from factory port as depot to some consumption port cities. The distance between cities is shown in Table 2. There are 10 cities, while city C1 is the depot. The distances are measured by ship trips in a nautical mile. The factory produces two types of products namely Ordinary Portland Cement (OPC) and Portland Composite Cement (PCC). The delivery of the cements does not depend on the demand of the consumption cities but the decreasing speed of every product sent to retail. When products are stock out, the company applies stockout cost. When the process is started, the initial for every product is set.

Big capacity ships are used to deliver cements. There are 4 ships used by the company, namely V1, V2, V3, and V4. These ships have the different number of compartments and the capacity of compartments. The ships used to deliver products are shown in Table 3.

Loading and unloading speeds are used to load cement from depot and to unload cement to the consumption city. This activity affects the processing and the exit port time. When the process is started, the initial for every ship is set. This is the initial time when the ships arrived at the depot.

The process is started by determining the long of time horizon expected to be calculated. Throughout the process, *LShip*() is executed one by one. The

Received: September 8, 2022. Revised: December 4, 2022.

Table 2. Distances between cities

City	Inventory Status/St (ton) and Decreasing Speed/Sp (ton/h)				Stockout cost ($/h)	Distance (Nautical Mile)									
	PCC		OPC												
	St	Sp	St	Sp		C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
C1	0	0	0	0	10	0	744	432	163	274	361	301	634	768	882
C2	459	8	0	3	10	744	0	369	616	617	928	606	382	150	449
C3	64	5	509	9	10	432	369	0	573	680	728	711	336	417	558
C4	565	4	547	5	10	163	616	573	0	153	376	170	774	713	1021
C5	412	1	0	3	10	274	617	680	153	0	352	78	881	715	1023
C6	580	7	643	7	10	361	928	728	376	352	0	412	930	1025	1178
C7	543	1	407	5	10	301	606	711	170	78	412	0	913	703	1011
C8	192	8	0	3	10	634	382	336	774	881	930	913	0	321	361
C9	82	7	91	3	10	768	150	417	713	715	1025	703	321	0	374
C10	361	7	209	6	10	882	449	558	1021	1023	1178	1011	361	374	0

Table 3. Ships used to deliver product

Ship	Compartment Capacity (Ton)		Speed (m/h)	Loading and Unloading Speed (ton/h)	Travel Cost ($/m)	Demurrage Cost ($/h)	Initial time when arrives to depot (h)
	1	2					
V1	1500	0	15	20	2	1	120
V2	5200	0	15	20	3	2	220
V3	4000	3500	12	20	5	4	310
V4	3400	2500	12	20	4	3	450

Table 4. *LShip*() generated from initialization

Record No	Time (h)	Ship	City
1	120	V1	Depot
2	220	V2	Depot
3	310	V3	Depot
4	450	V4	Depot

execution will stop when *LShip*().*Time* achieves time horizon. These steps of the process are described as follows:

Step 1 – *Ship Initialization* sets the time when every ship achieves to the depot as in Algorithm 2 line 1. Even though the ship is still on duty to deliver products to cities, this is not taken into account because the assignment of the ship is out of the current process. The initialization required for the ship is when the ship returns to the depot for the next assignment.

***Step* 2** – *Generate LShip*()

LShip() is generated from the position of the ships as in Algorithm 2 line 2. The ships create 4 records of *LShip*() from ship V1 to V4. This data is obtained from Table 3. *LShip*().Time is set from the time of ship when arrived at the depot and the records are ordered by the time of *LShip*(). Ship is set with the ships and *LShip*(). City with depot. Those 4 records of *LShip*() is executed one by one until time horizon is achieved. The generation of *LShip*() data from the ship initializations are shown as in Table 4.

Step 3 – *Inventory Initialization*

According to Algorithm 2 line 3, it sets the inventory status for every product in all cities. The status is set from the data from Table 2 for both kind of products (PCC and OPC). The time to set the number of every product is set to zero for the beginning.

Step 4 – *Processing LShip*()

LShip() generated by ship initializations is processed one by one as in Algorithm 2 line 5. There are 2 conditions of *LShip*() status according to the type of city where the ship arrives. If the city is a depot, then go to *Step 5* to assign the ship to deliver products to the consumption city, otherwise, go to *Step 8*.

Step 5 – *Assigning the Ship*

The first record of *LShip*() is ship V1 which arrives in the depot at (hour) 120. MHPSO provides the

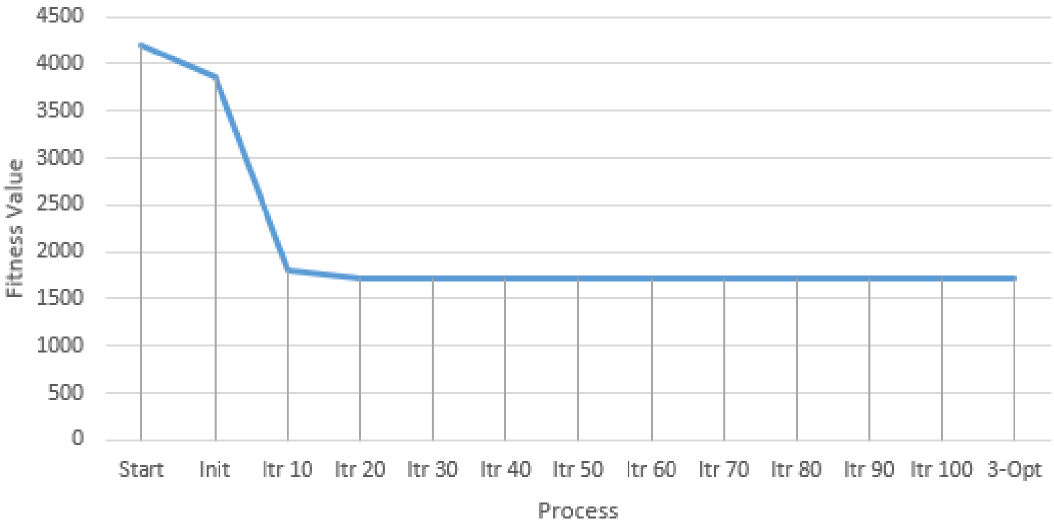

Figure. 5 Calculation result

sequence of cities that expect the delivery by the ship as in Algorithm 2 line 8. It starts from NEH, then is improved by 3-Opt, is processed by PSO, and the last is improved by 3-Opt. The result is stored in *pBest*. Every result that has a fitness value is compared with the previous the best *pBest*. This process takes some time and is iterated to obtain the best results of *pBest.* It is stored in the *gBest*. For example, MHPSO providing the sequence of cities (8, 2, 10, 3, 7, 4, 6, 9, 5) is stored in *SeqPort* as in Algorithm 3 line 1. The process looks for cities amounting to *MaxPort*. The cities are not in the plan of shipping by other ships as in Algorithm 3 line 2. The cities in order are 8, 2, 10, and return to depot. The products needed by the cities are recapitulated and sorted descending as in Algorithm 3 lines 3 and 4. The product required is the maximum product storage capacity in the cities, minus the inventory status of the product. The cities need product PCC = 1988 ton and OPC = 2791. OPC is more than PCC, it is, therefore, prioritized. Since the ship V1 has only one compartment, the ship only carries OPC as much as compartment capacities of 1500 ton. The ship starts to travel from one city to another, starts from city 8. The travel cost and arrival time are calculated. Product OPC is calculated on how many OPC are unloaded to the silo of city 8. If the inventory status was empty then the silo was filled to its capacity, otherwise, it was filled with the remaining available capacity. If inventory status had been emptied for many hours started from the last status time, the ship is charged a stockout cost. Process time is calculated to set the arrival and the departing time. If the contents of the compartment are still there, the ship continue its journey to the citiy 2 and 10. Especially for city 10 which is at the last destination, if the remaining amount of cement is still there, the ship wait until the entire contents of the compartment are sent. In this case, the ship is charged a demurrage cost and the traveling time is added. At this step of the process, the ship can only serve city 8 and 2 because of the compartment capacity. The assignments of the ship V1 are shown in Table 5. This data is recorded in *pBest* and repeated until the searching process by MHPSO completes. According to the fitness value of the data, the best calculation of *pBest* is recorded to the *gBest*.

The calculation result for the first record of *LShip*() is depicted in Fig. 5. Fig. 5 shows the initialization process increase the fitness value. It is increased after 20 iterations and does not change until the end of the process. The fitness value that is calculated from step PSO process is in the best result, therefore improving by 3-Opt does not provide better result.

Table 5. Assignments of the ship V1

Parameter	City 8	City 2	Depot	Global
Time in (h)	237	312	387	-
Time out (h)	287	337	-	-
Product delivered/ OPC (ton)	1000	500	-	-
Travel cost ($)	-	-	-	3520
Stockout cost ($)	-	-	-	4822
Demurrage cost ($)	-	-	-	0
Fitness	-	-	-	8342

Table 6. *LShip*() generated from initialization

Record No	Time (h)	Ship	City	Information
1	120	V1	Depot	Initialization
2	220	V2	Depot	Initialization
3	237	V1	8	Generation
4	310	V3	Depot	Initialization
5	312	V1	2	Generation
6	387	V1	Depot	Generation
7	450	V4	Depot	Initialization

Step 6 – generate *gBest* to *LShip*()
gBest obtained from MHPSO as shown in Table 5 is used to regenerate records in *LShip*() as in Algorithm 2 line 9. The updating of *LShip*()depended on the time when the ship arrives to the city as shown in Table 5. The updated *LShip*() is shown in Table 6.
***Step 7** – Status city is served*
All cities are set as "served" as in Algorithm 2 line 10. Therefore, there are no other ships will plan to be assigned to deliver product to the cities before the ship arrive to the city. The next process returns to the Algorithm 2 line 5. It is continued to the next record, namely record number 2 from Table 6.
***Step 8** – Update inventory status*
This step is continuation of *Step 4* when the current record of *LShip* is a record when the ship arrives in city beside the depot as in Algorithm 2 line 11. An example of this step is shown in Table 6 record number 3. When ship V1 arrive in city 8, the inventory status of OPC is updated from zero ton (Table 2) to become 100 ton as assignment of ship V1 to city 8 in Table 5.
***Step 9** – Count Total Cost*
Total cost is calculated after *LShip*().*Time* achieves the time horizon as in Algorithm 2 line 17. The calculation algorithm is shown as in Algorithm 4.

The experiments of MIRP data instances are conducted to test the performance of metaheuristic method proposed in this research, namely MHPSO. Hybrid PSO is used since it provides a better result than the original method [15]. The proposed method is compared with other metaheuristic methods, such as GA, TS, PSO, HGA and HTS to prove the ability of this method in solving MIRP to find the lowest total cost. In this case, the time horizon is observed for 720 hours (30 days). The process of PSO, GA, and TS are iterated for 100 iterations. The number of ships used to deliver the products is varied from three to five to measure the utility of the ship appropriately. The result is recalculated for 30 calculations and the data with minimum total cost of every method is selected as best method for certain data. Total cost is calculated from travel cost, demurrage cost, and stock-out cost. This process is observed from 10 sets of data instances from city C10 to C28 of which result can be seen in Table 7.

Table 7 shows that MHPSO provides on average 0.64% more effective than GA, TS, PSO, HGA and HTS for some data (C10, C12, C16, C26, C28). In this table, it is also concluded that hybrid method that is implemented in PSO (MHPSO) provides 70% better solution of PSO results, while hybrid techniques implemented in GA (HGA) and in TS (HTS) do not have better results of their original methods without hybrid.

Total cost which consists of travel, demurrage, and stock out costs of every method in Table 7 show a certain pattern. Fig. 6 shows a graphic of costs obtained from MHPSO. The minimum total cost of the data calculated by MHPSO is achieved when it is calculated data from C12 data instance.

Table 7. Total cost of metaheuristic methods ($)

Data	City Amt	GA	TS	PSO	HGA	HTS	MHPSO	Efficiency
C10	10	483238	521574	483238	521574	521805	**450214***	6.83%
C12	12	350772	351723	350772	350772	351740	**350067***	0.20%
C14	14	382057	**381714***	382057	382057	387457	382057	-0.09%
C16	16	413645	411169	412186	413023	415307	**410802***	0.09%
C18	18	436071	**431530***	431863	438785	442437	432649	-0.26%
C20	20	463552	**462705***	464695	464023	467270	464015	-0.28%
C22	22	487712	**487082***	488148	489086	491798	487678	-0.12%
C24	24	**520994***	521315	521162	521962	529339	521162	-0.03%
C26	26	548331	548865	548165	549504	558597	**547647***	0.09%
C28	28	582595	582901	580694	582975	583900	**580673***	0.00%
							Average	0.64%

Received: September 8, 2022. Revised: December 4, 2022.

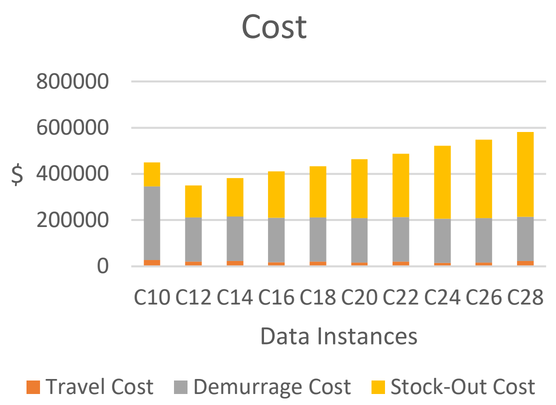

Figure. 6 Costs of MHPSO results

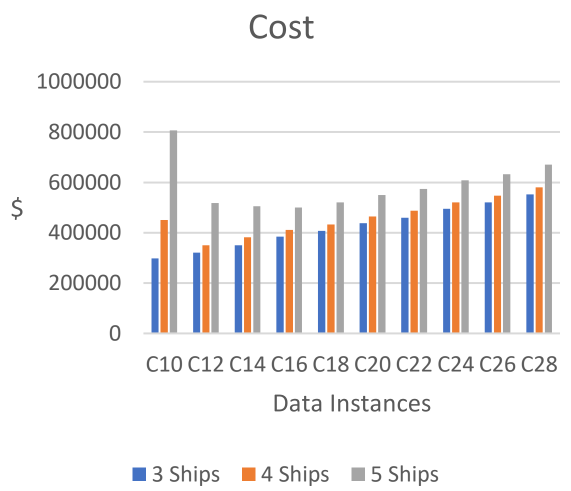

Figure. 7 Ship's utility

The total costs in Fig. 6 use four ships to deliver product. Different number of ships produce different optimum result which can be seen in Fig. 7. Based on Fig. 7 it can be concluded that the MIRP of all data has minimum cost when it uses three ships. This result can be considered as the reason why to choose three ships only to deliver product from depot to all consumption ports. The total cost in Fig. 7 still does not consider the charter cost of the ships. If the charter cost is considered, the saving cost will be much higher when using only three ships.

5. Conclusions

In this paper, the MIRP problem has been solved using some metaheuristics. MHPSO is hybrid metaheuristics that can provide the best solution effectively in reasonable computing time. MIRP can be solved by tracing the shipping activity. This technique is used to investigate the moving of ships to deliver products from depot to the consumption cities. Inventory status must be maintained to avoid stock out of products from depot to the demand cities. Some ships are used to deliver products to some different cities. The assignment process was repeated over the time horizon each time a ship arrives at the depot. This process takes a few moments to determine the best route to reach the demand cities, calculate the travel time and visit each city, determine the type and amount of product to be carried by ship, and calculate the costs incurred. This complex problem is better solved using metaheuristics due to time consumption reasons.

For the next research, MIRP problem can be applied to different cases. MIRP can be applied to other bulk products, such as bulk oil transportation. The varied data might be set in more detail to approximate the real conditions and create a flexible case. Costs can be broken down more thoroughly to obtain precise decision-making, and different ships can deliver products to the city that has been planned served by the other ship as long as it delivers different products. A new algorithm can be developed to calculate the optimum utility of the ships used to deliver product in MIRP.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, Nurhadi Siswanto and Antono Adhi; methodology, Nurhadi Siswanto and Antono Adhi; software, Antono Adhi; validation, Nurhadi Siswanto, Antono Adhi, and Budi Santosa; formal analysis, Nurhadi Siswanto, Antono Adhi, and Budi Santosa; investigation, Nurhadi Siswanto and Antono Adhi; resources, Nurhadi Siswanto, Antono Adhi, Budi Santosa; data curation, Antono Adhi; writing—original draft preparation, Nurhadi Siswanto and Antono Adhi; writing—review and editing, Nurhadi Siswanto, Antono Adhi, and Budi Santosa; visualization, Antono Adhi; supervision, Nurhadi Siswanto and Budi Santosa; project administration, Nurhadi Siswanto; funding acquisition, Nurhadi Siswanto.

Acknowledgments

The authors wish to acknowledge the funding of this research is from the Ministry of Education, Culture, and Research Republic of Indonesia through the Doctor's Dissertation Research Scheme Grant No. 869/PKS/ITS/2021.

Received: September 8, 2022. Revised: December 4, 2022.

References

[1] M. Christopher, “Logistics and supply chain management”, *4th edition, UK: Prentice Hall*, 2011.

[2] R. Setiawan, “Order allocation and purchasing transportation planning in the garment supply chain: a goal-flexible planning approach”, *Industrial Engineering & Management Systems*, Vol. 20, No. 2, pp. 201-212, 2021.

[3] E. S. Pavlova and R. A. Shichiyakh, “A robust mathematical model for vehicle routing problem with simultaneous pickup and delivery and worker allocation”, *Industrial Engineering & Management Systems*, Vol. 20, No. 2, pp. 201-212, 2021.

[4] O. Berman and Q. Wang, “Inbound logistic planning: minimizing transportation and inventory cost”, *Transportation Science*, Vol. 40, No. 3, pp. 287-299, 2006.

[5] M. Soysal, J. M. B. Ruwaard, R. Haijema, and J. G. A. J. V. D. Vorst, “Modeling a green inventory routing problem for perishable products with horizontal collaboration”, *Computers and Operations Research*, Vol. 89, pp. 168-182, 2018.

[6] X. Xin, X. Wang, X. Tian, Z. Chen, and K. Chen, “Green scheduling model of shuttle tanker fleet considering carbon tax and variable speed factor”, *Journal of Cleaner Production*, Vol. 234, pp. 1134-1143, 2019.

[7] B. Santosa, R. Damayanti, and B. Sarkar, “Solving multi-product inventory ship routing with a heterogeneous fleet model using a hybrid cross entropygenetic algorithm a case study in Indonesia”, *Production & Manufacturing Research*, Vol. 4, No. 1, pp. 90-113, 2016.

[8] N. Siswanto, D. Essam, and R. Sarker, “Multi-heuristics based genetic algorithm for solving maritime inventory routing problem”, In: *Proc. of IEEE International Conference on Industrial Engineering and Engineering Management*, 2011.

[9] Y. Shao, K. C. Furman, V. Goel, and S. Hoda, “A hybrid heuristic strategy for liquefied natural gas inventory routing”, *Transportation Research Part C*, Vol. 53, pp. 151-171, 2015.

[10] I. T. Peres, H. M. Repolho, R. Martinelli, and N. J. Montheiro, “Optimization in inventory-routing problem with planned transshipment: A case study in the retail industry”, *International Journal of Production Economics*, Vol. 193, pp. 748-756, 2017.

[11] M. W. Friske and L. S. Buriol, “A multi-start algorithm and a large neighborhood search for a maritime inventory routing problem”, In: *Proc. of IEEE Congress on EVolutionary Computation (CEC)*, 2020.

[12] A. Hemmati, L. M. Hvattum, M. Christiansen, and G. Laporte, “An iterative two-phase hybrid matheuristic for a multi-product short sea inventory-routing problem”, *European Journal of Operational Research*, Vol. 252, pp. 775-788, 2016.

[13] A. De, S. K. Kumar, A. Gunasekaran, and M. K. Tiwari, “Sustainable maritime inventory routing problem with time window constraints”, *Engineering Applications of Artificial Intelligence*, Vol. 61, pp. 77-95, 2017.

[14] D. Xu, K. Li, X. Zou, and L. Liu, “An unpaired pickup and delivery vehicle routing problem with multi-visit”, *Transportation Research*, Vol. 103, pp. 218-247, 2017.

[15] A. Adhi, B. Santosa, and N. Siswanto, “Hybrid metaheuristics for solving vehicle routing problem in multi bulk product shipments with limited undedicated compartments”, *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 5, pp. 320-335, 2021, doi: 10.22266/ijies2021.1031.29.

[16] J. Kennedy and C. Eberhart, “Particle swarm optimization”, In: *Proc. of IEEE Int'l Conf. on Neural Networks Vol. IV*, 1995.

[17] B. Santosa and P. Willy, “Metoda metaheuristik konsep dan Implementasi”, *Surabaya: Guna Widya*, 2011.

[18] M. Nawaz, E. E. Enscore, and I. Ham, “A heuristic algorithm for the m-machine, n-job flow- shop sequencing problem”, *The International Journal of Management Science*, Vol. 11, No. 1, pp. 91-95, 1983.

[19] M. Mitchell, “An Introduction to genetic algorithms”, *Cambridge, MIT Press*, 1996.

[20] D. Whitley, “A genetic algorithm tutorial”, *Computer Science Department, Colorado State University*, 1994.

[21] F. Glover, “Future paths for integer programming and links to artificial intelligence”, *Computers and Operations Research*, Vol. 13, No. 5, pp. 533-549, 1986.