IJMSIT

International Journal of Multidisciplinary Studies and Innovative Technologies

Volume : 2
Number: 2
Year: 2018
Pages: 67-71

A PSO Approach to Navigational Shortest Path Problem on an Experience-Based Network Model of the Aegean Sea

Timur İNAN[1, 2], Ahmet Fevzi BABA [3]

[1]*Electrics-Electronics Engineering, PhD. Student, Marmara University, İstanbul, Turkey*
[2]*Computer Programming Program, Instructor, Istanbul Arel Universiy, Istanbul, Turkey*
Mail: timurinan@arel.edu.tr, Tel:08508502735/2133
[3]*Electrics-Electronics Engineering, Prof. Dr., Marmara University, Istanbul, Turkey*
Mail: fbaba@marmara.edu.tr, Tel: (216) 336 57 70 / 1250

[*]*Corresponding author: timurinan@arel.edu.tr*

Abstract – Finding shortest routes for commercial vessels has ever been an important issue on marine science. Shortest route means less sailing time, less sailing time means faster delivery of cargo, less consumption of fuel, less human power. In this study, we present a particle swarm optimization approach to navigational decision support system. The study is applied on the Aegean Sea. The proposed system is a dynamic decision support system that calculates the shortest path from any starting node to any finishing node. The network model consists of 604 nodes. Any ship using this system can find the shortest route to finishing node dynamically.

Keywords – particle swarm optimization, decision support system, shortest path, Aegean Sea

I. INTRODUCTION

Shortest path problem (SP) has always been the focus on many studies. Finding the shortest path or route for vessels reduces fuel consumption and ensures efficient use of time. Particle swarm optimization (16) is an optimization method which is inspired by the behaviour of a swarm to find food. To solve a problem particles are created, at first every particle has their own coordinate and speed values. Every particle tries a function to find the best solution, this is called fitness function. After evaluating this function every particle has new fitness value, in every iteration this fitness function value is examined and particles obtain the best fitness value as local best. In this study, we propose a navigational decision support system which is based on particle swarm optimization. The network model has been prepared in the light of the experience gained in the time the author worked on commercial ships. To simulate the voyages, a map of the Aegean sea is created using two different shape files (2, 3). Total 61 ports on the Aegean sea are recorded by the means of latitude, longitude and names. For the realization of cruises, 604 nodes have been created and coordinates are recorded. The connections between the nodes and angles between them are calculated and recorded. The map showing the nodes on the prepared map can be seen in Fig. 1.

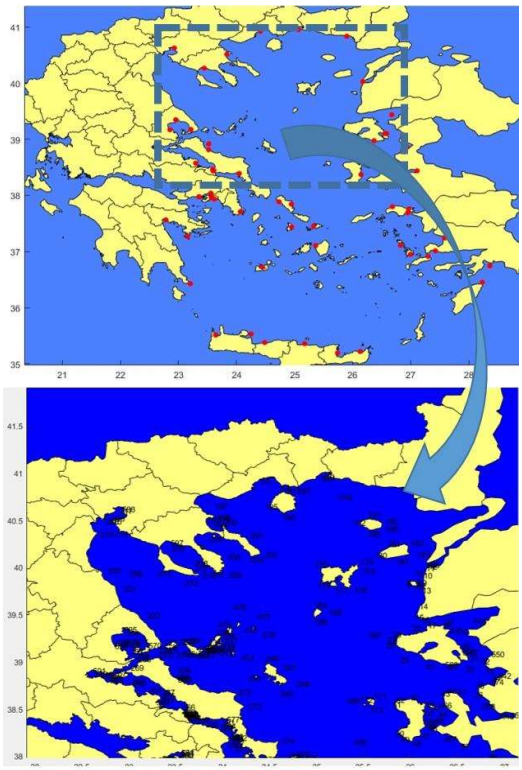

Fig. 1 The map, ports and the nodes

II. RELATED WORKS

To solve the shortest path problem, common and wellknown algorithms exist. For example Dijkstra(19) ve BellmanFord(20) algorithms are most popular algorithms for this purpose. Besides these algorithms, some optimization algorithms like genetic algorithm (18), ant colony algorithm (14), particle swarm optimization algorithm (17) and neural networks (15) are used for solving the shortest path problem. In marine science, SP problem exists for various subjects like

collision avoidance(13), weather routing(12), decision support system calculation(11). For the Aegean sea, some works have been carried out for some purposes like ship routing, fleet routing, weather routing, risk assessment. The related works are all includes only Greece side of the Aegean sea. This study includes both side of Aegean sea (Turkish and Greek side), all the port names and coordinates are same as the real world data. (10) proposed a system that uses ant colony algorithm for fleet routing. The study includes 13 ports and 39 sea links, the links between the ports are accepted as a direct line. In the real world, there are no direct connection between all ports. (9) proposed a weather routing system for fleet routing using genetic algorithm. The study again includes a few ports of the Aegean sea and uses Google Maps for mapping. (8) developed a weather routing system that uses an exact algorithm. This study like the previous works, only examines the Greek side of the Aegean sea and uses Google Maps for mapping. (7) proposed an optimal ship routing system that uses the Aegean sea as the application area. The system uses simulated annealing algorithm. In the study, the proposed algorithm only tested on a sample voyage from the port of Thessaloniki to the port of Agios Nikolaos. For prediction of environmental risk of a possible accident on the Aegean sea, (6) proposed a system using Bayesian network. The study uses all of the Aegean sea, but it examines the risk prediction not routing. We have previously used the ant colony algorithm(4) and genetic algorithm(5) for the system we have introduced. Our study differs from the related works. First; we use a unique map of the Aegean sea that is never been used in any other articles. Second; we examine all the ports of the Aegean sea. Third; the ports and nodes are same as the real world data, coordinates, distances and angles between nodes extracted using the unique map. Fourth; related works use Google Maps for mapping, this mapping style needs internet connection but our map doesn't. Fifth; for ship movement we use Fossen mathematical model (1) of ship which is never used in a study that examines the Aegean sea. Sixth; the links between nodes are not considered as a direct line, the connections are calculated over the created map.

III. MATERIALS AND METHODS

A. Mathematical Model Used

The mathematical model of ship was taken from the website of marine simulator system which is created by Fossen and Perez (21). There are several m files to use for MATLAB. We use tanker.m file for ship mathematical model.

B. Calculation of Distances Between the Nodes

For calculation of the distances, angles and connections between the nodes, calculations are made using i and j operators. As can be seen in equation (1).

$$i = [1, 2, 3, \cdots, 604] \, , j = [1, 2, 3, \cdots, 604] \quad (1).$$

For calculation of distances between the nodes, "distdim" and "distance" functions of MATLAB program was used. "distdim" function normally gives the distance result as kilometer, the distance result was converted into degree to use on map. Let's suppose D is a matrix that contains distance datas. D matrix was filled with the data using the equation (2).

$$D_i^j = \begin{cases} i \neq j, \text{distdim}(\text{distance}(lat_i, lon_i, lat_j, lon_j), \\ \text{'km'}, \text{' deg'})) \\ else\ 0 \end{cases} \quad (2)$$

C. Calculation of Angles Between the Nodes

For calculation of angles between the nodes, "distance" and "rad2deg" functions of MATLAB program is used. "distance" function normally gives both distance and angle data but for calculation of angle the distance value is omitted. "distance" function gives angle result in radians. To convert the angle from radians to degree "rad2deg" function was used. Let's suppose Θ is the matrix that contains angle datas. Θ matrix was filled with the data using the equation (3).

$$\Theta_i^j = \begin{cases} i \neq j, \text{rad2deg}(\text{distance}(lat_i, lon_i, lat_j, lon_j) \\ else\ 0 \end{cases} \quad (3)$$

D. Calculation of Connections Between the Links

After saving all the coordinates of the ports and nodes, the connection between each node must be calculated and saved. The connection between the nodes are calculated using Fossen ship mathematical model which can be seen in equation (4).

$$[\dot{x}, U] = \text{tanker}([\text{ship.suv}, \text{ship.swv}, \text{ship.yw}, \text{ship.x}, \text{ship.y}, \text{ship.psi}, \text{ship.delta}, \text{ship.n}], [\text{ship.deltac}, \text{ship.nc}, 40]); \quad (4)$$

Where,
suv=surge velocity (m/s),
swv=sway velocity (m/s),
yw= yaw velocity (rad/s),
x= position in x-direction (m),
y= position in y-direction (m),
psi=yaw angle (rad),
delta=actual rudder angle (rad),
n=actual shaft velocity (rpm) - nominal propeller 80 rpm,
deltac=commanded rudder angle (rad),
nc=commanded shaft velocity (rpm),
h=water depth, must be larger than draft (m) - draft is 18.46 m,
$\dot{x}$=derivative of suv, swv, yw, x, y, psi, delta and n,
U=velocity of the ship.

Ship begins the voyage from the starting node, the voyage continues to the finishing node. If the ship ends the voyage without any collision to land, connection value is 1, else connection value is 0. To obtain if the ship is on land or on sea, "ltln2val" function of MATLAB program was used. This function gives 3 different results. 0 for land, 1 for shore, 2 for sea. A matrix called C was created and filled with the connections results using the Algorithm 1.

Algorithm 1 Calculation of connection

```
1: for i=1 to 604 do
2: for j=1 to 604 do
3:   if i not equal to j then
4:   result=1;
5:     while true do
6:     move the own vessel using ship mathematical model;
7:     a = ltln2val(Z, R, pt.lat, pt.lon);
8:       if a==1 or a==0 then
9:       result=0;
10:      break;
11:      end if
12:      if ship reaches to the finishing node then
13:      break;
14:     end if
15:   end while
16:  C_i^j = result;
```

17: **else**
18: $C_i^j = 0$;
19: end **if**
20: end **for**
21: end **for**

p_t means point at t*th* step of the ship during the voyage. p_t. lat and p_t. lon means latitude of tth point's latitude and tth point's longitude. To obtain if the point is on land or not, the map converted into data grid and a referencing vector. Z is regular data grid converted from the map vector data, R is the referencing vector for the computed grid. For this purpose, "vect2mtx" function of MATLAB program was used.

E. Particle Swarm Optimization

Particle swarm optimization is an optimization method which is inspired from the behaviour of a swarm. To solve a problem all the particles of the swarm are created. All particles try a fitness function and get a score after trying the fitness function. If the score is better than the best score of the particle. The best score of the particle (pBest) is updated. After all the particles try the fitness function, the best score of the swarm is chosen, it is called local best. After choosing the local best value, it is compared with another value which is called global best. Global best is the score of all particles throughout all iterations. Global best value is updated at the end of every iteration by comparing it with the local best value. If local best value is better than global best value then global best value is exchanged with the local best value which is better than the global best value. At the end of every iteration particle with the best fitness value commands the other articles to change their speeds according to its coordinates and speed. Every particle changes their speed value according to leader particle's coordinates. The formulation of speed and position values can be seen in Equation (5,6).

$$\mathrm{v_{in}} = \mathrm{v_{in}} + c_1 * r_1 * (\mathrm{pBest_n} - \mathrm{pActual}_n) + c_2 * r_2 * (\mathrm{gBest_i} - \mathrm{pActual}_n) \quad (5)$$

$$x_{in} = x_{in} + v_{in} \quad (6)$$

Where $\mathrm{v_{in}}$ denotes speed of n*th* particle in i*th* iteration, c_1 denotes the effect coefficient of particle's best position to new speed of the particle. r_1 is the weight value that determines the final effect of the best position of the particle and is a random number between 0 and 1. c_2 denotes the effect coefficient of swarm's best position to new speed. r_2 is the weight value that determines the final effect of the best position of the swarm and is a random number between 0 and 1. $\mathrm{pBest_n}$ represents the nth particle's best position, $\mathrm{pActual}_n$ presents the n*th* particle's actual position. $\mathrm{gBest_i}$ represents the swarm's best position at ith iteration. At the end of the next iteration particles with new speed values may find a better value while evaluating fitness function by doing this every particle tries their chances to find a better global best value. Ending criteria can be several different criterias, that's up to the programmer. If programmer wants to end the loop in a certain step, ending criteria can be iteration size or if programmer wants to end the loop the ending criteria can be finding a certain value for the optimization. In this study to solve container loading problem, we are going to use particles, their speed and coordinate values to find the best solution for the problem. The biggest problem in applying the pso algorithm to the shortest path problem is encoding the nodes to the functioning of the particles forming the swarm. For doing this, an encoding technique must be used.

F. Encoding Technique

To adopt the shortest path problem to particle swarm optimization, every particle has a priority array. The priority array contains priority values of the nodes to be used for finding the shortest path. The values of the priority array is chosen randomly at first. When a particle tries a randomly created path, according to the score of the particle, the priority array is updated. The priorities of the nodes representing the shortest path solution are increased. Increment value is done using particle's speed value. The best path of the particle is represented by the particle's coordinate value. To find the shortest path, every particle tries the next possible node which has the highest priority. The priority values of priority array, speed value are randomly created. At the beginning, coordinate value (array representing the path) has only the starting node.

G. PSO Algorithm Parameters

The values of the parameters of the proposed algorithm are shown in Table I.

Table I. Parameters of the proposed algorithm

PSO Algorithm Parameters	
Number of birds	Obtained by the user
r_1	Random number between 0 and 1
r_2	Random number between 0 and 1
c_1	Random number between 0 and 1
c_2	$c_1 * 2$
Number of iterations	Obtained by the user
Speed	Random number between-0.5 and 0.5
Position	Vector presenting nodes

H. The Proposed Algorithms

Creation of particles can be seen in Algorithm 2.

Algorithm 2 Creation of particles
1: create a priority array same length as the node number;
2: fill the priority array with random values between determined minimum and maximum values;
3: set starting node and finishing node's priority to maximum;
4: create an array for the followed paths;
5: set speed of every particle for every node's priority;
6: set r1 to a value between 0.0 and 1;
7: set r2 to a value between 0.0 and 1;
8: set c1 to a value between 0.0 and 1;
9: set c2 to c1*2;

Shortest path calculations using particle swarm optimization is used from the work (17), which examines the solution of shortest problem using particle swarm optimization. Because the problem network is unique, some changes made while using this algorithm, the detailed flow of the proposed algorithm can be seen in Algorithm 3.

Algorithm 3 Algorithm for PSO collision avoidance

for i = 1 to numberofparticles **do**
  for j = 1 to numberofnodes **do**
   particle(i).speedarray(j)=Random number between −1 and +1;
  end for
  particle(i) c_1=Random number between 0 and 1;
  particle(i). c_2=particle(i).c1 ∗ 2;
  particle(i). r_1=Random number between 0 and 1;
  particle(i). r_2=Random number between 0 and 1;
  for j = 1 to numberofnodes **do**
  particle(i).priorityarray(j)=Random number between −100 and +100;
  end for
end for
define mostsuccesfullparticle with a high score to save best solutions;
for i = 1 to iterationnumber **do**
  for j = 1 to numberofparticles **do**
    particle(j).path=startingpoint;
    while particle(j).path(end)! = finishingpoint **do**
     find the node that is not in path array and has the maximum priority and add it to path array;
    end while
    if particle(j).score
      particle(j).bestscore=particle(j).score;
      particle(j).pBest=particle(j).priorityarray;
    end if
  end for
  calculate the best particle;
  if bestparticle.score
    mostsuccesfullparticle=bestparticle;
  end if
    update all particles' speeds and priority arrays according to equations (5)and(6);
end for

IV. RESULTS

A graphical user interface was developed to visualize the results of the proposed algorithm. By the interface the starting node, finishing node, particle number and iteration number can be selected. The graphical user interface can be seen in Fig. 2.

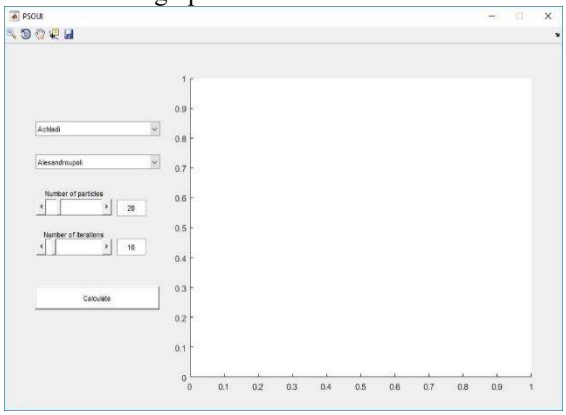

Fig. 2 Graphical User Interface for Calculation

For obtaining the success of the proposed algorithm, some example voyages carried out. In the first scenario, the shortest path from Achladi port to Alexandraoupoli port is calculated. Swarm size is set to 20 and the number of iterations set to 10.The result is shown on the map created, it can be seen of Fig. 3.

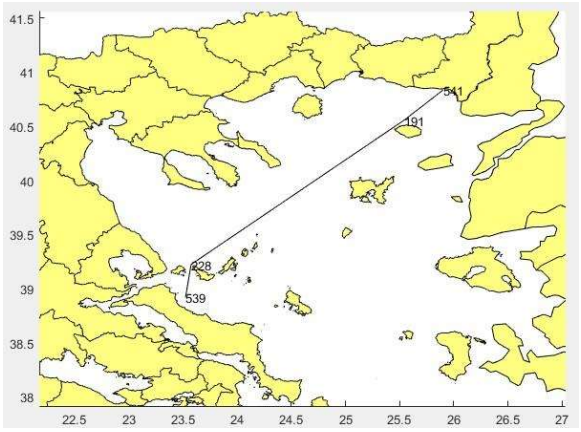

Fig. 3 The shortest path from Achladi to Alexandroupoli

The shortest path is found to be 539-228-191-541 nodes. 539 is the number of Achladi port which is the departure port. 541 is the number of Alexandraupoli port which is the destination port. The shortest path to the destination port found 269.6279 kilometers which is equal to 203.8618 nautical miles. In the second scenario, the shortest path from Gulluk port to Akra Andros port is calculated. 553 is the number of Gulluk port and 543 is the number of Andros port. Swarm size is set to 10 and iteration number is set to 40. The result can be seen in Fig. 4.

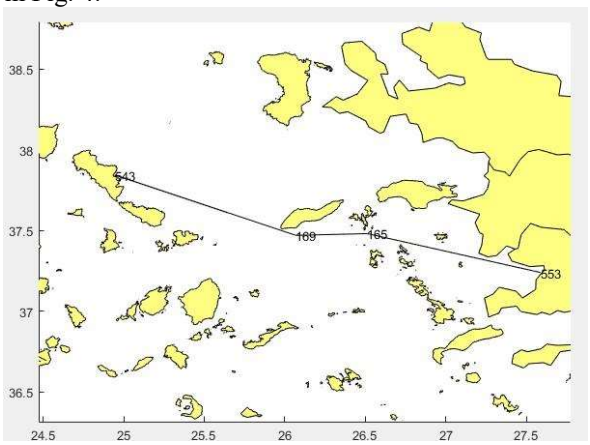

Fig. 4 The shortest path from Gulluk to Andros

The shortest path is found to be 553-165-169-543. The total distance to the destination port is found 244.9884 kilometers which is equal to 132.2832 nautical miles. In the third scenario, the shortest path from port of Kavala to port of Izmir is calculated. The swarm size is set to 10 and the iteration size is set to 70. The path is found to be 557-195-185-39-555 *.th* nodes. The result can be seen in Fig. 5.

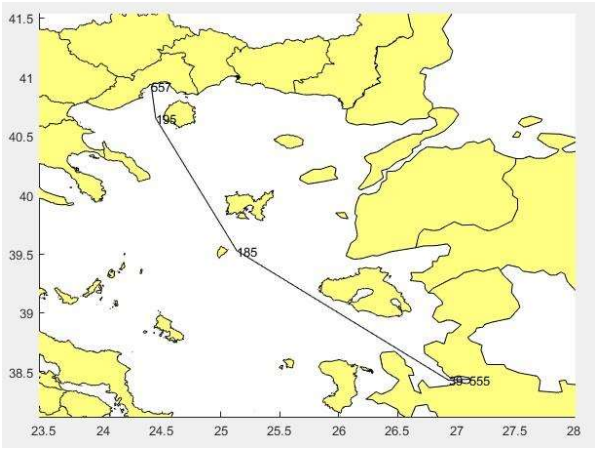

Fig. 5 : The shortest path from Kavala to Izmir

The shortest path was found to be 557-195-185-39-555. The total distance is found 377.5520 kilometers, it is equal to 203.8618 nautical miles.

V. DISCUSSION

The proposed system is prepared using MATLAB program using object oriented programming. The particles used for the calculation are objects which are member of the Particle class. The Particle class has speed, priority array, coordinate vector (array presenting the path), r_1,c_1,r_2,c_2 and pBest values. The calculations are made on a x64 computer which runs Windows operating system, the computer has an Intel Pentium 7 processor, 8 GB ram, running at 3,6 Ghz. In order to obtain the success of the proposed algorithm. The same scenarios examined using Dijkstra's and Bellman-Ford algorithm, the results are compared according to their distance values. The comparison can be seen in Fig. 6.

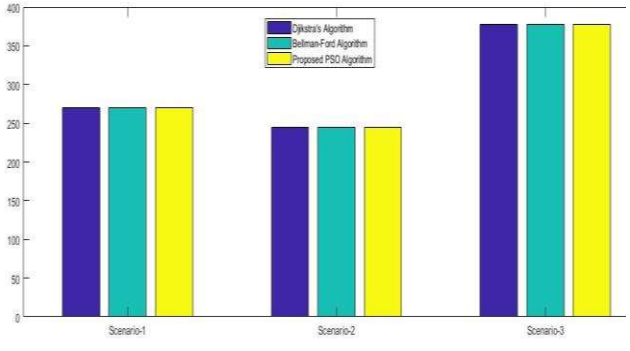

Fig. 6 Comparison of the proposed algorithm with two common algorithms by means of shortest distance.

As can be seen from Fig. 6. The three algorithms successfully found the shortest path result and the same nodes.

VI. CONCLUSION

In this study, a navigational shortest path problem was solved using particle swarm optimization. The network model is created using experiences of the author.The map is a shapefile that contains both side of the Aegean sea. When the solutions made with the algorithm are examined, it was observed that the solution was successful. Although the algorithm was slow compared to Dijkstra's and Bellman-Ford algorithms in terms of solution time, the system was considered highly usable, because the planning time doesn't have to be very fast. This study is a part of an intelligent system that includes route planning, collision avoidance and weather routing. In the previous works some other parameters (weather information, voyage changes, navigational warnings etc.) will be included.

REFERENCES

[1] Fossen, Thor I. Encyclopedia of Systems and Control, pp.1-9. Mathematical Models of Ships and Underwater Vehicles,2011.

[2] nomoioxxe. Prefectures HEMCO (Shapefile),25/12/2016. http://services.opendatagortynia.gr//geoserver// wfsrequestGetFeature servicewfs version1.0.0 typenamegortynia:NOMOI-OKXE.

[3] turkeyshapefile. European Environment Agency,26/12/2016. https://www.eea.europa.eu/data-andmaps/data/eea-reference-grids-2/ gis-files/turkey–shapefile.

[4] INAN, Timur and BABA, Ahmet Fevzi.(Turkish) 2018 Electric Electronics, Computer Science, Biomedical Engineerings' Meeting (EBBT).Examining the performance of ant colony optimization on shortest path problem: (Aegean sea example).2018

[5] INAN, Timur and BABA, Ahmet Fevzi.(Turkish) Marmara Fen Bilimleri Dergisi. Ege Denizinin Detaylı Bir Du¨g˘um¨ Haritasının Kullanılarak Genetik Algoritma ˙Ile En Kısa Yol Sonuc¸larının Elde Edilmesi.2018

[6] Koromila, Ioanna and Nivolianitou, Zoe and Giannakopoulos, Theodoros. Bayesian network to predict environmental risk of a possible ship accident. ACM International Conference Proceeding Series.2016

[7] Kosmas O.T.,Vlachos D.S. Simulated annealing for optimal ship routing. Computers and Operations Research.2012

[8] Veneti A.,Makrygiorgos A., Konstantopoulos c.,Pantziou G., Vetsikas I.A..Minimizing the fuel consumption and the risk in maritime transportation: A bi-objective weather routing approach.Computers and Operations Research.2017

[9] Kepaptsoglou K.,Fountas G.,Karlaftis M.G. Weather impact on containership routing in closed seas: A chanceconstraint optimization approach.Transportation Research Part C: Emerging Technologies.2015

[10] Nikolaos Alexandris and Chrysostomos Fountas and Aristidis Vlachos. The ant colony system: optimization for the logistics of marine cargo in the Aegean. Journal of Statistics and Management Systems.2005

[11] Zhuo, Yongqiang and Hearn, G.E. Chinese Control and Decision Conference. ,A Ship Based Intelligent AntiCollision Decision-Making Support System Utilizing Trial Manoeuvres,2008.

[12] Padhy C. P.,Sen D., Bhaskaran P. K. Application of wave model for weather routing of ships in the North Indian Ocean.Natural Hazards.2008

[13] Tsou, Ming-Cheng and Kao, Sheng-Long and Su, ChienMin. The Journal of Navigation. Decision Support from Genetic Algorithms for Ship Collision Avoidance Route Planning and Alerts,2010.

[14] Keivan Ghoseiri and Behnam Nadjari. An ant colony optimization algorithm for the bi-objective shortest path problem. Applied Soft Computing.2010.

[15] F. Araujo and B. Ribeiro and L. Rodrigues. IEEE Transactions on Neural Networks. A neural network for shortest path computation.2001

[16] J. Kennedy and R. Eberhart. Neural Networks. Particle swarm optimization, 1995.

[17] Ammar W. Mohemmed and Nirod Chandra Sahoo and Tan Kim Geok. Applied Soft Computing. Solving shortest path problem using particle swarm optimization,2008.

[18] Chang Wook Ahn and R. S. Ramakrishna. IEEE Transactions on Evolutionary Computation. A genetic algorithm for shortest path routing problem and the sizing of populations, 2002.

[19] E. W. Dijkstra (1959). A note on two problems in connexion with graphs. Numerische Mathematik. Vol.1. Pages. 269-271.

[20] Bellman, E., "On a Routing Problem", Appl.Math., Vol 16, 87–90, 1958

[21] Fossen, T. I. and T. Perez (2004). Marine Systems Simulator (MSS). ¡http://www.marinecontrol.org¿.