Ship Route Mining And Path Optimization In Hadoop

[1]Shabana M A, [2]Elizabeth Isaac

[1]Student, [2]Professor
Computer Science and Engineering,
Mar Athanasius College of Engineering, Kothamangalam, India

***Abstract*— Mining trajectory data has been attracting significant interest in the last years. By analyzing trajectory data, we are able to discover the movement behavior and location aware knowledge, and then develop many interesting applications such as movement behavior discovery, location prediction, traffic analysis, and so on. In the proposed system it provides a framework of ship route mining thereby discovering the shortest path between source and destinations also detect the ships moving in a different path which is not similar to the frequently taken path. The shortest path (SP) problem concerns with finding the shortest path from a specific origin to a specified destination in a given network while minimizing the total cost associated with the path. Particle swarm optimization is a population based stochastic optimization technique inspired by the social behavior of bird flock. The proposed system uses the PSO based search method to obtain a best route from source to destination. The PSO starts with a population of particles whose positions, that represent the potential solutions for the studied problem, and velocities are randomly initialized in the search space. The search for optimal position (solution) is performed by updating the particle velocities. The different path taken by the ships may be due some terrorist attack or due to weather conditions. The system is to generate the optimum ships routes and detect the ships showing different pattern in their route.**

***IndexTerms*— PSO (Particle Swarm Optimization), SP(Shortest Path)**

I. INTRODUCTION

Maritime transportation represents approximately 90% of global trade by volume, placing safety and security challenges as a high priority for nations across the globe. Maritime surveillance data are collected at different scales and are increasingly used to achieve higher levels of situational awareness.

Trajectory data mining is a challenge task because of the trajectory data is available with uncertainty. The uncertainty may be produced by the inaccuracy of positioning device and asynchronous sampling. With location uncertainty, a moving objects movement may not exactly repeat the same trajectory even the object has the similar movement behavior with others. Furthermore, discovering the valuable knowledge from maritime trajectory is made even more difficult due to the maritime area is a free moving space. Unlike the vehicles movements are constrained by road networks, there is no such a sea route for ships to follow in maritime area. Obviously, the maritime trajectory is moving free and the data is more complex than the trajectories moving along the road network. A ships movement may not exactly repeat the same trajectory even the ship has the similar movement behavior with others. Thus, mining the maritime routes where ships frequently navigate from collected ship trajectories is a challenging problem.

The shortest path (SP) problem concerns with finding the shortest path from a specific origin to a specified destination in a given network while minimizing the total cost associated with the path. This problem has widespread applications. Some important applications of the SP problem include vehicle routing in transportation systems, traffic routing in communication networks and path planning in robotic systems.

The SP problem has been investigated extensively. The well-known algorithms for solving this problem include the Bellmans dynamic programming algorithm for directed networks, the Dijkstra labeling algorithm and Bellman Ford successive approximation algorithm for networks with non negative cost coefficients. These traditional algorithms have major shortcomings; firstly, they are not suitable for networks with negative weights of the edges. Secondly, the algorithms search only for the shortest route, but they cannot determine any other similar or non-similar short routes. Thirdly, they exhibit high computational complexity for real-time communications.

The proposed system provides a framework of ship route mining thereby analyze the huge volume of historical data of ship trajectories further the movement behavior and find the shortest path between source and destination. As the data to be analyzed is very large Bigdata processing techniques are using to process the data. So MapReduce concept in Hadoop framework is used. Particle Swarm Optimization algorithm is used to find the shortest path. The most attractive feature of PSO is that it requires less computational bookkeeping and, generally, a few lines of implementation codes. The purpose of our approach is to investigate on the applicability and efficiency of PSO for this SP problem.

II. LITERATURE SURVEY

In 1998, Masaharu Munetomo, Yoshiaki Takai, and Yoshiharu Sato introduced 'A Migration Scheme for the Genetic Adaptive Routing Algorithm'. Path genetic operators (path mutation and path crossover) are designed for the network routing algorithms to generate alternative routes in routing tables. The operators are constrained to topology of the target network. The path mutation operator mutates a route to create an alternative route. The path crossover operator exchanges sub routes between a pair of routes. These operators create alternative routes in a routing table to find an optimal route which minimizes communication latency by

employing together selection operators based on fitness values which are calculated from communication latency. It can also balance the load of links by distributing packets among the alternative routes.

In 2013 Pallotta G., Vespe M., and Bryan K introduced 'Vessel Pattern Knowledge Discovery from AIS Data: A Framework for Anomaly Detection and Route Prediction'. It presents an unsupervised and incremental learning approach to the extraction of maritime movement patterns. The proposed methodology is called TREAD, which stands for Traffic Route Extraction and Anomaly Detection. TREAD converts raw data, i.e. ship position reports from different tracking platforms, into information that can be used to support decisions concerning the safety and security of shipping. The paper shows that understanding past maritime traffic patterns is a fundamental step towards Maritime Situation Awareness applications, in particular, to classify and predict activities. TREAD is a basis for automatically detecting anomalies, using past ship tracks and traffic patterns as an input to a Decision Support System. TREAD builds a statistical model in which the traffic knowledge is extracted from the data by means of ship objects, created and constantly updated based on the AIS position data stream. The changes in the state vectors, i.e. the course and speed, of many ship objects generate a series of spatial events that are clustered around waypoints used to reconstruct the traffic routes. Tracks that substantially deviate from other vessel paths on the same route are considered outliers and eliminated from the analysis. The result of the data analysis is fed into the last module of TREAD which provides the anomaly detection and route prediction functions.

The problem of reconstructing shipping lanes in a particular area is presented by Fernandez Arguedas in 2014. The proposed algorithm automatically produces a network of maritime shipping lanes extracted from historical vessel positioning data, by detecting the entry and exit 7 points in the ocean region and the so called breakpoints which divide a ship track into shorter segments. The proposed applications are track reconstruction in cases of tracking gaps, destination prediction, and detection of anomalous behavior.

In 2014 'RouteMiner: Mining Ship Routes from a Massive Maritime Trajectories' was introduced by Yu-Ting We, Chien-Hsiang La, Po-Ruey Lei and Wen-Chih Peng. The objective of the system is to discover the ships movement patterns hidden in their historical trajectories, and then detect the ship route. More specifically, it not only discover the movement pattern but also de_ne and detect the movement area of ship route in a free moving space. it propose RouteMiner to provide a framework for ship route mining. Given a set of ship trajectory in a monitoring area, RouteMiner explore the movement pattern from massive ship trajectories in a free moving space. Then, ship routes are defined and detected based on those behavioral pattern. Finally, the system generates a set of ship routes.

Later in 2017, 'Distributed Document Clustering Analysis Based on a Hybrid Method' was introduced by J.E. Judith, J. Jayakumari. It provides an overview of how Particle Swarm Optimization can be implemented in MapReduce Framework. PSO is used to take advantage of its global search ability to provide optimal centroids which aids in generating more compact clusters with improved accuracy. In order to support data intensive distributed applications, an open source implementation based on Hadoop is used for processing of large datasets. MapReduce is a functional programming model
for distributed processing over several machines. The important idea behind MapReduce framework is to map the datasets into a group of <key, value> pairs, and then reduce all pairs with the same key. A map function is performed by each machine which takes a part of the input and maps it to <key, value> pairs. This is then send to a machine which applies the reduce function. The reduce function combines the result for further processing. The outputs from the reduce function are then fed into the appropriate map function to begin the next round of processing.

III. PROPOSED SYSTEM

The proposed system provides a framework of ship route mining thereby discovering the shortest path between source and destinations using PSO. The basic architecture includes mainly two modules, Ship route mining and finding optimized path. The Maritime Trajectory Dataset is given to the Ship Route Mining module. Where the data is preprocessed and ship routes are detected and analyzed the patterns of ship routes. A number of such ship routes between a source and destination is given to the next module to find the optimized path among them. In this module of optimization, Particle Swarm Optimization algorithm is used. The result from this module is the output of the system.

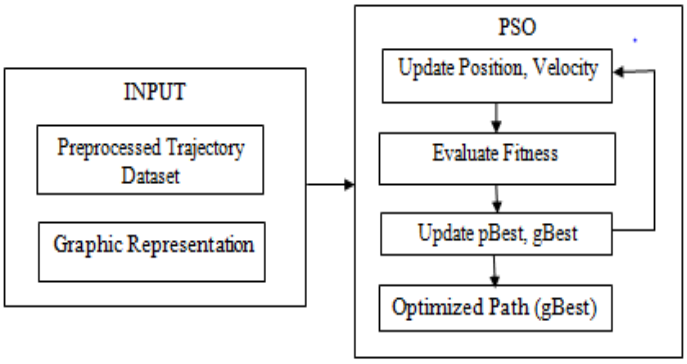

Figure 1: Architecture

Preprocessing

Input data consist of ship_id, time, latitude, longitude. Structure of the input data can be represented as; H, US, 41457, 36.000, -9.800, 1970, 8, 10, 11.12, 2147316, 4.

Pre-processing is done to convert this data in to a graph which contains
1) Starting node and end node.
2) A number of intermediate nodes.;
3) A number of links which forms a number of paths.

The pre-processing step mainly consists of three MapReduce Jobs. The first MapReduce job is to separate the whole database based on the ship_id. The output of this MapReduce job is a number of files each file Consist of a number of paths travelled by a single ship. A path is consist of a number of nodes connected together and each node is a location through which the ship is travelled. This location is a Latitude, Longitude pair. Second MapReduce job is to generate unique ids for each such location. Third MapReduce job is to generate links between the locations.

Particle Swarm Optimization

Particle swarm optimization (PSO) is a population based optimization technique inspired by the social behavior of bird flock. The algorithmic flow in PSO starts with a population of particles whose positions, that represent the potential solutions for the studied problem, and velocities are randomly initialized in the search space. The search for optimal position (solution) is performed by updating the particle velocities, hence positions, in each iteration/generation in a specific manner as follows.

In every iteration, the fitness of each particles position is determined by some defined fitness measure and the velocity of each particle is updated by keeping track of two best positions. The first one is the best position (solution) a particle has traversed so far. This value is called pBest. Another best value is the best position (solution) that any neighbor of a particle has traversed so far. This best value is a neighborhood best and is called nBest. When a particle takes the whole population as its neighborhood, the neighborhood best becomes the global best and is accordingly called gBest. A particles velocity and position are updated as follows.

$V_{id} = v_{id} + c_1 r_1 (b_{id} * x_{id}) + c_2 r_2 (b^n_{id} * x_{id})$ (1)

$i = 1; 2; :::; N_s$

$d = 1; 2; :::; D$

$X_{id} = x_{id} + v_{id}$ (2)

Where c_1 and c_2 are positive constants, called acceleration coefficients, Ns is the total number of particles in the swarm, D is the dimension of problem search space, i.e., number of parameters of the function being optimized, r_1 and r_2 are two independently generated random numbers in the range [0,1] and n represents the index of the best particle in the neighborhood of a particle. The other vectors are defined as: $x_i = [x_{i1}, x_{i2}, . . ., x_{iD}]$ is the position of ith particle; $v_i = [v_{i1}; v_{i2}; . . . ; v_{iD}]$ is the velocity of ith particle; $b_i = [b_{i1}, b_{i2}, . . ., b_{iD}]$ is the best position of the ith particle (pBesti), and $b^n_i = [b^n_{i1}, b^n_{i2}, . . . ; b^n_{iD}]$ is the best position found by the neighborhood of the particle i (nBesti). The pseudo-codes for general algorithmic flow of PSO are listed bellow.

```
Initialize the position and velocity of each particle in the population randomly.
Calculate fitness value of each particle.
Calculate pBest and nBest for each particle.
Do
  Update velocity of each particle.
  Update position of each particle.
  Calculate fitness value of each particle.
  Update pBest for each particle if its current fitness value is better than pBest.
  Update nBest for each particle, i.e, choose the particle with the best fitness value among all the neighbors as the nBest for
        a specific neighborhood topology.
While termination criterion is not attained.
```

Equation (1) calculates a new velocity for each particle based on its previous velocity, the particles position at which the best possible fitness has been achieved so far, and the neighbors best position achieved. Equation (2) updates each particles position in the solution hyperspace.c1 and c2 are two learning factors, which control the influence of pBest and nBest on the search process.
Figure 2 shows an example of indirect scheme for path representation from node 1 to node 20. The path construction starting from node 1 is performed as follows. From the node adjacency relations, the node with highest priority, i.e., node 3 (priority value = 60), is selected to be included in path, out of the nodes 2, 3, 4 and 5 (possible nonvisited nodes to be visited from node 1). Then, out of the possible nonvisited nodes that can be visited from node 3, node 8 is selected because of its highest priority and is put into the path. These steps will be repeated until a complete path 1, 3, 8, 14, 20 is obtained, as per the represented priority values shown in Figure 3 (b). There may be situations when the path does not terminate at the destination node leading to an invalid path. This situation is illustrated in Figure 3 (c). In this case, the partial path takes the node sequence 1, 3, 8, 7, 6, 2. From node 2, the partial path should not be allowed to grow further as it will result in a loop (the solution to the SP problem must not include any loop).

In this sense, such types of nodes (node 2) can be called as no-exit nodes. Therefore, the possibility of invalid path generation (due to such no-exit nodes) is high resulting in substantial fruitless computational effort.
The position vector of a particle in PSO is represented by a priority vector of the type shown in Figure 3 (a) along with the following incorporated features.

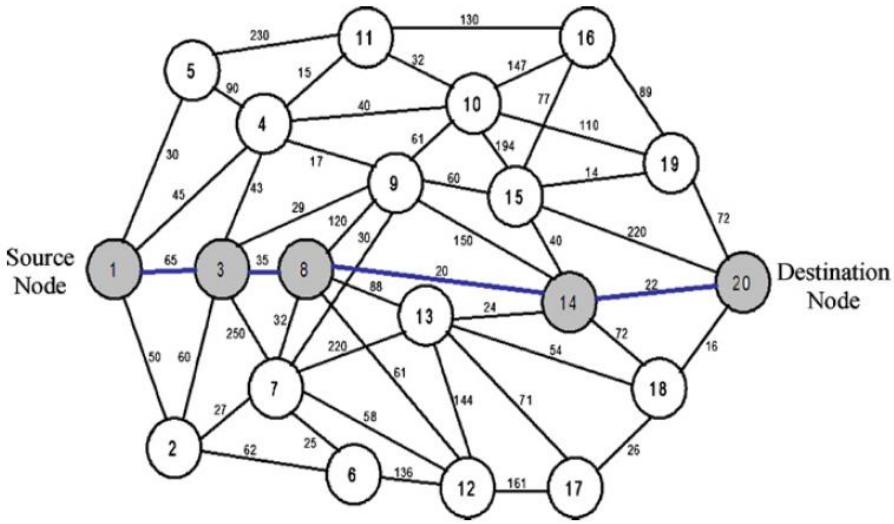

Figure 2: Sample Network

Node ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Node Priority	p1	p2	p3	p4																p20

(a) Encoding scheme

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	10	60	12	17	34	19	56	13	6	41	39	5	55	18	29	23	25	27	51

(b) Particle leads to a valid path: 1→3→8→14→20

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	65	70	12	17	67	61	76	13	6	41	39	5	55	18	29	23	25	27	51

(c) Particle leads to an invalid path: 1 → 3→ 8→ 7→6→2

Figure 3: Path Encoding

The nodes are allowed to take both positive and negative priority values of any magnitude. The node that is already included in a growing path will be assigned a large negative priority value (for example, $-N_\alpha = -50000$); thus that node is highly unlikely to selected again while the PSO runs for a finite number of iterations. To reduce the possibility of building a backward path (hence, possible loop formation) and also, simultaneously keeping some room for any potential backward movement, a heuristic operator is incorporated as follows. A node is selected as to be next node in the growing path if its ID is larger than the present node ID by a certain specified value, if

ID of next node - ID of present node > =M (3)

Where M is a positive integer.

Let N_{max} be the maximum number of nodes in the network Let V be a partial path (corresponding to the position/priority V^p_k vector of a particle) under growth, which contains k + 1 nodes with the terminal node t^k (k = 0 corresponds to the partial path with source node only), and $(k + 1)^{th}$ node is to be selected. Let x^k be the dynamic priority vector, which initially contains the priority values (position vector of the particle), referred to by x. Every time a node is added to the partial path, the corresponding position in x^k is given a large negative value (-N_) as explained earlier. Without loss of generality, node number 1 is taken as source node and destination node ID is N_{max}. The implementations of the modified priority-based encoding along with gradual path construction process are summarized in the following steps:

Step 1: (Initialization) Let k = 0, $V^p_k = \{1\}$ and $x^k = x$; $t^k = 1$ and $x^k(t^k) = -N\alpha$.

Step 2: (Termination test) If $t^k = N_{max}$, or $k > N_{max}$, go to Step 4; else k = k + 1 and go to Step 3.

Step 3: (Path extension) Select node t^k as the node with highest priority from among the nodes having direct links with node t^{k-1} if $(t^k - t^{k-1}) > -M$. Set $V^p_k = \{V^p_{k-1}, t^k\}$ and $x^k(t^k) = -N\alpha$.

Step 4: (Complete path) Return complete valid path V^p_k or return invalid path V^p_k if the terminal node is not the destination node.

The quality of a particle (solution) is measured by a fitness function. Here, the fitness function is obvious as the goal is to find the minimal cost path. Thus, the fitness of ith particle is defined as:

$$f_i = \left(\sum_{j=1}^{N_i - 1} C_{yz} \right)^{-1}, \; y = \mathbf{PP}^i(j) \quad \text{and} \quad z = \mathbf{PP}^i(j+1)$$

Where PP^i is the set of sequential node IDs for ith particle, $N_i = |PP^i|$ = number of nodes that constitute the path represented by i^{th} particle, and C_{yz} is the cost of the link connecting node y and node z. Thus, the fitness function takes maximum value when the shortest path is obtained. If the path represented by a particle happens to be an invalid path, then its fitness is assigned a penalty value (=0) so that the particle's attributes will not be considered by others for future search.

PSO In Hadoop

This proposed methodology utilizes Hadoop and MapReduce framework which provides distributed storage and analysis to support data intensive distributed applications.

In the proposed system the preprocessed trajectory dataset consist of a large number of paths from a source to a destination. In order to find the shortest optimized path from this large set of paths we need a large number of traversals and iterations. Doing this process in sequential manner is very complex and takes an infinite time to complete the whole process. So it is essential to divide this whole work in to a number of tasks and has to distribute these tasks to carry out the execution in a parallelized manner. In this proposed work, PSO is implemented in Hadoop for better speedup and accuracy of optimization.

The input to the Map function includes a set of particles (a set of possible paths from source to destination) stored in HDFS. Map function read each particle in the swarm and generate key as ship id and values as the corresponding particles. The reducer function calculates the velocity, fitness value, change the pBest and gBest and generate the global best score.

```
Procedure Mapper
For each particle in the swarm.
{
Generate (key, value) pairs where key=ship id and value= the particle.
}
Procedure Reducer
calculate pBest and nBest for each particle.
Do
   Calculate velocity of each particle.
   Update position of each particle using Equation (2).
   Evaluate fitness value of each particle.
   Update pBest for each particle if its current fitness value is better than pBest.
   Update nBest for each particle, i.e, choose the particle with the best fitness value among all the neighbors as the nBest for
         a specific neighborhood topology.
While there is more (key, value) pairs in HDFS.
```

The ship_id represents the key and the corresponding content as the value. The collection of (key, value) pairs contained in files is referred to as blocks. The pseudo code for the map and reduce function is given above. The particle swarm is retrieved from the distributed storage. For each particle the map function generates (key, value) pairs. Where key is the ship id and corresponding value is the particle itself. The reduce function reads each (key, value) pairs which are generated by the map function and calculates the velocity, fitness value, change the pBest and gBest and generate the global best score.

IV. PERFORMANCE ANALYSIS

Standard PSO Versus PSO In Hadoop

The proposed PSO based algorithm for SP search in Hadoop is tested on a number of networks having different number nodes as well as links. To compare the performance of the proposed PSO in Hadoop algorithm with the standard PSO, different network topologies of (5–20) nodes with randomly assigned links are generated. Figure 4 shows the comparison of the proposed system with the standard PSO by comparing their average time required to execute, to achieve the results with network topologies in the order of increasing in number of links. The number of particles generated is set as 100 in both the cases.

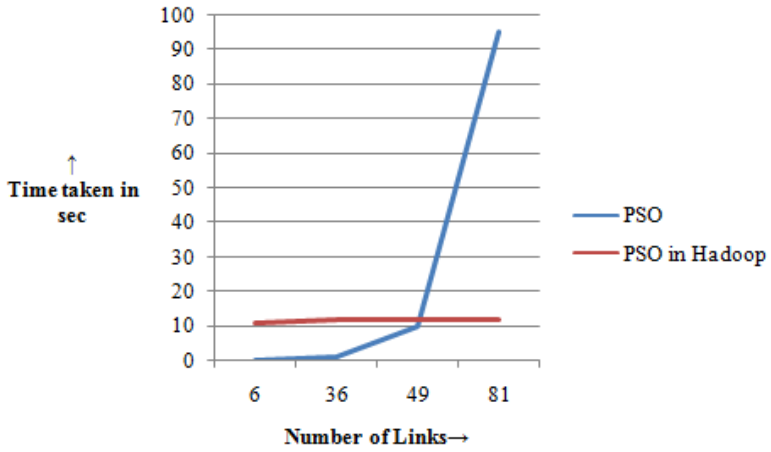

Figure 4: PSO v/s PSO in Hadoop

The figure shows that as the number of nodes as well as links increases in the input network topology then the time required to process the network to find the shortest path will also increases. But the rate of increase in the execution time is very large in the standered PSO when compared with that of PSO in Hadoop.
It clearly illustrates that the quality of solution and time efficiency obtained with PSO in Hadoop algorithm are higher than those of standard PSO search.

20 nodes Versus 4038 nodes dataset

The proposed PSO in Hadoop is tested with a dataset having size of 1.8 mb which consists of 4038 nodes and more than 90000 links. Two types of execution times are recorded one is the time for path construction and another one is the time for optimization.

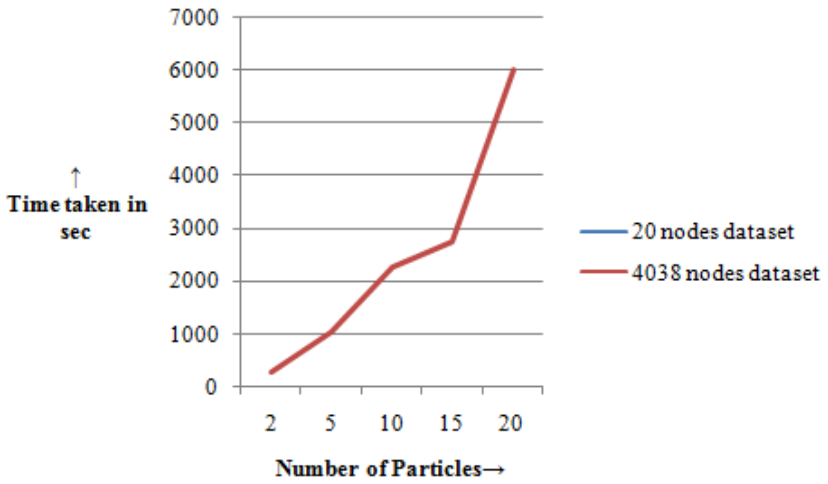

Figure 5: Time for path construction

The system is tested several times by increasing the number of particles generated. The results obtained are compared with a simple input dataset which is having only 20 nodes. Figure 5 shows the performance evaluation of the PSO in Hadoop while considering the time for path construction. It clearly illustartes that as the dataset size increases it becomes more difficult to process data and it takes more time for constructing the path from source node 0 to destination node 4038. Also when the number of particles increases time also increases.

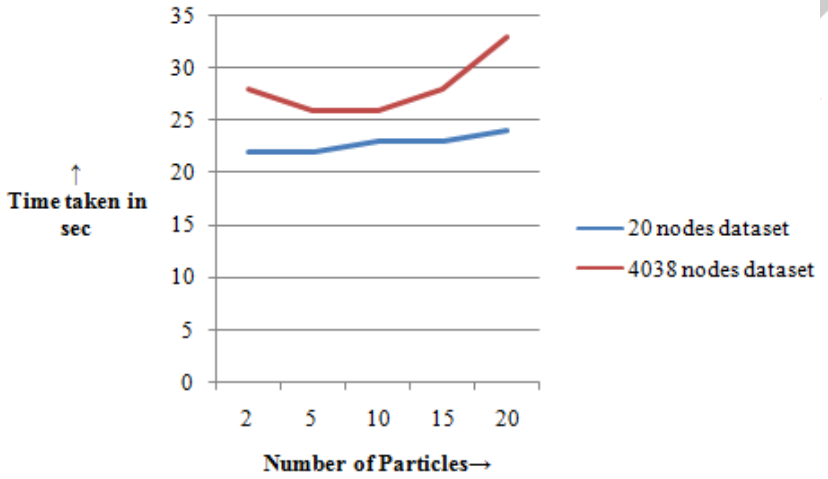

Figure 6: Time for Optimization

Figure 6 shows the performance evaluation of the PSO in Hadoop while considering the time for path optimization. It illustartes that as the dataset size increases there is no such a large difference in the path optimization time of 4038 nodes dataset as compared to 20 nodes dataset.

Considering the above two graphs it can be concluded that major portion of the execution time is spending in the stage of path construction. The path optimization needs very less time to execute.

V. CONCLUSION

In this method, a PSO in Hadoop algorithm is presented and tested for solving the shortest-path problem in networks of ships. The performance of the PSO in Hadoop has been evaluated by tests on different random networks with varying number of nodes and links. Evaluation results shows that PSO in Hadoop will take less time to process networks having large number of links when compared to standard PSO. Also the proposed PSO in Hadoop is tested with a dataset having size of 1.8 mb which consists of 4038 nodes and more than 90000 links. The performance is evaluated by comparing the path construction as well as optimization time

with that of a simple 20 nodes graph. It clearly illustrates that the quality of solution and time efficiency obtained with PSO in Hadoop is much better than other cases.

REFERENCES

[1] Ammar W. Mohemmed, Nirod Chandra Sahoo, Solving shortest path problem using particle swarm optimization, Applied Soft Computing, 2008.
[2] C.W. Ahn, R.S. Ramakrishna, A genetic algorithm for shortest path routing problem and the sizing of populations, IEEE Trans. Evol. Comput. 2002.
[3] F. Araujo, B. Ribeiro, L. Rodrigues, A neural network for shortest path computation, IEEE Trans. Neural Netw, 2001.
[4] Giuliana Pallotta, Michele Vespe and Karna Bryan, Vessel Pattern Knowledge Discovery from AIS Data: A Framework for Anomaly Detection and Route Prediction, Entropy 2013.
[5] J.E. Judith, J. Jayakumari, Distributed Document Clustering Analysis Based on a Hybrid Method, China Communications , February 2017.
[6] M.K. Ali, F. Kamoun, Neural networks for shortest path computation and routing in computer networks, IEEE Trans. Neural Netw. 1993.
[7] M. Munemoto, Y. Takai, Y. Sato, A migration scheme for the genetic adaptive routing algorithm, in: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, 1998.
[8] Yu-Ting Wen, Chien-Hsiang Lai, Po-Ruey Lei and Wen-Chih Peng, RouteMiner: Mining Ship Routes from a Massive Maritime Trajectories, IEEE 15th International Conference on Mobile Data Management, 2014.
[9] Ying-Tung Hsiao, Cheng-Long Chnang, and Cheng-Chih Chien, Ant Colony Optimization for Best Path Planning, lnternational Symposium on Communications and Information Technologies, 2004.
[10] La Spezia, Contextual Anomalous Destination Detection For Maritime Surveillance, Maritime Knowledge Discovery and Anomaly Detection Workshop, 2016.
[11] Mark P. Wachowiak, Member, IEEE, Mitchell C. Timson, and David J. DuVal, Adaptive Particle Swarm Optimization with Heterogeneous Multicore Parallelism and GPU Acceleration, IEEE Transactions, 2016.
[12] Qisheng Cai, Teng Long, Zhu Wang, Yonglu Wen, Jiaxun Kou, Multiple paths planning for UAVs using particle swarm optimization with sequential niche technique, IEEE Transactions, 2016.
[13] Giuliana Pallotta, Michele Vespe and Karna Bryan, Vessel Pattern Knowledge Discovery from AIS Data: A Framework for Anomaly Detection and Route Prediction, entropy ISSN 1099-4300, 2013.
[14] Weather Routing, Principles Of Weather Routing, National Imagery And Mapping Agency.