Indian Journal of Fundamental and Applied Life Sciences ISSN: 2231– 6345 (Online)
An Open Access, Online International Journal Available at www.cibtech.org/sp.ed/jls/2014/04/jls.htm
2014 Vol. 4 (S4), pp. 1474-1491/Ghaemi et al.
Research Article

EVALUATION OF EFFICIENCY OF NEW COMBINED PSO-EO ALGORITHM IN SOLVING SAILOR ASSIGNMENT PROBLEM

***Reza Ghaemi[1], Kazem Shekofteh S.[2], Saeed Khakmardan[3], Hanie Poostchi[4] and Mahboub Farimani R.[5]**

[1]*Department of Computer Engineering, Quchan Branch, Islamic Azad University, Quchan, Iran*
[2]*Young Researchers and Elite Club, Quchan Branch, Islamic Azad University, Quchan, Iran*
[3]*Islamic Azad University, Mashhad Branch, Mashhad, Iran*
[4]*Ferdowsi University of Mashhad, Mashhad, Iran*
[5]*Department of Computer Engineering, Ferdows Higher Education Institute, Mashhad, Iran*
**Author for Correspondence*

ABSTRACT

The sailor assignment problem is a classic example of the generalized task assignment problem. In this instance, it is tried to assign the optimum tasks to the sailors in various time intervals according to their skills, experience and their location with different workloads. This problem has numerous applications in real world however due to many constraints and targets; the solution is not straightforward and requires intelligent algorithms. There are a few previous studies in on this problem that typically have used Genetic Algorithm and its Multi-objectives methods. In this study, we are using a powerful combined PSO-EO method to solve this complicated problem. In this method, the discovery ability of PSO along with extraction ability of EO is combined to find the optimum Pareto front of this complicated problem in the least possible time. To evaluate the efficiency of e proposed method, Pareto front, execution time, frequency of diversity of the samples and extent of search coverage have compared with those of NSGA-II algorithm and Kuhn-Munkres base algorithm. This comparison confirms the efficiency of the proposed method.

Keywords: *Sailor Assignment, NSGA-II, Genetic, PSO, Scheduling, Optimization*

INTRODUCTION

The general assignment problem (GAP) was first introduced as a scheduling problem for parallel processing machines with costs as following by Shmoys and Tardos (1993): Given a set of boxes and a set of products with different sizes and value relative to selected box, the objective is to package the products in the boxes in order to maximize the value of packaged products.

This problem could be also defined as a task assignment as follows:

Problem: assume we have n agent and m tasks to be done by the agents considering that each agent has various restrictions in power, time and skill. Therefore, assigning task to each agent requires cost and using part of the agent resource.

Objective: To find an optimum solution to assign tasks to agents such that not only satisfies agent conditions but minimize the total cost incurred

For a mathematical modeling of this problem, assume b_i is the budget for agent i, R_{ij} is the resource and C_{ij} is the cost to assign the task j to agent i. In this case GAP can be expressed an integer linear problem:

$$\text{Minimize} \quad \sum_{i=1}^{n}\sum_{j=1}^{m} C_{ij}x_{ij}$$

Subject to:

$$\sum_{j=1}^{m} x_{ij} R_{ij} \le b_i \qquad \forall i \in \{1,2,\ldots,n\}$$

$$\sum_{j=1}^{m} x_{ij} \le 1 \qquad \forall i \in \{1,2,\ldots,n\}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad \forall j \in \{1,2,\ldots,m\}$$

Where $x_{ij} \in \{0,1\} \qquad \forall i \in \{1,2,\ldots,n\}, \forall j \in \{1,2,\ldots,m\}$.

Task based sailor assignment can be viewed as a fine-grained instance of this problem in which it is expected to perform a joint task to be done by more than one sailor in a specific day considering limited number of sailors. Therefore, each day is divided to work shifts such that sailors can tackle the various tasks in different shifts. The constraints of the problem are that each sailor can only do one task at a specific shift. In fact, instead of assigning one task to a sailor, each sailor has to perform a set of tasks which have been distributed in certain time intervals. As a result, by dividing each day to work shift, the sailors shall do the required tasks at each shift.

For a practical solution of the problem, the strength and weakness of each sailor for varies tasks should be evaluated in order to assign different weights to each sailors for each task. The sailor weighted task assignment would be consists of several parameters such as their scores to do each task and their training to perform each task.

After the sailor team was determined each sailor should rate the tasks that have been assigned to him/her. The final set of assignments shall be according to navy regulations also to incur the least cost for the fleet. Obviously, an automatic system for the scheduling problem would play an important role in efficient and swift operation of the Navy.

The sailor task based assignment problem can be defined as following: Let n sailors, m tasks, and t time intervals. Each sailor can do any task in any time interval considering his skilland resources that will be used to the tasks. Therefore, the goal is to assign tasks to the sailors in all of time intervals such that minimize the number of deployed sailors and to maximize the primary objectives of the problem such as training, priority of sailors, preference of commanders and inverse of cost.

In addition the following constraints shall be included in TSAP:

- Total used resources in each assignment in each interval shall be more than resource capacity
- Each sailor cannot do more than one task in any interval
- Number of sailors assigned in each interval should be equal to number of tasks assigned in that interval

Assume cap_i is the capacity of sailor i, R_{ij} resource and C_{ij} cost of task assignment j to agent i. Also, y_{jk} is number of required tasks from class j in interval k. In this case TSAP is defined as a linear integer multi dimension problem.

$$\text{minimize} \qquad F(x) = (f_1(x),\ldots, f_N(x))$$

In which $x = (x_{ijk})_{i,j,k}$; $f_{obj} \quad \forall obj = 1,\ldots,N$ are objectives that should be minimized.

$$f_{obj}(x) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{t} x_{ijk} C_{ij}^{obj}$$

In the above equation C_{ij}^{obj} is the cost of task assignment j to sailor I in time interval k considering objective obj. F(x) consists of the following constraints:

$$\sum_{j=1}^{m} x_{ijk} R_{ij} \le cap_i \qquad \forall i \in \{1,2,...,n\}, \forall k \in \{1,2,...,t\}$$

$$\sum_{j=1}^{m} x_{ijk} \le 1 \qquad \forall i \in \{1,2,...,n\}, \forall k \in \{1,2,...,t\}$$

$$\sum_{i=1}^{n} x_{ijk} = y_{jk} \qquad \forall j \in \{1,2,...,m\}, \forall k \in \{1,2,...,t\}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i \in \{1,2,...,n\}, \forall j \in \{1,2,...,m\}, \forall k \in \{1,2,...,t\}$$

In summary, the sailor task assignment problem is intended to do the best task assignment for the sailors which maximize satisfaction criteria for both sailors and Navy. There is different number of assignment versus number of sailors in any season. There are typically a few hundred sailors in low activity seasons. However, the number of sailors increases to thousands in busier seasons.

In addition to extent of the problem and its size, the satisfaction of assignments does not evaluated only by one objective. Therefore, we need a combination of multiple criteria. Navy would like to maximize the sailor's satisfaction and employ the best talents with the lowest cost.

Considering these two targets, the problem becomes more challenging which requires high volume of computations and has many constraints and objectives. Therefore, the solution demands intelligent algorithm. Also, since GAP problem is NP-hard then sailor task assignment is not essentially tractable.

It is worth mentioning that considering the numerous application of task assignment in real world, solution to the sailor task assignment as a complicated instance of them could lead to a more general solution for these types of problem.

In the next section of this study, a review of previous works and algorithms used to solve this problem. In section 3, the proposed model for solving the problem and algorithms is addressed. In section 4 the result of the proposed model is compared with other works and the conclusion will be in section 5.

1- Related Works

The sailor task assignment is a classic problem which tries to assign sailors to various jobs in different time intervals considering their skills and experience. This is an important problem in the Navy with 300,000 sailors and 120,000 staff with job rotation annually.

According to Navy's policy, sailor's jobs should be changed every three years so that both sailors and Navy are satisfied.

In the past a manual method was used to tackle the problem. First sailors information including their skills and interests were recorded.

Then a group of specialists would review all the tasks and assign a list of permitted tasks to each sailor whcih had to select one of them.

After recveing the sailors feedbacks, the final assignemtn would be made if there were no conflicts. This procedure is shown in Figure 1. It is evident that the manual procedure is not practical with hig volume of data (Garrett *et al.*, 2005).

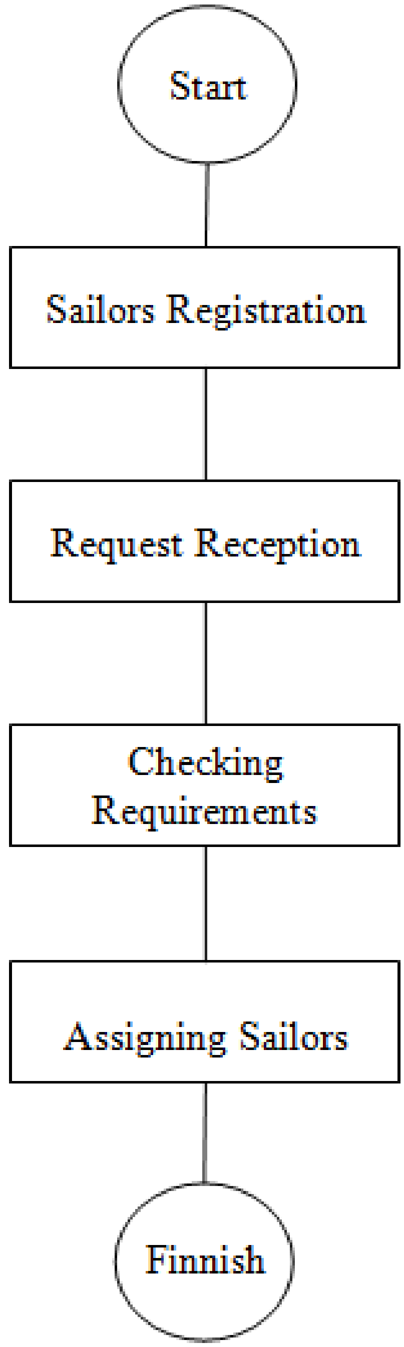

Figure 1: The procedure of task assignment

1-1- Kuhn-Munkres Algorithm

The first algorithm was introduced for the sailor task assignment problem was Kuhn-Munkres which is also known as Hungarian algorithm (Kuhn, 1955). This algorithm is a classic method to solve a linear assignment problem. In a linear selection problem, n number of tasks are assigned to n number of personnel so that to minimize the total costs incurred by the assignees. Moreover it is assumed that every personnel are capable of doing each task. In this case there is only one objective and number of jobs and staff are equal. Thus, the allocation problem can be considered as a two part graph as shown in Figure 2.

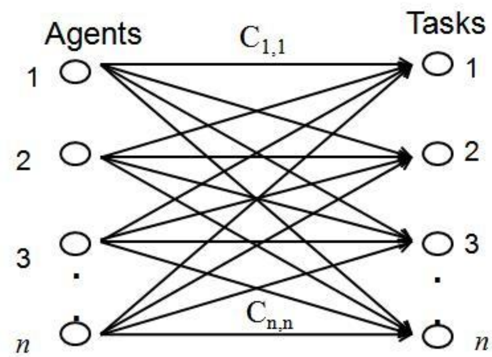

Figure 2: Linear assignment problem as a complete bipartite graph

1-2- Single (GA) and Multi Objectives (NSGA-II, SPEA2) Evolutionary Algorithm

Since the sailor assignment problem is not completely a two part graph and violates Kuhn-Munkres assumptions, many studies have employed multi-objectives evolutionary algorithm. In (Garrett *et al.*, 2005) a genetics algorithm was used to solve the problem. To use evolutionary methods develop a proper coding that doesn't violate the constraints is necessary. In this study integer numbers representation is used. In this representation each chromosome is related to a sailor and integer value of the chromosome indicates the task assigned to the sailor. To avoid assigning irrelevant tasks to the sailors, each chromosome is related to the tasks permitted to the sailor. With this representation each individual in the society is representative of one permitted task.

But in this representation it is possible to assign a task to several sailors that are able to do hat task. To avoid this problem each person is studied for similar task in a pre processign stage. If one task has been assigned to two persons then another permitted task will be assigned to one of them and if there is no task to assign, -1 task is allocated to this person. Therfore, reseanoale reuslts can be acheived by using propoer function that covers all the cases. The problem is that after each cross and mutation individuals shall be checked and restored. In addition in this study several genetics algorithm along with cross and mutation methods has been investigated in order to identify thee best parameters to sovel the proble using genetic algorithm in heuristic approach.

In (Deb *et al.*, 2002) and (Zitzler *et al.*, 2001) two evolutionary algorithm NSGA-II and SPAE-2 has been used consecutively to solve a multi-objective problem which did not converge due inadequate information about input. As a result in reference (Dasgupta *et al.*, 2008) algorithm NSGA-II was utilized along with initialization using Kuhn-Munkres algorithm. The correct initialization of the sample in evolutionary algorithm has yielded good results. In reference (Dasgupta *et al.*, 2008) three constraints are considered:

1 number of tasks is more than number of sailors

2 each sailor is only capable of doing the task he has trained for. Each sailor selects a set of the tasks and ranks them based on his interest.

3 sailor task assignments are a multi-objective problem which four objects shall be achieved simultaneously: maximize satisfaction of sailors (SR), commanders (CR), training scores (TS), and minimize relocation of sailors (PCS).

These three constraints would help multi objetive evolutionary algorithm to be supported with many solution in pareto front. Two more multi objective evolutionary algorithm NSGA-II and SPAE2 which are typically used to solve this problem is briefly discussed in the following section.

1-2-1- SPEA2 Algorithm

This algorithm has been developed in response to shortcomings of SPEA algorithm. There are two specimen in SPEA algorithm, primary and archive pools. Initially the archive set is empty. The following stages are repeated in algorithm generations. Non dominated individuals are copied into archive and all the dominated individuals are eliminated in the archive and copies. If size of archive exceeds the defined limit, extra individuals will be eliminated by a clustering algorithm which keeps the non-dominated front. Then a specific fitness is given to individuals. In the next step more individuals are selected from two sets and after passing through crossover and mutation, they will be considered as new individuals in the pool.

1-2-2- NSGA-II Algorithm

In NSGA-II algorithm, first P_0, a new primary pool is started and individual are sorted as non-dominated. Each solution is ranked based on its fitness or non-domination level (level 1, 2,..) (Minimization problem). Then utilizing normal genetic operators (such as binary competitive selection, crossover and mutation) a child pool Q_0 with size of N is built. Since elitism is presented comparing current population with previous solution which had most non dominations, the algorithm structure will be different after first population.

First two parent and child populations is combined (to guarantee elitism) and they will sort based on non-domination (a population with size 2N). Then N individuals are selected from them based on the best non-denomination sets of F_1, F_2, … F_l respectively. Since it is possible that the size of population exceeds N by selecting all of the individuals F_l, this set and others will be sorted descending based on population-comparison operator $\prec_n$ and a proper number of them would be removed. Now from derived population P_{t+1} a new population Q_{t+1} is established utilizing selection, crossover and mutation operators.

The binary-competitive selection operator ($\prec_n$) is based on population-comparison operator however since this operator requires crowding distance and ranking, these values are obtained during establishment of population P_{t+1}. This procedure is shown in Figure 3.

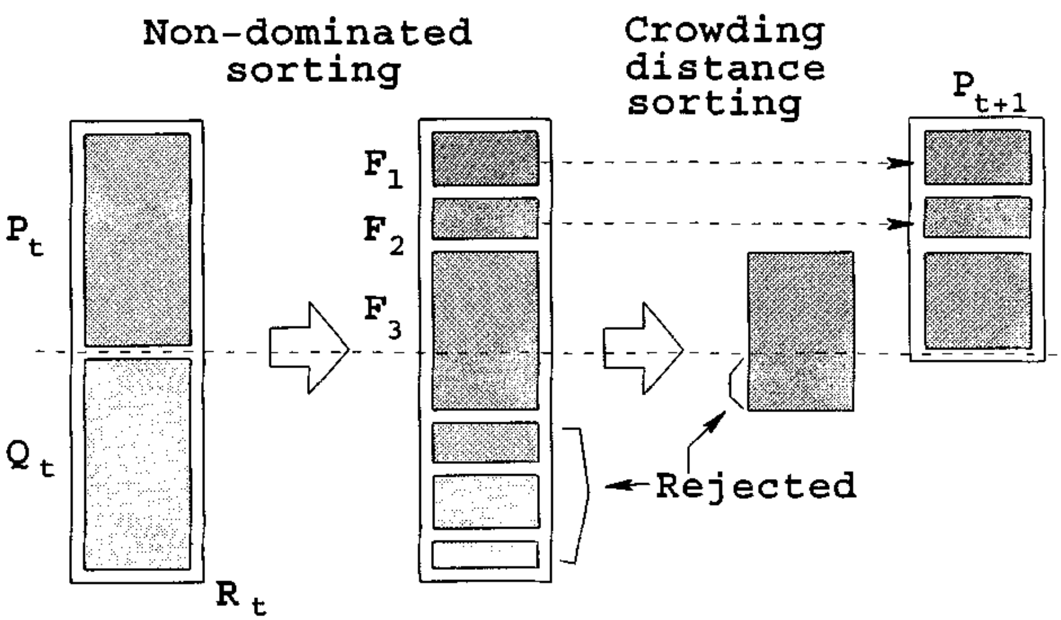

Figure 3: The procedure of NSGA-II

The authors of (Dasgupta *et al.*, 2008) have developed their work in (Dasgupta *et al.*, 2009; Dasgupta *et al.*, 2012) to study NSGA-II behavior when the problem is not static anymore and condition is changing dynamically. In this case which is called task based sailor assignment; it will assign a task to each sailor in various time intervals. In fact, instead of assigning a single task to each sailor, a set of tasks are assigned which are distributed specific time slots (for example, during a day each sailor has different job in different shifts). An example of this layout is shown as a chromosome in Figure 4.

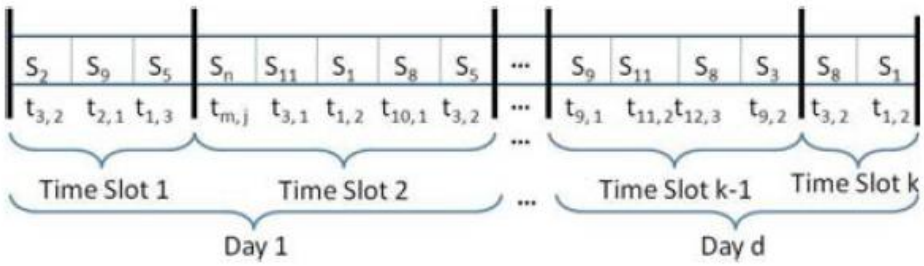

Figure 4: Reconfiguration of chromosome in TSAP

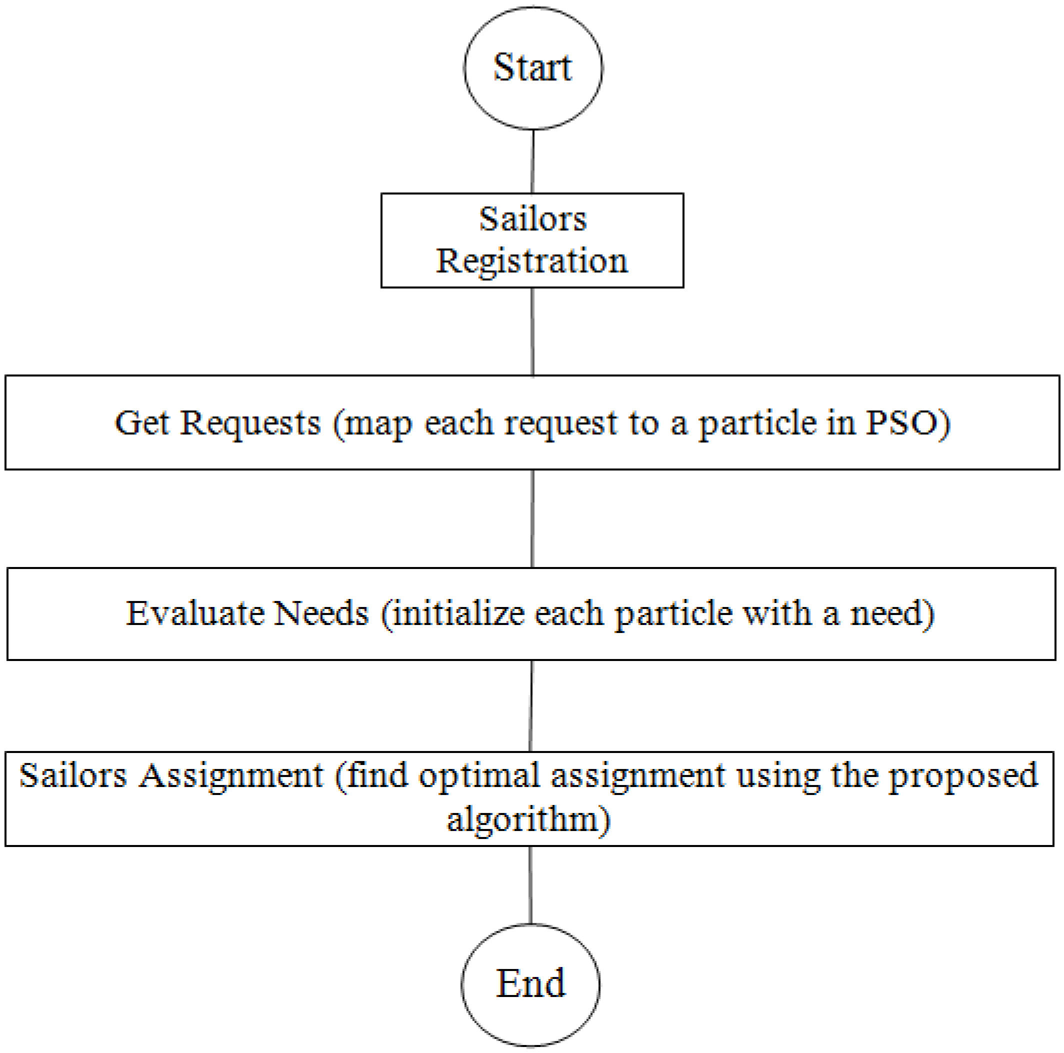

Figure 5: Proposed Algorithm Flowchart

In (Sutton and Dimitrov, 2013) another instance of this problem has been designed and solved which can be used in other environments such as consulting firms. In this paper a criteria is defined for importance if each objective which can be changed based on decision maker's interest. In this paper, data envelopment analysis method has been used (Sutton and Dimitrov, 2013). In this paper a different example of problem has been defined which is different than original sailor assignment problem.

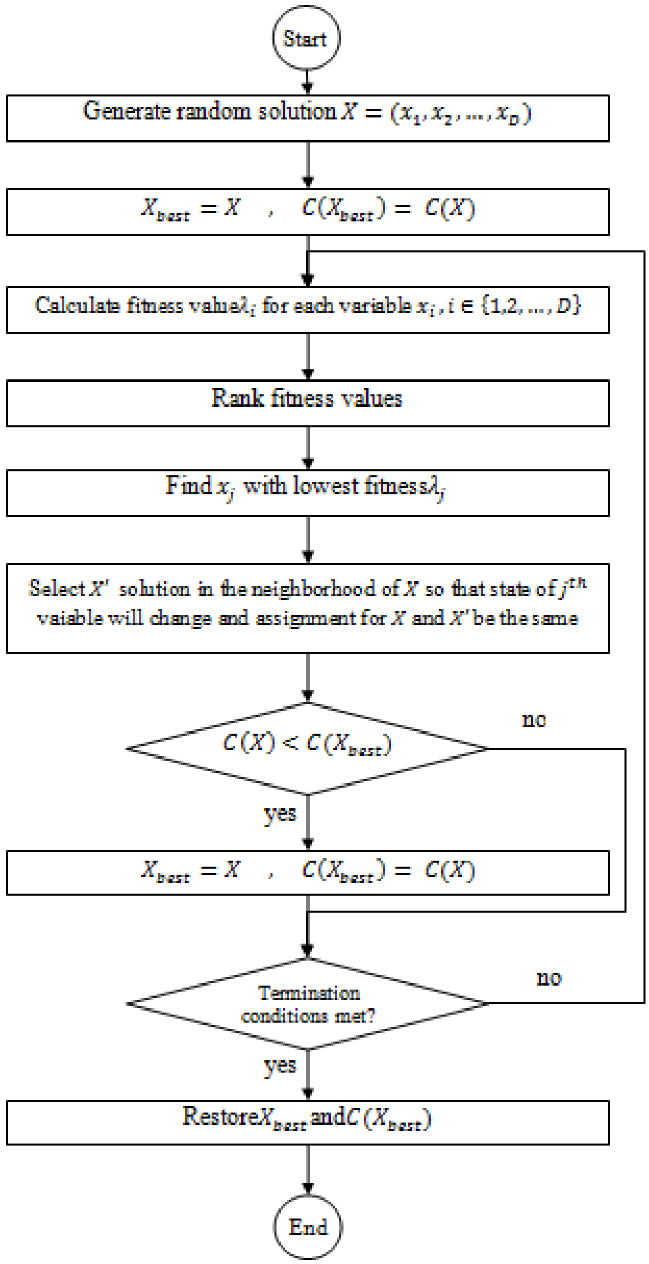

Figure 6: EO algorithm Flowchart

2- The Proposed Model

In current multi-objective evolutionary algorithm such as NSGA-II and SPEA, the diversity and evaluation of the search space can only be achieved by adjusting mutation and synthesis operators

Therefore, in this study we are trying to search the space thoroughly utilizing particle swarm optimization PSO algorithm and EO algorithm without adjusting adaptation parameters. For this purpose, first two PSO and EO algorithm is addressed and then their combination is described. At the end, modeling of sailor assignment problem is presented in PSO-EO algorithm approach. The procedure for solving sailor assignment problem is shown in Figure 5. After registering sailors in the first step, each sailor interests and particles is scripted in PSO which was explained in 1-3. In the third step, value of each particle is done by a demand to be able to use PSO features to find the optimum assignment. In the last step, the optimum assignment is found using proposed algorithm which is presented in section 3-3 and its flowchart is shown in Figure 7.

2-1- Particle Swarm Optimization (PSO)

PSO is a tool for optimization based on population such that a system is initialized by a random particles population. Search algorithm to find the optimum point is achieved using updating generations. Assume search space has D dimension, location of a particle can be shown in vector $X_i = (x_{i1}, x_{i2}, ..., x_{iD})$ and its velocity in vector $V_i = (v_{i1}, v_{i2}, ..., v_{iD})$. The best position of particle i in previous step is defined by $P_i = (p_{i1}, p_{i2}, ..., p_{iD})$ and position of best general point is shown by $P_g = (p_{g1}, p_{g2}, ..., p_{gD})$

The fitness value for each particle can be calculated from applying it in the target function. Also, the new pace and position for particles can be recalculated using the equations (3) and (4).

$$(3) \qquad v_{id}^{t+1} = w^t v_{id}^t + c_1 r_1^t (p_{id}^t - x_{id}^t) + c_2 r_2^t (p_{gd}^t - x_{id}^t)$$

$$(4) \qquad x_{id}^{t+1} = x_{id}^t + v_{id}^{t+1}$$

where, $d \in \{1,2,..., D\}$ and $i \in \{1,2,..., N\}$ in which N is the population size. Also, the superscript t is the iteration number. Moreover, the coefficient w is constant weight, r_1 and r_2 are random numbers in the range of $[0..1]$, c_1 and c_2 are cumulative traversal parameters which are positive constants.

2-2- Extremal Optimization (EO)

This approach was invented to describe the complexity due to physical systems. EO can successfully update undesirable variables in an optimum sub-solution by assigning new random values. In addition, any change in the fitness value of a variable makes change in the fitness of neighboring variables lends elegance. High dynamic variance in population effectively reveals many local optimums. In such way, EO shows its strong local search ability. EO algorithm flowchart for minimization problems is shown in Figure 6.

In EO algorithm, each variable of the local solution X is called component which is used often in biology. For example, if $X = (x_1, x_2, x_3)$, then x_1, x_2, and x_3 are called X's components. It can be observed that in EO algorithm in contrast to GA which works on a population of candidate solutions, a sub-optimal solution is selected and some changes is locally made on the worst element of X(Chen *et al.*, 2010). The λ_i fitness value for each component x_i is required to be used in ranking of the components within an iteration. This manner is different form approaches such as evolutionary algorithms in which equal fitness values are considered for all components in the solution.

2-3- PSO-EO

PSO consists of a particle set in which each particle moves inside a common search-space at a determined velocity that is constantly updated by previous best particles in the last and overall movement iterations. It is possible to simply implement PSO with low memory and computational power requirements. Despite having an efficient implementation of PSO, it may suffer from premature convergence particularly when there are several climaxes in a problem (Kennedy and Eberhart, 1995).

To avoid premature convergence in PSO, the idea that will be discussed in this paper is a hybrid of PSO with EO.

Almost all hybrid methods use the collaboration to ensure faster PSO convergence within search-space while EO by having the capability of local searches could significantly improve the chance of escaping from local optimums.

This hybrid method is made of most potential in PSO exploration and EO productivity. As the result, limitations of PSO such as trapping in local optimums could be resolved. Although applying EO in all iterations of PSO may increase the length of each iteration time, it would reduce the overall convergence time. For better integration of PSO with EO, once INV iterations of PSO occurred EO should be run (for example INV = 10 means that EO is run once every 10 iterations of PSO).Hence, PSO-EO method can augment PSO to converge quickly and escape from local optimums to achieve a better global result by EO capabilities.

2-3-1- PSO-EO Algorithm

PSO-EO flowchart which is shown in Figure 7, demonstrates needed steps to solve a D-dimensional minimization problem. The fitness value for all components of solution has to be calculated for finding the worst component in EO. In contrast, PSO-EO procedure computes the fitness value by applying the position of each particle in the target function. Therefore, each solution component's fitness value for non-constraint minimization problems can be obtained by the following:

$\lambda_{ik} = OBJ\ (X_{ik}) - OBJ\ (P_g)$

where, λ_{ik} is the fitness value for i^{th} particle of the k^{th} component which defined as the cost for a mutation, X_{ik} is the particle's new position, $OBJ(X_{ik})$ is the value of target function for X_{ik}, and also $OBJ\ (P_g)$ is the value of target function for the best position of the current iteration.

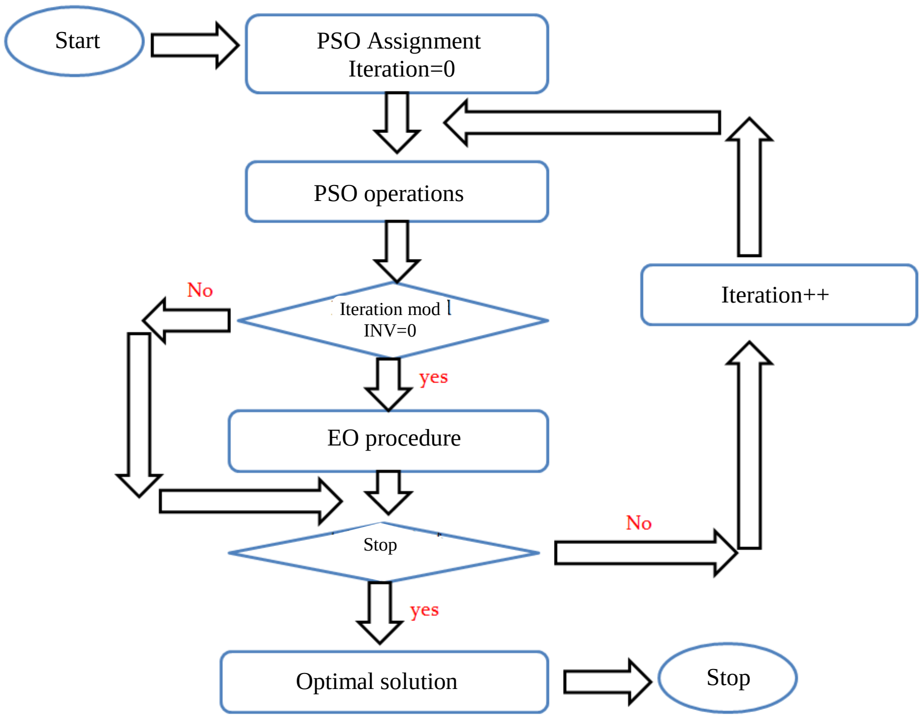

Figure 7: PSO-EO Algorithm Flowchart

2-3-2- Mutation Operator

After introduction of merely mutation operator, the mutation plays a key role in EO. In this work, we used a combination of GC mutation operator which is introduced by Chen and Lu (Chen and Lu, 2008).This combinatorial method is composed of Gaussian and Cauchy mutations. The mechanisms of Gaussian and Cauchy mutations were studied by Yao. The basic features of Cauchy and Gaussian mutations are coarse-grain and fine-grain search, respectively. In a GC mutation, Cauchy is applied at first which means that a long step will be used for each mutation. If the mutation produces a new variable which was not in the desired range, the mutation can be repeated TC times while the generation remains in the admissible range. Otherwise, Gaussian mutation will be repeated for TG times until the produced generation is admissible. Here, the length of the step is less than previous steps. If after the mutation the newly created variable is still not within the admissible range, the low or high threshold will be assigned for it. Thus, GC mutation is composed of all advantages of fine and coarse-grain search. Some switch algorithms can indicate which search strategy should be chosen. However, GC mutation does not need such decision makings.

Gaussian mutation in this method carried by the following formula:

$$(5) X_k' = X_k + \mathrm{N}$$

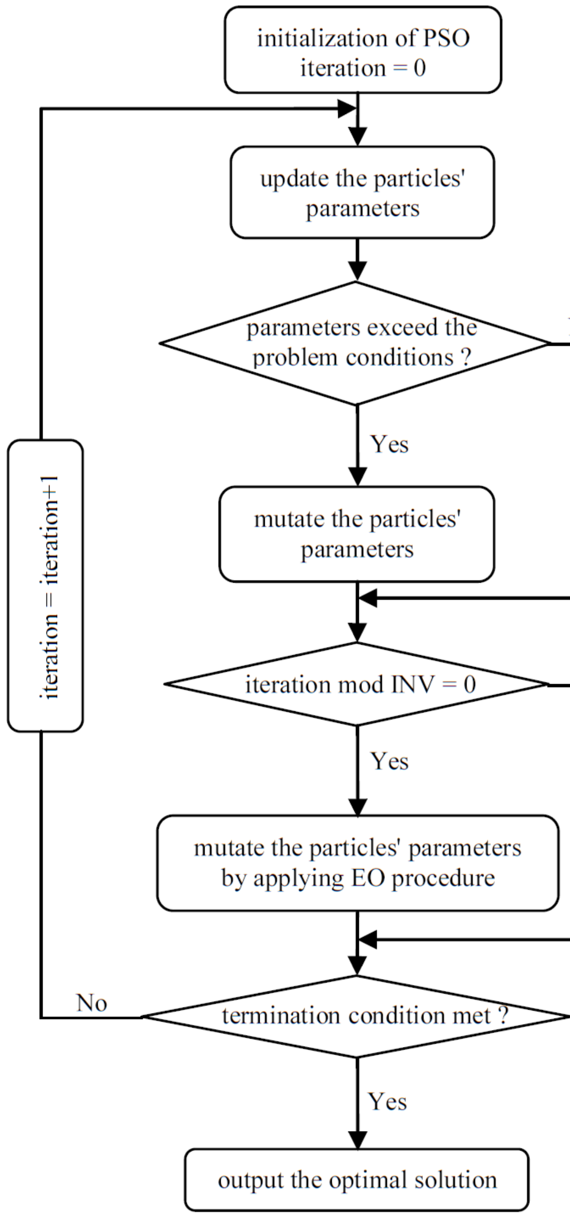

Figure 8: PSO-EO Algorithm Flowchart

Where,X_k and X_k' are k^{th} decision variables before and after the mutation, respectively. Also, $\mathrm{N_k}(0,1)$ is a random Gaussian number with mean of zero and standard deviation of one which is generated for k^{th} decision variable?

Cauchy mutation occurs by the following:

$$(6) X_k' = X_k + \delta$$

Where, δ_k is Cauchy's random variable with size variance parameter of one that is generated for k^{th} decision variable.

In GC mutation, TC and TG parameter values are preset by the user.TC has effect on the time of coarse-grained search. Similarly, TG has the same effect on the time of fine-grained search. It should be noted that none of these values can be large, because they extend search process time and increase the computational overhead.According to (Chen and Lu, 2008), appropriate values for TC and TG could reside in the range of [2..4].

2-4- Modified PSO-EO (MPSO-EO)

During velocity update in standard PSO algorithm, if the velocity of a particle is more than maximum or less than minimum admissible values, it will be replaced with the maximum or minimum value, respectively. This causes that out of solution range particles are replaced by maximum and minimum when the update is taking place. In some problems with difficulty in finding the global optimum which require a high number of iterations to reach their optimum, this will cause reduction in diversity and also late or never convergence.

In order to consider problem constraints and to avoid the loss in diversity, the product of particle values into admissible maximum and a random number between zero and one divided by admissible maximum value was used for updating the outliers instead of maximum and minimum values (formula (7)).Afterwards, to increase diversity and change the velocity direction, for outlier particles the velocity will be inversed and multiplied by a random number between [0..1] (formula (7)).

(6)$\mathrm{P_{ik}} = $ remai

(7)$V_{ik} = -V_{ik} *$

The proposed method that was depicted in Figure 8, is a combination of PSO exploration and EO productivity.

2-5- MPSO-EO Solution For Sailors Assignment Problem

As mentioned in the previous section, one of the things that can be imposed as more research on this method, is to apply it on the combinatorial optimization problems such as sailor assignment problem. Hence, this section describes the necessary definitions and introduces new encodings using the proposed hybrid method for solving sailor's assignment problem.

2-5-1- Sailors Assignment Problem

The sailors assignment problem is known as a NP problem that can be defined as the following: In the bipartite graph $G = (V,E)$, V is the set of nodes (the assignment) and E is the set of edges (sailors and tasks).For example, there is$N$ assignments so that $V = \{1,2,\dots,N\}$and $\mathrm{E} = \{(\mathrm{i},\mathrm{j},\mathrm{w_{ij}}),\mathrm{i},\mathrm{j}\epsilon\mathrm{V},\mathrm{w_{ij}}\epsilon\mathrm{R^+}\}$.The goal in sailors assignment is to find the optimum assigned set of works with the lowest possible assignment cost for each sailor. This set can be presented as the following:

The sequence of nodes is presented like $\mathrm{v_{k_1}v_{k_2}v_{k_3}\dots v_{k_{N-1}}v_{k_N}v_{k_1}}$and the sequence of edges as$\mathrm{e_{k_1k_2},e_{k_2k_3},\dots,e_{k_{N-1}k_N}\,e_{k_Nk_1}}$.

2-5-2- Description of Position Vector and Search-Space

Since the result of assigning sailors problem is the shortest path Hamiltonian cycle, the position of all particles nodes can be considered as a Hamiltonian cycle. Thus, the search space is a set of positions that includes whole generated cycles of $N-1$ nodes.In this sense, each of particle positions represent a solution of the problem. For example, the i^{th} particle can be noted by $\mathrm{X_i} = (\mathrm{v_{i1}},\mathrm{v_{i2}},\dots,\mathrm{v_{in}},\mathrm{v_{i1}})$.In this notation, $\mathrm{v_{i1}},\mathrm{v_{i2}},\dots,\mathrm{v_{in}},\mathrm{v_{i1}}$shows a path along N nodes starting from $\mathrm{v_{i1}}$ traversing through all other nodes to reach$\mathrm{v_{i1}}$again via $\mathrm{v_{in}}$. This notation corresponds to an assignation between sailors and tasks.

2-5-3- Cost Function

The function for i^{th} particle is defined as following:

in which, if $k = N$ then $k+1 = 1$. The goal isto find the position of a particle so that it has the lowest possible fitness value.

2-5-4- Velocity Representation

Due to properties of sailor assignment problem, the iterative formula (3) is not applicable. Therefore, this formula is reconstructed as the following:

Research Article

Definition 1: Suppose the sequence of solution in sailor assignment problem with N nodes is as $\mathrm{X} = (\alpha_i), i = 1,2,\dots,N+1$. Mapping operator $\mathrm{T}(i_1, i_2)$ is to remove α_i from i_1^{th} position and add it to the i_2^{th} position within X solution. As a result, the new solution of X' will be established.
For example, by applying $V = T(5,2)$on $X = (1,5,4,3,2,1)$, there will be $\mathrm{X'} = \mathrm{X} + \mathrm{V} = (1,5,4,3,2,1) + \mathrm{T}(5,2) = (1,2,5,4,3,1)$. In this example, '+' sign has a new meaning that is mapping of the solution with respect to the operator or mapping sequence.
Definition 2: $\mathrm{ST} = (\mathrm{T}_1, \mathrm{T}_2, \dots, \mathrm{T}_n)$ is a mapping sequence that is composed of mapping operators $\mathrm{T}_1, \mathrm{T}_2, \dots, \mathrm{T}_n$. The order of running the operator is important for a mapping sequence.
Definition 3: Different mapping sequences may lead to the same solution. All of these sequences are the same set of mapping sequence.
Definition 4: Two mapping sequence can be combined to make a new mapping sequence. In order to provide a notation, $\odot$ is defined as the binary combining operator which produce a new mapping sequence from combining two of them.
Below is an example of making a mapping sequence that can be seen. Here, the aim is to make ST mapping sequence and apply it on X_1 to obtain solution X_2.

Initially, the first point of difference between two solutions should be found that is $\mathrm{X}_2(2) = \mathrm{X}_1(3)$, means that T1(3,2).Now, X_1^1 should be found i.e. $X_1^1 = X_1 + T_1\ (3,2) = (1,2,3,4,5,1)$and $\mathrm{X}_2(3) = \mathrm{X}_1^1(5)$.For the second time, as previous the mapping operator is $T_2(5,3)$and $\mathrm{X}_1^2 = (1,2,5,3,4,1)$. Next, $\mathrm{X}_2(4) =$ X12(5) should be computed with *T35,4* operator to get desired solution *X2*. Finally, the mapping sequence is provided as the following:

in which, the sign '-'means finding mapping sequence between two solutions.
According to presented descriptions, the velocity of a particle can be a mapping sequence of nodes' positions. In fact, the velocity is the subtraction X from Y as the following:

In (3) to calculate the velocity, there is$c.V$ so that V is velocity and c is a random number.By looking at the above formula, V is a mapping sequence while c is a random number. Thus the multiplication of them can be defined as follows:

(8)

where, $\|V\|$ is the number of operators in the mapping sequence. Also, $\|V\| = 0$ means that there is no mapping operator within the mapping sequence. Furthermore, $\lfloor c\|V\|\rfloor$ shows the lowest limit of product $c$by $\|V\|$. In (3), $w.V_i^t$ means holding the mapping operator at the velocity of last state considering (8). Thus, weight factor w can be computed by the following:

In which, Mt is the maximum number of algorithm iterations that is a constant value. Also, t is the current iteration number in each iteration of algorithm execution (Zhang and Xiong, 2009).

2-5-5- Augmenting Particle Congestion Optimization By EO

Based on the several executions of particle congestion optimization algorithm, it was observed that the algorithm mostly stops in a local optimum. In order to escape from such locals, an augmentation by adding EO is proposed in this paper. To do so, initially two points should be selected randomly within the solution range. Afterwards, its solution substring should be inserted into the main solution. After recalculation of the fitness value, if the value was better than before, this change would be made permanent. Otherwise, this method should be repeated for TC times to reach a desired solution. In the case of failure, one point inside the rage of solution should be chosen randomly and from that point to the end of the solution string must be inverted. If the resulted fitness value was better than before, the change would be made permanent. Otherwise, there will be no change on the particle. This method also runs a number of TG times until the produced generation is admissible. Also, the values of TC and TG are discussed in section 4.

3- Experimental Results

Since sailor assignment problem is not in the form of bipartite graph and has different premises than Kuhn-Munkres algorithm, the following issues should be considered:

1. In sailor assignment problem, no person can do all tasks. Therefore, Inf (infinity) value should be used for tasks that a person is not able to do it.
2. In sailor assignment problem, there may occur some situations in which there is no acceptable answer. In such cases, dummy tasks should be added to the task set.

To evaluate the performance of the proposed algorithm, it is compared to the algorithm presented in (Dasgupta *et al.*, 2008). As mentioned before, (Dasgupta *et al.*, 2008) used NSGA-II algorithm with Kuhn-Munkres as initializer. To compare five assignments goals resulted from Kuhn-Munkres with TS, SR, CR, PCS, and weighted average, both NSGA-II algorithms from (Dasgupta *et al.*, 2008) and the proposed algorithm were implemented. This intelligent initialization causes to have a better start in both algorithms. Because the paper's sample database is belonged to US Navy and is not publicly available, the problem is investigated on randomized cost matrices. It is noteworthy that both algorithms were investigated on the same randomized cost matrices which were stored on a secondary memory to prevent them from loss.

Table 1: Cost of Kuhn-Munkres algorithm regarding five cost matrices

Number of sailors	Number of tasks	Goal	Cost
100	110	TS	1.3270
		PCS	1.3625
		SR	1.3550
		CR	1.3226
		w1*TS+w2*PCS+w3*SR+w4*CR	16.6974
200	210	TS	1.5183
		PCS	1.4370
		SR	1.3692
		CR	1.4772
		w1*TS+w2*PCS+w3*SR+w4*CR	29.8226
300	310	TS	1.5314
		PCS	1.4269
		SR	1.4704
		CR	1.3910
		w1*TS+w2*PCS+w3*SR+w4*CR	40.9468
400	410	TS	1.5717
		PCS	1.4970
		SR	1.4853
		CR	1.5041
		w1*TS+w2*PCS+w3*SR+w4*CR	50.1418
800	810	TS	1.5861
		PCS	1.5342
		SR	1.5498
		CR	1.6179
		w1*TS+w2*PCS+w3*SR+w4*CR	85.2073

Table 1 presents assignment costs obtained from running Kuhn-Munkres on five randomized cost matrices that were described above.

In order to implement (Dasgupta *et al.*, 2008), the following coding was used:

- Inidi.a: A vector with equal size of sailors' count in which each element represents the task number for the corresponding individual.
- Indi.F: A vector with equal size of tasks' count in which each element with the value of 1 indicates that the corresponding task is free to assign and also 0 values indicates the unavailability of the task.
- Indi.f: A vector with five elements in which each element includes one of five mentioned goals regarding to the current assignment.
- *Crossover Operator:* The crossover operator occurs on two different parents by the probability of 0.7.In fact, this operator substitutes the assigned tasks of two parents. If these assignments were not admissible, this manner will be repeated until reaching an admissible assignment.
- *Mutation Operator:* If crossover was not occur, the mutation operator may be applied by the probability of 0.3. In this sense, one parent will be selected and if it was idle, one available task would be given to it. If there is no available task to assign, one of assigned tasks will be seized from an individual and this task will be assigned to the selected parent.

Both C_1 and C_2 parameters of the PSO-EO method is set to 2. Also, the constant weight W will be assigned linearly between 0.4 and 0.9 with respect to iteration counts. Furthermore, upper and lower limitations for each dimensional velocity will be calculated as $(V_{min}, V_{max}) = (X_{min}, X_{max})$. Moreover, TC and TG \parameters are set to 3 in GC mutation.

As can be seen in Figure 9 to 18 corresponding to different number of sailors and tasks, both NSGA-II and MPSO-EO reach to almost identical solutions that are of pareto front of the problem. However, the individuals in the last generation of the proposed algorithm are more converged than the ones in last generation of NSGA-II. This suggests that people of PSO are more placed into the pareto front in comparison with NSGA-II people.

In these Figures, horizontal axis contains 1,3,5,7 which are respectively TS, PCS, SR, and CR goals. The running time of these algorithms for reaching to pareto front is noticeable. Considering the same population for both algorithms (30 persons for NSGA-II and 30 persons for PSO-EO), as seen in Table 2, the proposed algorithm converged to pareto front similar to (Dasgupta *et al.*, 2008) but in much less time. It is noteworthy that both algorithms run until no noticeable change observed in the best person of population. By increasing the problem size, more difference in running time of the algorithms arises. Figure 19 shows a comparison between running times of the two mentioned algorithms.

Table 2: Running time in seconds

Number of sailors	Number of tasks	PSO-EO running time	NSGA-II running time
100	110	6.63	22.07
200	210	19.78	59.54
300	310	39.82	119.17
400	410	68.76	212.65
800	810	241.17	851.98

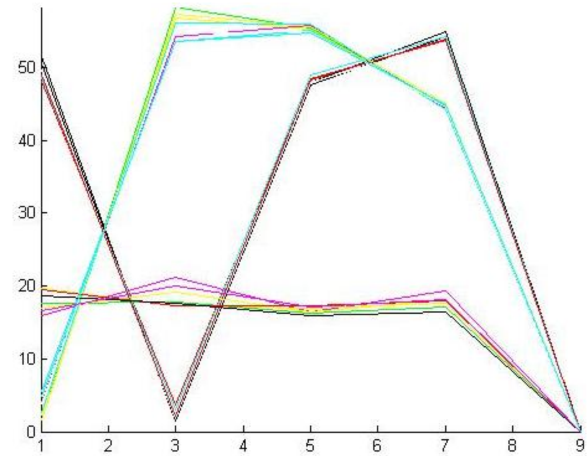

Figure 10: Sailor assignment problem solution with 100 sailors and 110 tasks using NSGA-II

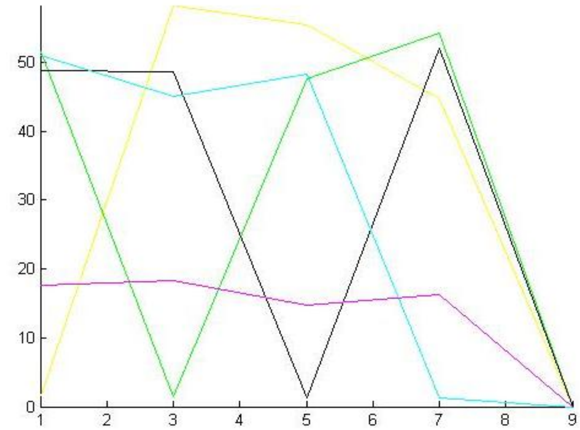

Figure 9: Sailor assignment problem solution with 100 sailors and 110 tasks using PSO-EO

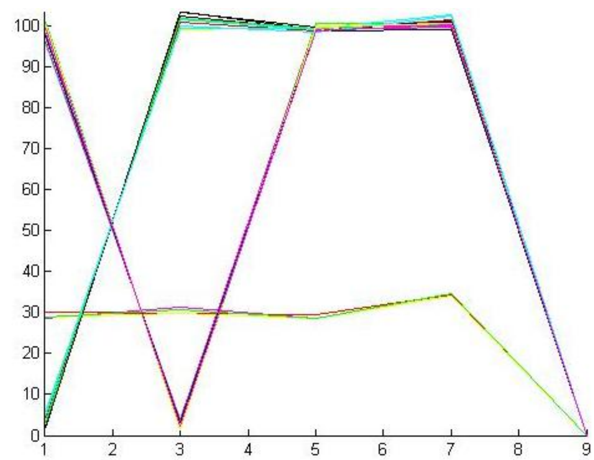

Figure 12: Sailor assignment problem solution with 200 sailors and 210 tasks using NSGA-II

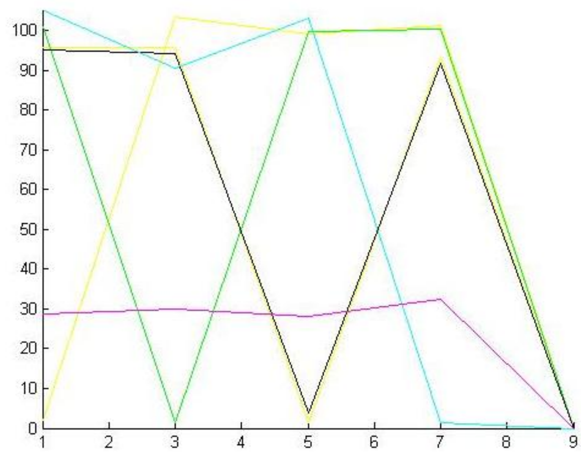

Figure 11: Sailor assignment problem solution with 200 sailors and 210 tasks using PSO-EO

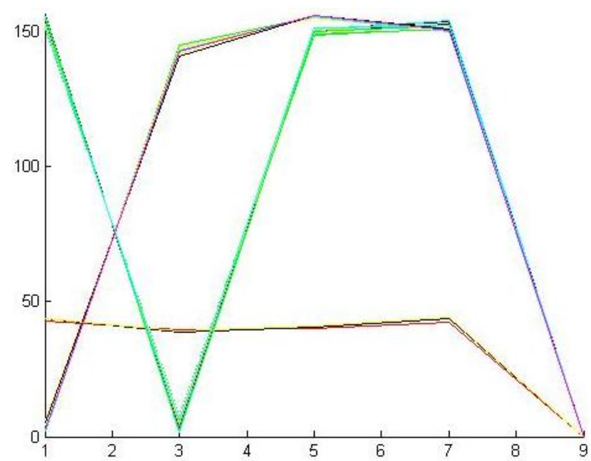

Figure 14: Sailor assignment problem solution with 300 sailors and 310 tasks using NSGA-II

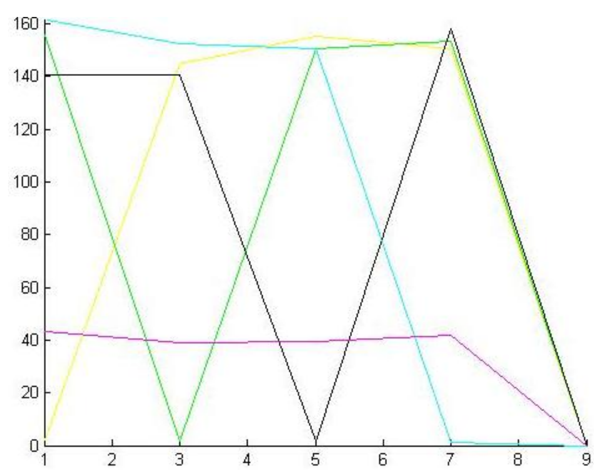

Figure 13: Sailor assignment problem solution with 300 sailors and 310 tasks using PSO-EO

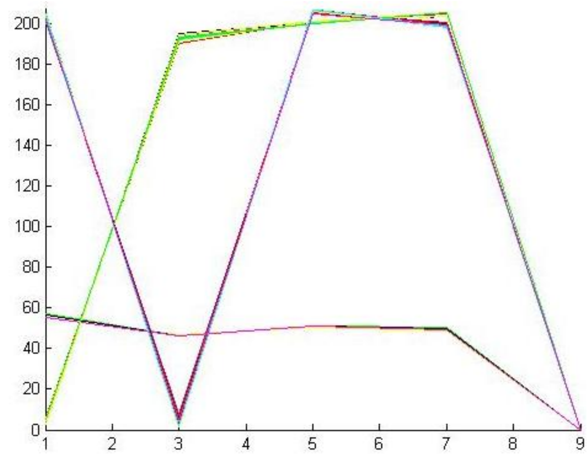

Figure 16: Sailor assignment problem solution with 400 sailors and 410 tasks using NSGA-II

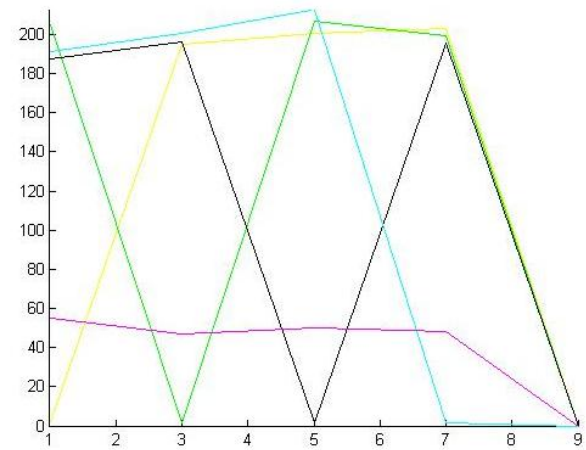

Figure 15: Sailor assignment problem solution with 400 sailors and 410 tasks using PSO-EO

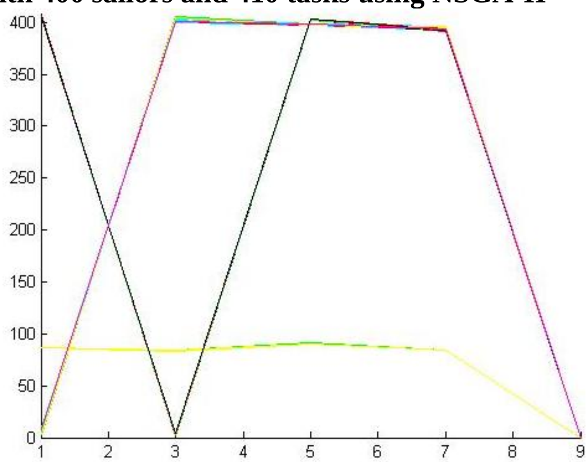

Figure 18: Sailor assignment problem solution with 800 sailors and 810 tasks using NSGA-II

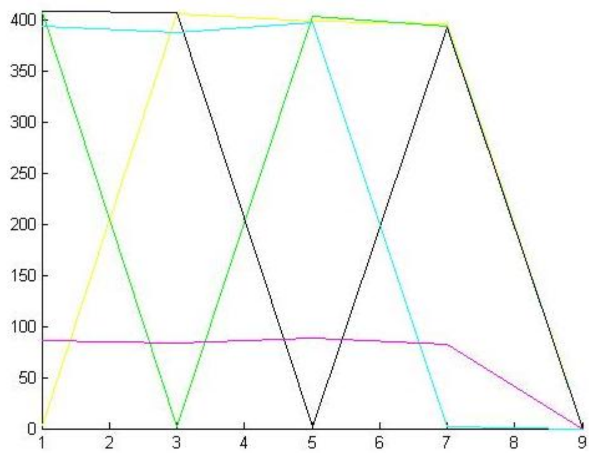

Figure 17: Sailor assignment problem solution with 800 sailors and 810 tasks using PSO-EO

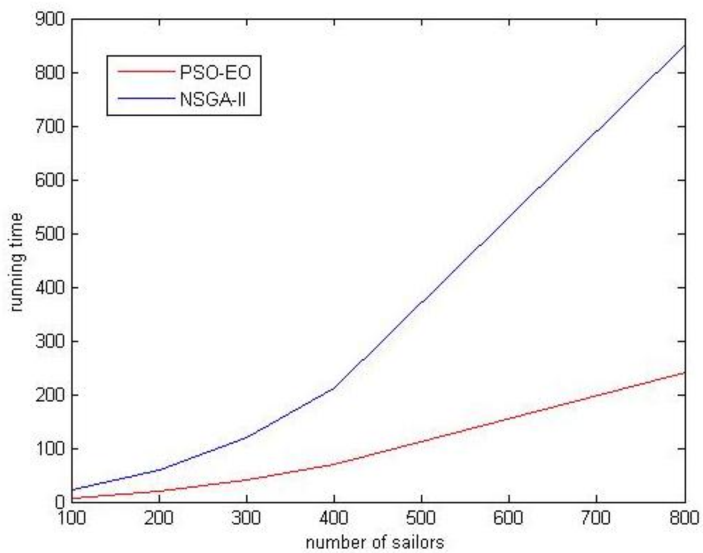

Figure 19: Running time comparison

Conclusion and Future Works

In this paper, we developed a new hybrid optimization method called MPSO-EO that consists of PSO and EO. This hybrid method benefits from exploration in PSO and productivity in EO. Results imply the capability of the proposed method for solving intractable problems within shorter time in analogy with other randomized algorithms.

Future works can include the followings: 1) Investigate combinatorial optimization problems around the proposed algorithm such as vehicular routing, graph partitioning, graph coloring and so on.2) Use alternative evolutionary methods such as ABC instead of PSO. 3) Devise a method for tuning the proposed algorithm parameter to avoid adjusting by the user. Moreover, in the structural details of PSO some revisions could be performed such as change in topology, neighbors, and similar methods.

ACKNWLEDGEMENT

We are grateful to Islamic Azad University, Quchan branch authorities, for their useful collaboration.

REFERENCES

Chen MR, Li X, Zhang X and Lu YZ (2010). *A Novel Particle Swarm Optimizer Hybridized with Extremal Optimization* (Elsevier) *Applied Soft Computing* 367-373.

Chen MR and Lu YZ (2008). A novel elitist multi objective optimization algorithm: multi objective extremal optimization, *European Journal of Operational Research* **188** 637–651.

Dasgupta D, Garrett D, Niño F, Banceanu A and Becerra D (2012). A Genetic-Based Solution to the Task-Based Sailor Assignment Problem, *In Proceedings of Variants of Evolutionary Algorithms for Real-World Applications* 167-203.

Dasgupta D, Hernandez G, Garrett D, Vejandla PK, Kaushal A and Yerneni R (2008). Comparison Of Multiobjective Evolutionary Algorithms with Informed Initialization and KuhnMunkres Algorithm For The Sailor Assignment Problem, Gecco, USA.

Dasgupta D, Nino F, Garrett D, Chaudhuri K, Medapati S and Kaushal A (2009). A Multiobjective Evolutionary Algorithm for the Task Based Sailor Assignment Problem, Gecco, Canada.

Deb K, Pratap A, Agarwal S and Meyarivan T (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* **6**(2) 182–197.

Garrett JD, Vannucci J, Silva R, Dasgupta D and Simien J (2005). Genetic algorithms for the sailor assignment problem. *In Proceedings of the 2005 Genetic and Evolutionary Computation Conference (GECCO-05)* (ACM press).

Kennedy J and Eberhart RC (1995). Particle swarm optimization, in: Proceedings of the 1995 IEEE International Conference on Neural Networks, IEEE Service Center, Piscat away 1942–1948.

Kuhn HW (1955). The hungarian method for the assignment problem. *Naval Research Logistic Quarterly* **2**(1) 83–97.

Shmoys DB and Tardos E (1993). An approximation algorithm for the generalized assignment problem. *Math Programs* **62**(3) 461–474.

Sutton W and Dimitrov S (2013). The U.S. Navy explores detailing cost reduction via Data Envelopment Analysis, *European Journal of Operational Research* **227** 166–173.

Zhang JW and Xiong W (2009). An Improved Particle Swarm Optimization Algorithm and its Application for Solving Traveling Salesman Problem, *IEEE, World Congress on Computer Science and Information Engineering*.

Zitzler E, Laumanns M and Thiele L (2001). SPEA2: Improving the strength pareto evolutionary algorithm. Technical Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Gloriastrasse 35, CH-8092 Zurich, Switzerland.