Maritime Resource Allocation using Evolutionary Algorithms

Mayamin Hamid Raha
Department of Computer Science and Engineering
University of Nevada, Reno
Reno, Nevada
mraha@nevada.unr.edu

***Abstract*—We aim to approach a bin packing problem with an evolutionary algorithm. The idea is to use an Elitist Genetic algorithm (GA) for searching through a space of 6 triangles placed in six different locations of an area representing a ship. The triangles are placed in various orientations from the positive x axis(through co-ordinate system transformation for every location's co-ordinate) in clockwise direction such that it maximizes the area coverage inside the circle. The application field of this bin packing approach is related to placement of assets on a ship in various locations with such orientations that maximizes the ship's protected region. This work is an attempt to observe whether GA is able to find the optimum solution for a cases like this. We found that GA is highly efficient in searching through huge search space of 2^{144} achieving an average maximum fitness of 97 percent within less than 70 generations.**

***Index Terms*—Elitist Genetic Algorithm, bin-packing, selection, crossover, mutation**

I. Introduction

Large sheets of metal, wood, glass needs to be cut in various industries which gave rise to packing and cutting problems. The solutions to encounter these problems comprises of objectives related to minimizing resource wastage through minimization of the number of cuts and maximization of the area utilization. Packing problems in general are optimization problems that tries to find good arrangement of multiple items in larger containing regions. Cutting and packing problems known as cutting stock problems (CSP) have been existing since 1960 [4], [5].These problems are mostly concerned with packing rectangles of different sizes inside a big rectangle [3], [7]. Polygon packing problems have also gained attention of researches and many works have been done on packing polygons inside rectangles. [9].

Inspired by all these works, we decided to define our problem as a Bin-packing problem and used evolutionary algorithm due to their efficient search methodology. In our work, we are considering the use of genetic algorithm that finds the optimum arrangement of triangles representing the area covered by assets of a ship inside a fixed circular region. Here, the circular region's radius is 128 units from the center of a circle which we are considering the center of ship. We attack this circular region packing by triangle problem with a CHC selection [13] based genetic algorithm to observe which combinations of height and angles of triangles cover most of the area inside a circle where the maximum value range for height and angle between two adjacent sides of triangle is from 1 to 256 units and 120 units respectively.Here we will be placing triangles in 6 different locations inside the circle(see 1) giving each of the angles a freedom of rotation (representing real life ship's asset movement) from positive x axis that means the triangles can rotate in each location having a rotation angle from 0 to 360 units.

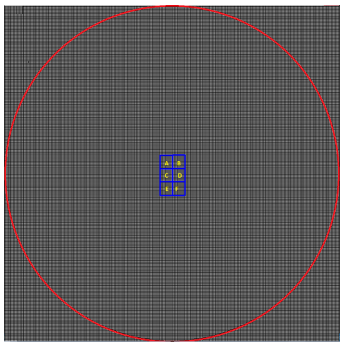

Fig. 1: Circular area in red and blue rectangular area representing a ship with 6 different points namely A, B ,C, D , E , F where triangular regions representing assets will be placed

II. Related Work

One of the earliest bin-packing problems that existed is orthogonal bin-packing problem (OPP). In a given finite number of rectangles and a rectangular board, OPP requires a disjunctive placing of rectangles on the board such that the edges are parallel to x and y axis [2]. Later on, polygon packing problems evolved and [15] tackled this by placing polygons directly on the board and they used an algorithm that does local optimization by shifting and rotating the polygons ,whereas, [1] approached the problem by placing cluster of 3 polygons on rectangular board. In [9], Jakob et al. used two methods for solving polygon packing problem by applying genetic algorithm directly to polygons and by only applying genetic algorithm [12] to rectangle in which the polygons are embedded followed by use of a deterministic algorithm.

Another branch of cutting stock problems (CSP) is called Open Dimensional Problem (ODP) which consists of deter-

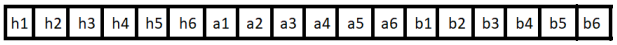

Fig. 2: Chromosome structure

mining the best arrangement of a set of rectangular items onto a rectangular strip of material for a fixed height and width where overlapping is not allowed.

Moreover, packing problems can be multi-objective too. In [14],the authors use multi-objective evolutionary algorithms(MOEAs) for optimization of multi-objective formulation of ODP. MOEAs have been used further for 2D ODP [8]. Furthermore recently 3D bin packing problems came into being, where the only difference from Bin-packing is that now the orthogonal packing of given set of rectangular object is inside three dimensional rectangular bins. [11].

Some of the recent advancements in approaching bin-packing problem with genetic algorithms includes [6] where the Score Constrained Packing Problem (SCPP) that involves packing items into a minimal number of bins such that the order and orientation of each items within each bin satisfies a sum constraint has been addressed with an evolutionary algorithm that has option for 3 kinds of recombination. Another recent study on solving irregular bin packing problem was attempted with genetic algorithm by Luo et al. in [10]. This type of problems have a special boundary constraint and the objective is to fir irregular shaped pieces with free rotations to fill up a large irregular stock sheet where goal is to maximize the number of filled pieces.

Motivated by all these approaches we have developed a dynamic bin packing approach involving packing of triangles with some specific height, angle and orientation ranges placed in 6 different co-ordinates with in a specific boundary region that comprises of a circular area having a fixed radius.

III. Parameters used

A. Chromosome

- For our genetic algorithm we are using a chromosome of length 144 where there are 6 height values (8 bits each), 6 corresponding angle values(7 bits each) and 6 different values for orientation from transformed positive x axis (9 bits each).
- Our chromosome structure is visualized in Figure 2 where h represents height value and a represent corresponding angle values and b represent the angles from positive x axis in transformed 6 co-ordinates of triangle 1 to 6. The height, angle and orientation values are binary comprising of 8, 7 and 9 bits respectively.
- These values were chosen considering the fact that height values can be from 1 to 256, angle values can be from 1 to 120 and orientation.i.e. rotation angle values from positive x axis can be 0 to 360. In our case all triangles covering the maximum area in the screen (at least, covering the most region inside the circle) is considered to be the optimum. This problem does not have any fixed optimum values. There can be multiple optimums.

B. Other parameters used

- Through experimentation we found that probability of mutation of 0.05 and probability of cross over of 0.9 works best for our GA.
- We have used CHC based selection with Lamda value of 2.
- We have used 30 different random seeds having values (1-30). We averaged the minimum, maximum and average fitness of our GA in every generation over these 30 random seeds and plotted them in figures that we will see later on in the report.

IV. Fitness calculation

A. Orientation calculation for each triangle

For fitness function calculation, we are considering the calculation of percentage of area covered inside a circle. For this we are using the Pygame library to create a grey color display of size 970 x 970 since it is the biggest window size that fit the screen of our device . After that, we draw a 256 x 256 grid there. Then we draw a circle having radius of 128 grid units where each grid unit is equivalent to 3.78 pixels. Finally, we start placing triangles from a starting location of positive x axis in clockwise direction of each of our six co-ordinates having values [506,527],[468,527],[465,486],[504,486],[466,453],[501,453] in our Pygame screen. We are considering all the triangles to be isosceles triangle since in case of ships the assets field of view is equally distributed from principal axis of rotation. Every triangle starts from alpha angle(named b1-b6 in 2) in from the positive x axis in clockwise direction where alpha can have values from 0 to 360. Each isosceles triangle can have an adjacent angle(theta named a1-a6 in 2) value from 1 to 120 degree and height values(named h1-h6 in 2) ranging from 1 to 256 units.

B. Drawing the triangles

For drawing the triangles using Pygame, we are required to have location of 2 corner points of each triangle sides along with one predefined co-ordinate point on the ship area from where the triangles will be originated. 2 adjacent sides of isosceles triangle has one point in common which is the center of the triangle. We calculated the co-ordinate of end points of each of 3 sides of triangles by solving 3 trigonometric equations for each side.We considered the trigonometric equation of a a line whose perpendicular makes and angle theta from positive x axis and that line is alpha angle away from positive x axis. In this way we do similar calculation for each triangles with respect to each given location point (A to E) on a transformed co-ordinate system(see Algorithm 1). Finally, we transform them with respect to the origin of the circle whose pixel location in the Pygame window and our original co-ordinate system is (485, 485), we can find this Algorithm 2.

Once the cor-ordinates of each sides end point is found we draw green color filled triangles using pygame's gfxdraw library's drawFilledTrigon() function.

Algorithm 1: Calculating the 3 corner co-ordinates of each triangle with respect to transformed co-ordinate system

Result: 3 corner Co-ordinates of a triangle in transformed co-ordinate system($x, yco-ordinatevalues$)
Input: List of heights, angles between two adjacent sides of each isosceles triangles and orientation angles.i.e. $\alpha frompositivexaxisintransformedco-ordinatesystem$,theta_angles
a1 = 270+α
a2 = $\alpha + theta/2$
a3 = 270+ $\alpha + theta$
foreach $theta\_angles\ in\ theta\_angles\ (where\ theta(i) > 0$ **do**
  xcos(a1) + ysin(a1) = 0
  xcos(a2) + ysin(a2) = height
  xcos(a3) + ysin(a3) = 0
end

Algorithm 2: Calculating the 3 corner co-ordinates of each triangle with respect to original co-ordinate system

Result: 3 corner Co-ordinates of a triangle in original co-ordinate system ($x, yco-ordinatevalues$)
Input: List of 3 corner co-ordinates of a triangle in transformed co-ordinate system.i.e. list of 3 ($x, yco-ordinatevalues$) and co-ordinate of a given pixel location on ship's rectangular area
foreach $indexinrangelen(co\text{-}ordinate_solutions)$ **do**
  $co\text{-}ordinate_solutions[index][0] = co-ordinate_solutions[index][0] + location_point[0]co-ordinate_solutions[index][1] = co-ordinate_solutions[index][1] + location_point[1]$
end

C. Area covered calculation

For area coverage calculation we simply calculate the number of green pixels inside the boundary of the circle and multiply that with pixel size.

$$CircularArea = 3.146 * (radius)^2 \quad (1)$$

For calculation of area covered inside circle by triangles, we use the following equation:

$$AreaCovered = pixelSize * trianglePixelCount \quad (2)$$

Here trianglePixelCount is the number of pixels whose RGB pixel value is green and are inside circular boundary. Finally for calculation of percentage of area covered, we use equation 3. We can see how our display looks with various height, theta, alpha values in the Figure 3 - 7

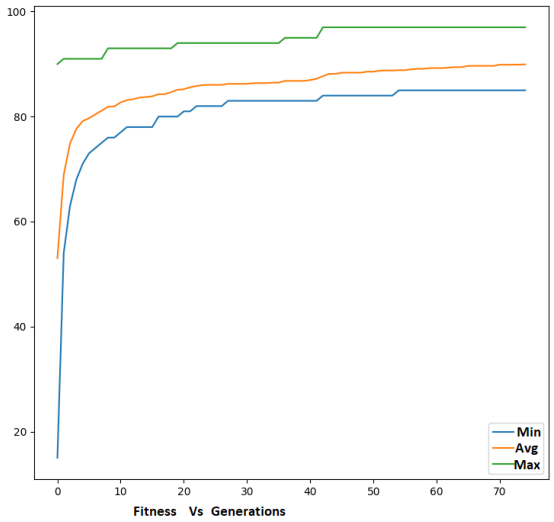

Fig. 3: Minimum, Average and Maximum Fitness Vs Generation plot averaged over 30 random seeds for each generation

$$Fitness = (AreaCovered/CircularArea) * 100 \quad (3)$$

It should be noted that the values that we have used in the code for area calculation is subject to change by a factor of horizontal or vertical grid size since we have done all the calculations considering a 256x256 grid over our display which is of size 970x970. Therefore, our horizontal or vertical grid size in this case is 3.78 pixels approximately.

V. Results

Upon usage of a population size of 50 with generation number 75 for our evolutionary algorithm, we see from Figure 3 that the fitness value start from 10 and within 20 generation it improves to 80 ,whereas, average fitness values start from 60 and reaches value greater than 80 within 5 generations. Moreover, the maximum fitness curve starts from a value of 90 and slowly reaches 95 within 43 generations. Here, the computational complexity of GA is 2^{144} which makes it an NP hard problem. Upon further experimentation with population size of 100 and generation number 150, we observe that average maximum fitness reaches a peak value of 97 within 60 generations. Average minimum fitness value reaches a peak value of 85 within 100 generations. Therefore, from Figure 9 there is a gradual improvement in the results when the population and generation size is increased proving that genetic algorithms are a reliable choice in case of complex problem set like maritime resource allocation.

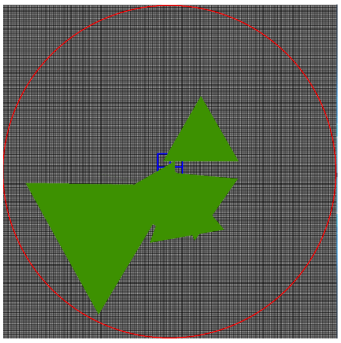

Fig. 4: height1-height6 = [100,50,50,50,50,50], theta1-theta6 =[300,0,60,60,60,60], alpha1-alpha6 = [1,190,5,50,300,90] where Generation number = 1 and Area covered = 48 percent

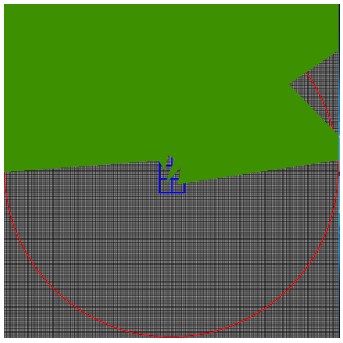

Fig. 5: height1-height6 = [114, 70, 231, 58, 171, 125] and theta1-theta6 =[71, 21,42,3,95,115], alpha1-alpha6 =[280, 289, 239, 296, 175,210] where Generation number = 18 and Area covered = 66 percent

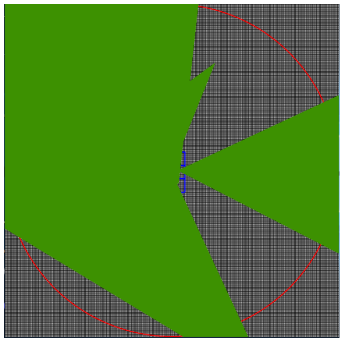

Fig. 6: height1-height6 = [127, 52, 201 ,169 ,109 ,169] and theta1-theta6 =[65, 116, 50, 21,87,65], alpha1-alpha6 =[235,114,236,56,346] where population size = 50 and Generation number = 37, Area covered = 69 percent

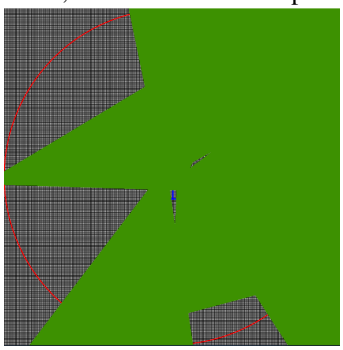

Fig. 7: height1-height6 = [139, 178, 177, 225, 123, 217] and theta1-theta6 =[96,115, 43,70,51,2],alpha1-alpha6 =[235,114,236,56,346] where population size = 50 and Generation number = 70, Area covered = 75 percent

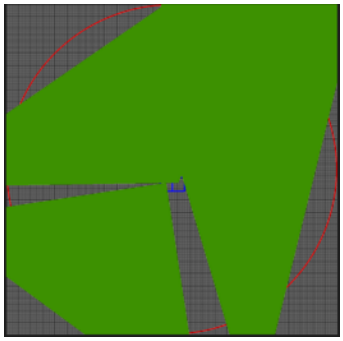

Fig. 8: height1-height6 = [221, 199, 255, 130, 31, 127] and theta1-theta6 =[37, 84, 63, 40, 73, 28], alpha1-alpha6 =[97, 239, 220, 195, 286, 215] where population size = 100 and Generation number = 103, Area covered = 82 percent

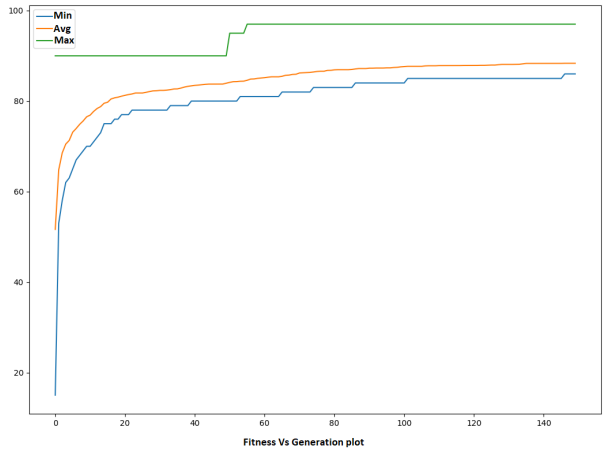

Fig. 9: Minimum, Average and Maximum Fitness Vs Generation plot averaged over 30 random seeds for each generation

VI. Conclusion

In conclusion we can say that evolutionary algorithms have highly reliable when it comes to searching through complex spaces for asset utilization for navy ships. Even within a short span of 150 generations it has showed promising performance by finding area coverage.i.e. fitness value of around 97 percent approximately. Here since the problem set is very complex and CHC selection based genetic algorithm is being used as an optimizer to find a set of height, theta and alpha that maximizes our area coverage. The idea is to cover most of the area considering the entire circular area to be the minimum area it can cover which can ensure the maximum safety for ships in a given circular region 9.

References

[1] Michael Adamowicz and Antonio Albano. Nesting two-dimensional shapes in rectangular modules. *Computer-Aided Design*, 8(1):27–33, 1976.

[2] Brenda S Baker, Edward G Coffman, Jr, and Ronald L Rivest. Orthogonal packings in two dimensions. *SIAM Journal on computing*, 9(4):846–855, 1980.

[3] Dayanne G Coelho, Elizabeth F Wanner, Sergio R Souza, Eduardo G Carrano, and Robin C Purshouse. A multiobjective evolutionary algorithm for the 2d guillotine strip packing problem. In *2012 IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2012.

[4] Paul C Gilmore and Ralph E Gomory. A linear programming approach to the cutting-stock problem. *Operations research*, 9(6):849–859, 1961.

[5] Paul C Gilmore and Ralph E Gomory. A linear programming approach to the cutting stock problem—part ii. *Operations research*, 11(6):863–888, 1963.

[6] Asyl L Hawa, Rhyd Lewis, and Jonathan M Thompson. Exact and approximate methods for the score-constrained packing problem. *European Journal of Operational Research*, 2022.

[7] EBCH Hopper and Brian CH Turton. An empirical investigation of meta-heuristic and heuristic algorithms for a 2d packing problem. *European Journal of Operational Research*, 128(1):34–57, 2001.

[8] Simon Illich, Lyndon While, and Luigi Barone. Multi-objective strip packing using an evolutionary algorithm. In *2007 IEEE Congress on Evolutionary Computation*, pages 4207–4214. IEEE, 2007.

[9] Stefan Jakobs. On genetic algorithms for the packing of polygons. *European journal of operational research*, 88(1):165–181, 1996.

[10] Qiang Luo, Yunqing Rao, and Deng Peng. Ga and gwo algorithm for the special bin packing problem encountered in field of aircraft arrangement. *Applied Soft Computing*, 114:108060, 2022.

[11] Silvano Martello, David Pisinger, and Daniele Vigo. The three-dimensional bin packing problem. *Operations research*, 48(2):256–267, 2000.

[12] Jeffrey R Sampson. Adaptation in natural and artificial systems (john h. holland), 1976.

[13] Anabela Simoes and Ernesto Costa. Chc-based algorithms for the dynamic traveling salesman problem. In *European Conference on the Applications of Evolutionary Computation*, pages 354–363. Springer, 2011.

[14] Xiang Song, CB Chu, YY Nie, and Julia A Bennell. An iterative sequential heuristic procedure to a real-life 1.5-dimensional cutting stock problem. *European Journal of Operational Research*, 175(3):1870–1889, 2006.

[15] Johannes Terno, Rudolf Lindemann, and Guntram Scheithauer. *Zuschnittprobleme und ihre praktische Lösung*. VEB Fachbuchverlag, 1987.