**USAGE OF AI IN MARINE TO FIND OPTIMAL ROUTE**

DSEWP ZG628T DISSERTATION

by

Gourav Chakraborty

BITS ID No. 2019hw04001

Dissertation Work carried out at

Wipro Technologies, Bangalore

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE

Pilani (Rajasthan) India

July, 2021

DSEWP ZG628T DISSERTATION

**USAGE OF AI IN MARINE TO FIND OPTIMAL ROUTE**

Submitted in partial fulfilment of the requirements of

M. Tech Data Science and Engineering Degree Program

by

Gourav Chakraborty

ID No. 2019hw04001

Under the supervision of

**Dr. Gopichand Agnihotram**

Dissertation work carried out at

Wipro Technologies, Bangalore

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE

PILANI (RAJASTHAN)

(July, 2021)

**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI**

**CERTIFICATE**

This is to certify that the Dissertation entitled **USAGE OF AI IN MARINE TO FIND OPTIMAL ROUTE**

and submitted by : Gourav Chakraborty ID No: 2019hw04001

in partial fulfillment of the requirements of DSEWP ZG628T Dissertation, embodies the work done by him/her under my supervision.

Signature of the Supervisor

Name

Designation

Date:\_\_\_\_\_\_\_\_\_\_

1. Introduction:

**Core Module:**

In the marine world there are lot of things in which automation can dig in and using intelligent methods reducing human errors,

One of the major automation problem among them is to automatically finding the optimal routs to traverse all the ports to minimize the distance to reduce the fuel cost

Problem statement goes like that suppose there are random amount of port at random location and we need to traverse each one of them with lowest amount of distance possible without creating any cycle.

There are lot of methods in optimizing those that includes Greedy and Dynamic programming methods as the number of nodes(Ports) in the Graph increases the time required to optimize the path is also more.

To solve this and find the current optimal route in realtime we are using Reinforcement Learning (RL) method , there are lot of RL algorithm we can choose of but most suitable for this task is Genetic Algorithm(GA) , but Genetic Algorithm is not designed to optimize the Graph traversing therefore we need to modify the GA to suit our needs

**Additional Module (Optional module):**

Additionally the Computer Vision model to avoid collision in which our model is trained using the dataset from the Simulation module in unity which contains 3 virtual camera and one Virtual speed sensor to get the realtime environment data out of , which is then used to train the Neural network model

The main problem which is faced to make neural network is to how to feed 3 images at one network, there are various method to do so the method I choose to make 3 different sequence Conv2d layer and Polling layer for each image and then flatten all those and concatenated vertically to make the dense layers

Architecture and problem I faced in the process will be discussed in the latter chapters (Note: This is an Experimental module so the success rate of this module is not confirmed till the end )

**Simulation Module**:

A simulation environment just like a game is created from scratch to gather in-game data which should be approximately similar to the real world in terms of physics and visual, all kind of visual effects are added to make it look close to the real-life including custom designed Shaders for water and its watery mechanics from scratch

**Tools Used:** Unity3d , Blender , C# ,DirectX ,Nvidia Phsyx & Shader, Python , Pytorch

1. **Chapter 1 Project Architecture:**

The project consist of two sub projects/modules first and the core module is to find the optimal path to traverse each and every Nodes of a fully connected graph or in this case “Ports”

A Reinforcement learning algorithm is built to find the optimal route in every constant interval of time by traversing each and every port

Fig 1.1:

**Unity 3d simulation Environment**

**Genetic Algorithm**

**(GA)**

**Port Traversal sequence File**

**Ship Location CSV file**

**Ports Coordinates CSV file**

Not in the scope of midsem

Our simulation Environment Generates two files one contains the coordinates of all ports and one contains the current location of the ship which is updated every constant interval of time, which file then feeded to the Genetic Algorithm the genetic Algorithm then decides the sequence of location in which our ship should go to minimize the fuel requirement.

The Second sub project or Sub module is an experimentation module which is an Additional part I shouldn’t be marked on the basis of that so this sub module contains a part in which our ship sails automatically avoiding the obstructers comes in the way since the system requirement of this model is so high so that it can’t be simulated in real time with the current knowledge I have but using the instance of still images to test the model I got some interesting results

Fig 1.2:

Key Bindings

Conv NN model

Speed

Image 3

Image1

Image2

Unity 3d Simulation Environment

So theoretically the Unity 3d environment sends out ut 3 images from around the ship and the ships current speed to the the NN model our NN model then splits out a key bindings to control the unity environment, this is still in the Experimental stages and not sure that this will work as fool proof as we wanted it to be , the architecture of the NN will be discussed in later chapters

1. **Chapter 2 the simulation Environment Unity 3d**

**Introduction to Unity 3d:**

Unity is an engine for creating games on multiple platforms. Unity was released by Unity Technologies in 2005. The focus of Unity lies in the development of both 3D and 2D games and interactive content. Unity now supports 27 different target platforms for deploying. The most popular platforms are Android, PC, and iOS systems.

* Unity is an integrated platform that is used as a gaming engine and framework.
* Unity allows you to develop once and publish everywhere.
* Although unity is considered to be more appropriate for creating 3D games, it can also be equally used to develop 2D games.
* In Unity, it is possible to develop games with heavy assets without depending on the additional frameworks or engines. It really enhances the experience of users.
* With the help of Unity, our game developers can access a wealth of resources like intuitive tools, ready-made assets, clear documentation, online community, etc. free of cost for creating exciting 3D contents in the games.
* Asset tracking and rendering, scripting are some of the features of Unity game development that we use in reducing time and cost.

**Scope of Unity in This project:**

For midsem the scope of unity is gathering data to feed our models so that our model could make our decisions.

Apart from the animations and the Game mechanics to make it visually look good there was no way to make it communicate it to our python model , so the solution to that problem was to communicate through Files, Generally for any particular instance of time our unity Simulation environment gathers the environment data and make a file out of it , which is then feeded to our model to make decision , please refer to the Fig 1.1 and 1.2 to find out the files created by our unity env

Please refer to the below Pics attached of the Environment:

1. **Chapter 3 Optimize Traversing the graph using Reinforcement Learning Method(Genetic Algorithm) (Core module)**

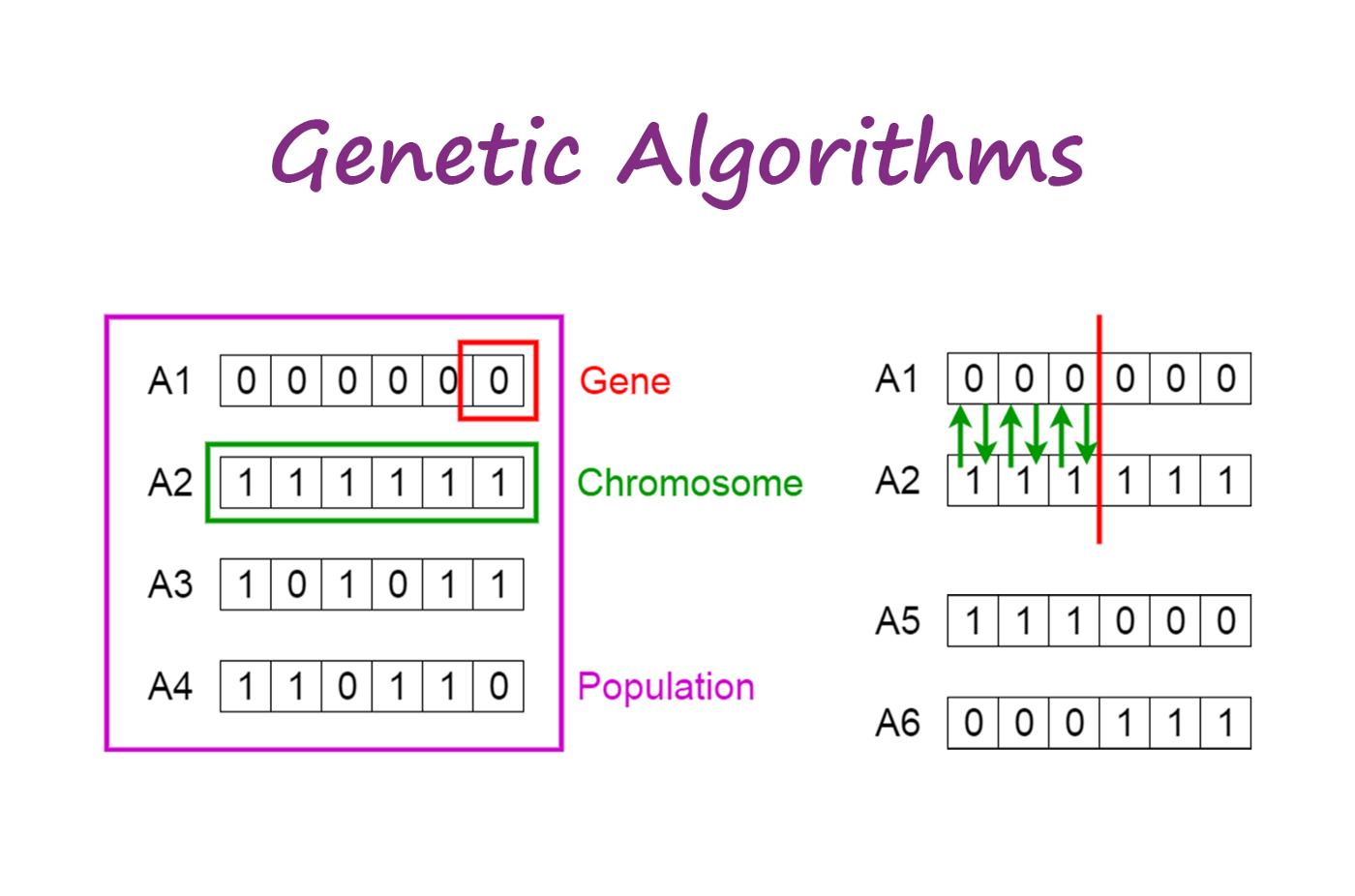
Lets suppose there is some shipments to make with the list of some Ports/Nodes which has unique coordinates our module decides the sequence of ports to be traversed which has least amount of distance hence reducing the Fuel requirement

By reviewing Fig 1.1 we can see that our simulation environment generates two files one consists all the coordinates of ports and one contain real time location of ship, when both is feeded into our GA model our GA model then splits out a optimal sequence in which the ship needs to travel to optimize the total distance and fuel cost

**3.1 Basics of Geneic Algorithm:**

A **genetic algorithm** is a search heuristic that is inspired by Charles Darwin’s theory of natural evolution. This algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation.

Fig 3.1



Notion of Natural Selection

The process of natural selection starts with the selection of fittest individuals from a population. They produce offspring which inherit the characteristics of the parents and will be added to the next generation. If parents have better fitness, their offspring will be better than parents and have a better chance at surviving. This process keeps on iterating and at the end, a generation with the fittest individuals will be found.

This notion can be applied for a search problem. We consider a set of solutions for a problem and select the set of best ones out of them.

Five phases are considered in a genetic algorithm.

1. Initial population
2. Fitness function
3. Selection
4. Crossover
5. Mutation

**Initial Population:**

The process begins with a set of individuals which is called a **Population**. Each individual is a solution to the problem you want to solve.

An individual is characterized by a set of parameters (variables) known as **Genes**. Genes are joined into a string to form a **Chromosome** (solution).

In a genetic algorithm, the set of genes of an individual is represented using a string, in terms of an alphabet. Usually, binary values are used (string of 1s and 0s). We say that we encode the genes in a chromosome.

Fig 3.2



Population, Chromosomes and Genes

**Fitness Function:**

The **fitness function** determines how fit an individual is (the ability of an individual to compete with other individuals). It gives a **fitness score** to each individual. The probability that an individual will be selected for reproduction is based on its fitness score.

**Selection:**

The idea of **selection** phase is to select the fittest individuals and let them pass their genes to the next generation.

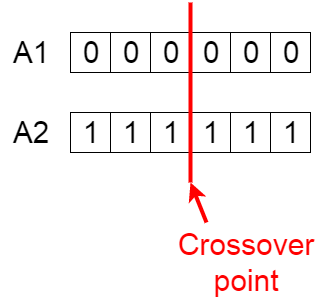
Two pairs of individuals (**parents**) are selected based on their fitness scores. Individuals with high fitness have more chance to be selected for reproduction.

**Crossover:**

**Crossover** is the most significant phase in a genetic algorithm. For each pair of parents to be mated, a **crossover point** is chosen at random from within the genes.

For example, consider the crossover point to be 3 as shown below.

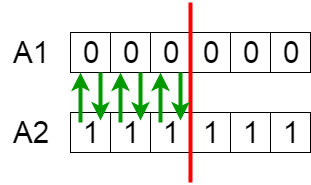
Fig 3.3



Crossover point

**Offspring** are created by exchanging the genes of parents among themselves until the crossover point is reached.

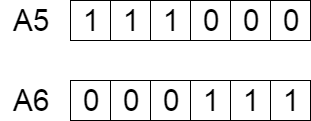
Fig 3.4



Exchanging genes among parents

The new offspring are added to the population.

Fig 3.5

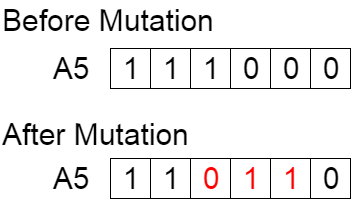


New offspring

**Mutation**

In certain new offspring formed, some of their genes can be subjected to a **mutation** with a low random probability. This implies that some of the bits in the bit string can be flipped.

Fig 3.6



Mutation: Before and After

Mutation occurs to maintain diversity within the population and prevent premature convergence.

**Termination**

The algorithm terminates if the population has converged (does not produce offspring which are significantly different from the previous generation). Then it is said that the genetic algorithm has provided a set of solutions to our problem.

**Comments**

The population has a fixed size. As new generations are formed, individuals with least fitness die, providing space for new offspring.

The sequence of phases is repeated to produce individuals in each new generation which are better than the previous generation.

**Pseudocode**

START  
Generate the initial population  
Compute fitness  
REPEAT  
 Selection  
 Crossover  
 Mutation  
 Compute fitness  
UNTIL population has converged  
STOP

**3.2 Usage of Genetic Algorithm in this project:**

As in this project each chromosomes of the population contains the port sequence in which it should traverse the only constraint in each chromosomes that there should be no duplicates of genes in that chromosome

Below example shows single **chromosomes in the population** and the size of population is 1000

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | 1 | 5 | 4 | 3 | 6 | 8 | 7 | 10 | 9 |

The **Fitness function** is used as a cost function , instead of Maximizing the fitness we are minimizing the cost function

Using Roulette weel method to chromosome is selected as Natural selection for crossover. Now the **Crossover method** is tweaked according to the need like

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P1 | 2 | 1 | 5 | 4 | 6 | 7 | 9 | 8 | 3 |
| P2 | 9 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 8 |

0 1 2 3 4 5 6 7 8

Let suppose crossing over at index 3 4 dividing both the parents P1 and P2 at point 3

Now we will just use the RHS part of the divided point and crossover with the original parent before division , to do that we need to first remove all those element that is present in the 2nd parent and then concatenate both

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| P1.1 | 2 | 1 | 5 | 4 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| P1.2 | 6 | 7 | 9 | 8 | 3 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| P2.1 | 9 | 7 | 6 | 5 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| P2.2 | 4 | 3 | 2 | 1 | 8 |

C1 = P1 x P2.2

C2 = P2 x P1.2

Where “x” operation is the crossover operation

Removing all the element of P1 that is present in P2.2 then concatenating both and similarly for the other crossover we get:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| C1 | 5 | 6 | 7 | 9 | 4 | 3 | 2 | 1 | 8 |
| C2 | 5 | 4 | 2 | 1 | 6 | 7 | 9 | 8 | 3 |

For the mutation technique we are using multi point Mutation where we are using any point and swap the data with any other point so that our unique constraint is still satisfied

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

0 1 2 3 4 5 6 7 8

Suppose index 2 is swapped with 6 we get the below mutated chromosome

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 7 | 4 | 5 | 6 | 3 | 8 | 9 |

0 1 2 3 4 5 6 7 8

This process goes on for total of 100000 generation max and terminates when there is no change in the least distance value for last 10000 generation

After termination the Final sequence is the Optimal sequence which has the optimized fuel requirement and can be used to traverse the ports

1. **Chapter 4 Self Sailing the Boat Using Conv Neural Network (Experimentation Module)**

**Note: This Module Is an experimentation module so there is no Guarantee of completion of model and I should not be marked on the basis of this**

This module is crated so that it could take real time frame Images from Left,Mid and Right sides of the ship in the simulation environment with its current speed as an Input and splits out the Key Bindings to Sale the boat , as this is an experimentation module there is no Guarantee of its accuracy or even the model will complete

The Conv NN Architecture which I used takes 3 images and one speed metric as an images and then splits out one of the below key bindings to sail the ship

* Null
* Up
* Down
* Right
* Left
* Up|Left
* Up|Right
* Down|Left
* Down|Right

Conv

Conv



Speed