Finding Optimal Location of Hospitals Using Hill Climbing and Genetic Algorithm

INTRODUCTION

Hill climbing is a type of local search algorithm in which the neighbour states and current states are compared, and if a better neighbour is found, the current node is changed from the current state to the neighbor state. The essence of hill climbing lies in its iterative process: starting with an initial solution, the algorithm evaluates its quality against a defined objective function. Through successive iterations, neighbouring solutions are explored by making small adjustments to the current solution. The algorithm then selects the most promising neighbour, gradually converging towards a local optimum.

Genetic algorithms (GAs) are a family of randomised search optimisation heuristics that are based on the biological process of natural selection. These algorithms start with an initial set of solutions called a population. Each individual in the population is called a chromosome and represents a solution to the problem. The chromosomes evolve through successive iterations, called generations. During each generation, the chromosomes are evaluated using some measures of fitness. To create the next generation, new chromosomes, called offspring, are formed by either a) merging two chromosomes from the current generation using a crossover operation or b) modifying a chromosome using a mutation operator.

In general cases, Genetic algorithms offer several advantages over hill climbing algorithms, mainly when dealing with complex optimisation problems, like escaping local optima, i.e., GA is powerful enough in navigating complex search spaces due to their inherent mechanisms of crossover and mutation. These operators facilitate the exploration of diverse regions within the search space, significantly increasing the probability of locating the global optimum. GA can handle complex landscapes unlike hillclimbing, which finds in difficult to navigate such landscapes and may get trapped in a suboptimal peak; GA suited for parallelisation, where the evaluation of multiple individuals in a population can be done simultaneously on different processors.

Our project delves into the application and refinement of the hill climbing optimisation algorithm and also further uses GA; aiming to find an optimal location for a hospital in the IISERB campus.

OBJECTIVE

To identify whether the hospital's current location inside IISERB is optimal and to suggest an optimal location if not using hill climbing and genetic algorithms.

DATA

The data for this project was generated by combining using OpenStreetMap and QGIS software. The IISERB map is obtained from OpenStreetMap (https://www.openstreetmap.org/search?query=iiser%20bhopal#map=15/23.2886/77.2741). Using QGIS, we first remove the unwanted portions from the map. We are to focus on the portion pertaining to our study area. Further, centroids of all the buildings on the map are generated. This was done by using the polygon centroid tool in the QGIS software. We can then add the latitudes and longitudes of all these centroids. The distance matrix is also generated, which contains the distances between every pair of buildings (centroids) on the map. A dataset that contains all the buildings and their corresponding longitudes and latitudes is then saved as a CSV file. This dataset is used in the hill climbing code to obtain the optimal point. The distances were calculated using the Manhattan distance formula.

Centroids of all buildings in IISERB Map (generated using QGIS)

METHODOLOGY

Hill Climbing to find the optimal location.

The algorithm used in Hill Climbing begins with a first answer and iteratively refines it with little adjustments to make it better. These adjustments are predicated on a heuristic function that assesses the solution's quality. The algorithm makes these little adjustments until it hits a local maximum, at which point the present set of movements can no longer be improved upon.

Pseudocode:

function Hill-Climb(problem):
current = initial state of problem
repeat:
neighbor = best valued neighbor of current
if neighbor not better than current:
return current
current = neighbor

A good termination criterion would be when none of the neighbors have a higher objective value than the current state.

The starting point is the current location of the health centre with coordinates **(23.28540615, 77.27402556)**. The neighbors are also found using the Manhattan distance. The cost functions help in generating an efficient point at a minimal distance from buildings. Iterations are performed till no further improvements are obtained. In each iteration, the better neighbour cost is returned as a result, which represents the sum of the total distances between all the buildings and the hospital in that particular location. The algorithm then moves to the least cost found, thereby finding the configuration that helps reduce the travel distances.

Random restart is done to arrive at the optimal point coordinates for placing the health centre. Considering the time complexity, the iterations were done five times, and the results were obtained. The coordinates are then plotted and visualised in an IISERB map using QGIS plugins.

CampusSpace Class

The implementation utilizes a 'CampusSpace' class to represent the campus environment and handle hospital optimization.
The class includes methods for initializing hospitals, calculating available spaces, performing hill-climbing optimization, computing costs, and generating neighboring configurations.

Limitations of Hill Climbing

While using hill climbing algorithm to generate the optimal results, it should be kept in mind that there are some limitations associated with performing this algorithm.
It includes how the algorithm can possibly get stuck at local optima. Suboptimal solutions are better than their neighbours but definitely worse than the global optimum. Another limitation is how the algorithm finds it difficult to overcome plateaus and flat regions; it has no uphill moves.
If there are multiple peaks and valleys, it is possible that the global optimum maybe missed. The quality of the final solution is dependent on the initial solution chosen. These limitations can be overcome by the use of genetic algorithms to generate a much more accurate result.

Genetic Algorithm

Genetic algorithms offer a powerful alternative to hill climbing for tackling complex optimisation problems. Their ability to escape local optima, handle intricate landscapes, and leverage parallelisation make them a valuable tool in various AI applications. Genetic algorithms can search an entire space and jump between places through crossover and mutation, allowing them to find global solutions. It is performed over the same dataset obtained by the steps done earlier in QGIS. It is then used to produce a more optimal location of where the hospital should ideally be placed in the IISERB campus.

Optimal location using Genetic Algorithm

After loading the latitude and longitude data, a campus-bound dictionary is created with the maximum and minimum values determining the outermost boundaries encompassing all the data points.

The first step is to initialise the population. A population of potential hospitals is created randomly. Euclidean distance is used to find the distance between the current hospital and the rest of the buildings in each iteration. The mean of all these distances is then taken, and the least distance is selected. The fitness function calculates the mean distance of the current hospital

location. We will calculate the fitness score from this function. The roulette wheel criterion is further used to select parents for crossover.vr. Since we want the minimum distance between the buildings, we take the reciprocal of the fitness score and then normalize it. Small fitness scores get higher probabilities and then are normalised so that the sum gives 1. Based on these adjusted probabilities, individuals are then chosen randomly.

Crossover is then done with the selected parents by a crossover rate. It gives the probability of performing the crossover between the two parents. For each pair, it generates a random crossover point. A crossover criterion is given such that if the crossover rate is greater than the generated number then crossover takes place, else not. It then creates two offspring by swapping the coordinates beyond the crossover point between the parents. This crossover is applied with some probability, and the offspring replace the parents in the new generation. Otherwise, the parents are simply copied to the next generation. These offspring are added to the list. A mutation is also done by generating a random number, and if it is less than this number, the mutation is carried out, and all these offspring are added to the list. With a small probability, the algorithm randomly modifies the coordinates of offspring locations. This adds some variation to the population and helps explore different potential hospital placements. The modification follows a normal distribution with a mean of 0 and a very small standard deviation (0.001). This ensures the changes are minor and keep the locations within reasonable boundaries. Finally, optimisation is done to obtain the final optimal location.

Pseudocode

The pseudocode for the Genetic Algorithm can be illustrated as follows:

1. Initialize Population (population_size, grid_size):
- Generate a random location within the grid_size.

2. Evaluate Fitness (population):
FITNESS-FN a function that measures the fitness of an individual
- For each individual in the population:
- Calculate the total distance to all buildings using the distance_matrix.

3. Selection (population, fitness_scores):
- Use roulette wheel selection to choose two parents based on their fitness scores.

4.Crossover (parent1, parent2):
- Randomly select a crossover point on the location coordinates.
- Create a child by combining the first part of parent1 with the second part of parent2.

5. Mutation (child, mutation_rate):

- With a probability of mutation_rate, randomly generate a new value within the grid_size.

if (small random probability) then child ← MUTATE(child)

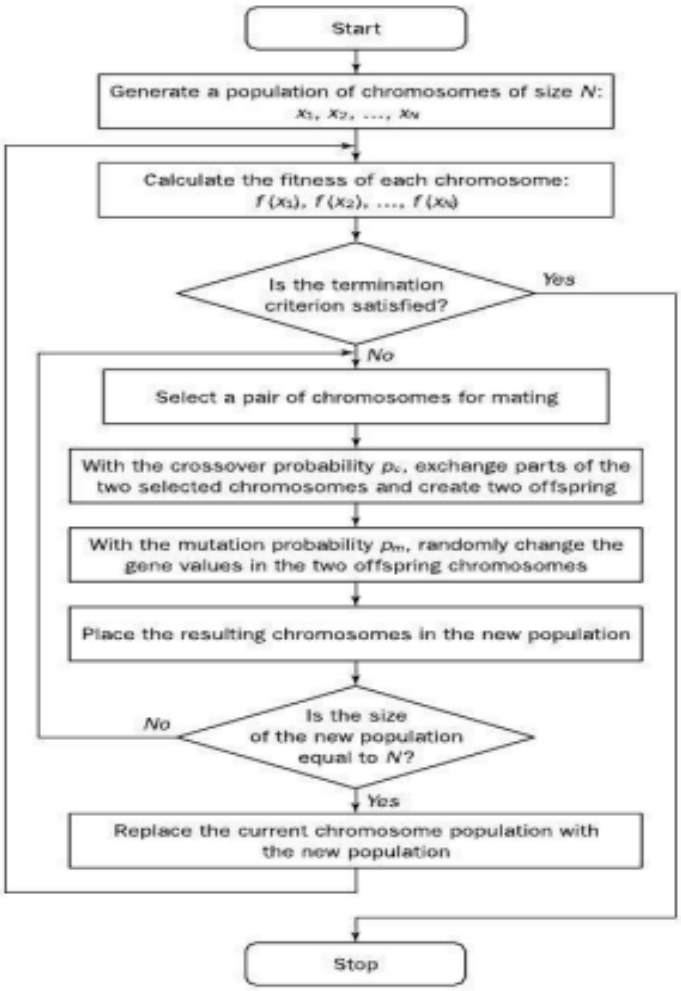
add child to new population

population ← new population

6. Return Best Solution (population):

- Find the individual in the population with the lowest fitness score.

- Return the best solution (location coordinates).



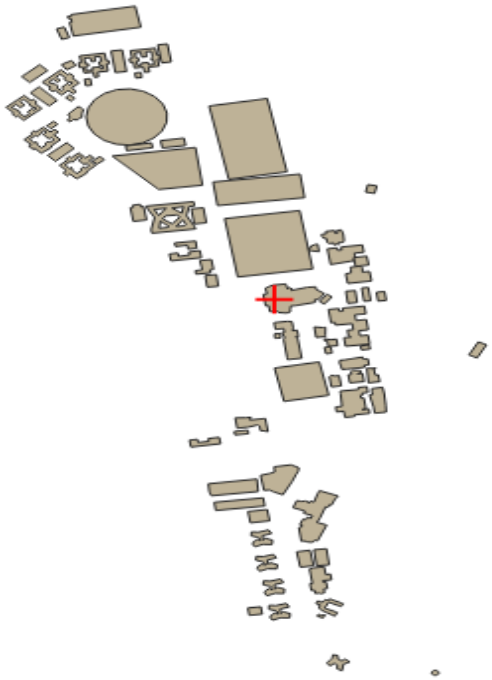
Flowchart illustrating the genetic algorithm.

Combined Approach

To summarize, we basically tried a combined approach to the dataset to provide a comprehensive optimization strategy. Both the hill-climbing algorithm and the genetic algorithm were applied to the campus dataset. The resulting hospital locations from both methods will be now compared to identify the most suitable placement that optimizes healthcare accessibility across the campus.

RESULTS AND FINDINGS

The primary objective of this project was to identify whether the current location of the hospital inside the IISER Bhopal campus is at an optimal location or not. We tried to analyse this using two popular algorithms, hill climbing and genetic algorithm. The results of both algorithms proved that the current location of the hospital inside the campus is non-optimal. Therefore we tried to find the most optimal location. The aim was to find a potential location at an ideal distance from all the buildings inside the campus. As a result, the **hill climbing algorithm** gave the location coordinates **(23.288106825694 77.27439926997496)**, and the **Genetic algorithm** gave **(23.28810666, 77.274589)** as the optimal location. Both of these locations are near the lecture hall complex inside the campus. Therefore, we can conclude that the ideal location for the hospital inside IISERB is near the LHC, which is at the minimum possible distance from all other buildings inside the campus. The images showing the optimal location in the map are shown below:



Optimal location using Hill Climbing plotted in IISERB Map (Crossmark indicates LHC as the optimal point)

Optimal location using GA plotted in IISERB Map (Crossmark indicates LHC as the optimal point)

CONCLUSION

The integration of multiple optimization approaches, including hill-climbing and genetic algorithms, enhances the robustness and effectiveness of hospital location optimization within the campus environment. By leveraging both methodologies, institutions can achieve improved healthcare accessibility and operational efficiency, contributing to the well-being of the campus community.

This report outlines a comprehensive approach to optimize hospital locations on campus, providing a foundation for further research and practical implementation in diverse campus environments.

REFERENCES

1. Liu, Jingkuang & Li, Yuqing & Li, Ying & Chen, Zhibo & Lian, Xiaotong & Zhang, Yingyi. (2022). Location optimization of emergency medical facilities for public health emergencies in megacities based on genetic algorithm. *Engineering, Construction and Architectural Management.* 30. 10.1108/ECAM-07-2021-0637.

2. Jorge H. Jaramillo, Joy Bhadury, Rajan Batta, On the use of genetic algorithms to solve location problems, *Computers & Operations Research,* Volume 29, Issue 6, 2002, Pages 761-779, ISSN 0305-0548, https://doi.org/10.1016/S0305-0548(01)00021-1

3. J. Bhadury, J. Jaramillo and R. Batta "On the Use of Genetic Algorithms for Location problems" *Computers and Operations Research*, Vol. 29, 761-779 (2002). doi:10.1016/S0305-0548(01)00021-1

4. Genetic Algorithms - GeeksforGeeks

5. Artificial Intelligence: A Modern Approach Third Edition by Stuart Russell and Peter Norvig.

6. Tabassum, M., & Mathew, K. (2014). A genetic algorithm analysis towards optimization solutions. *International Journal of Digital Information and Wireless Communications (IJDIWC)*, *4*(1), 124-142.
7. Zaritsky, Assaf & Sipper, Moshe. (2004). The Preservation of Favored Building Blocks in the Struggle for Fitness: The Puzzle Algorithm. Evolutionary Computation, IEEE Transactions on. 8. 443 - 455. 10.1109/TEVC.2004.831260.

GitHub Link:
https://github.com/gourihari/Optimal-hospital-location-using-Hill-Climbing-and-Genetic-Algorithm-in-IISERB-Campus

Contributions by team members :

Bhavana M R : Replicating the code, data generation using QGIS, coding, writing report (25%)
Karthika P S : Replicating the code,data generation using QGIS, coding, writing report (25%)
Gouri H : Replicating the code, data generation using QGIS, coding, writing report (25%)
Himadri Sonowal : Replicating the code,data generation using QGIS, coding, writing report (25%)