CS 7641 Assignment 2: Randomized Optimization

Nimesh Chudasama
nimesh@gatech.edu

Abstract—This paper investigates the relationship between 4 optimization algorithms. In particular it highlights the differences between Random Hill Climbing, Simulated Annealing, MIMIC and Genetic Algorithms. This paper will highlight the key differences between each algorithms and how they behave on different optimization problems to gain a better understanding of each algorithm. All analysis was done through Python using mlrose to implement the algorithms and matplotlib to visualize their respective performance.

1 ALGORITHMS AND PROBLEMS

Three optimization problems were selected to highlight difference between each respective algorithm. In this paper we have selected Max K Color, Knapsack, and the 8 Queens problems to run the 4 different algorithms.

In addition we trained a Neural Network using random hill climbing, simulated annealing and genetic algorithms to see how the different algorithms compared to the MLP Classifier in SKLearn.

1.1 Random Hill Climbing RHC

RHC searches for optima by searching by starting at a position randomly and moves up to the peak. It declares the peak a global optima and restarts until it's gone through all its iterations.

The highest peak it sees is the global maxima.

1.2 Simulated Annealing SA

SA basically samples a new point within the instance space, and jumps to a new sample with a probability dependent on temperature. If the fitness of a neighbor is greater than or equal to the current instance, move to that with a probability.

Over time decrease T, so the algorithm has a chance to reach the global maxima. As T decreases, the harder it is to find a global maxima and climb steeper "hills"

1.3 Genetic Algorithm GA

GA finds optima uses a biologically based process. It initializes a population with random states. A predefined top percentage moves on to the next generation and mutations occur within those population.

Over more and more iterations the mutations evolve to the optima.

1.4 MIMIC

MIMIC uses structure of the problem by finding optima by evaluation probabalistic density function. It directly models the distribution and updates every iteration to lead to the global maxima.

2 MAX K-COLOR

The problem is to color adjacent vertices such that no neighboring vertex has a color similar to another neighboring vertex.

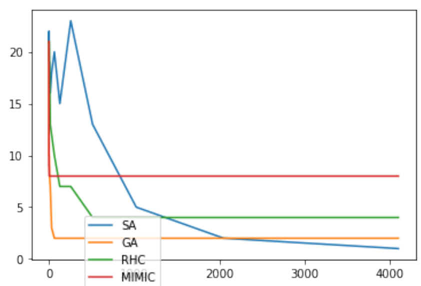

Figure 1—Iteration vs. Fitness

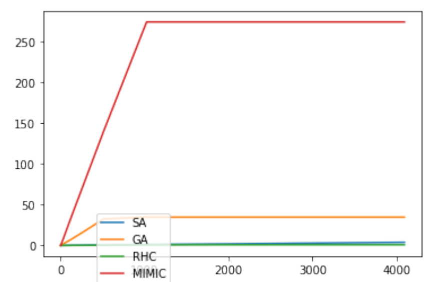

Figure 2—Iteration vs. Time

2.1 Summary

Based on seeing both the fitness and time, one can conclude Simulated Annealing is the best algorithm for this problem. Simulated Annealing did do worse than the other, but at iteration 2048 it surpassed GA and remained having a better fitness than GA.

In addition to the fitness level, it also boasts one of the lowest times.

3 KNAPSACK

The Knapsack problem is described as follow "Given a set of items, each with a weight and a value, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible" **knapsack2021**, **knapsack2021**

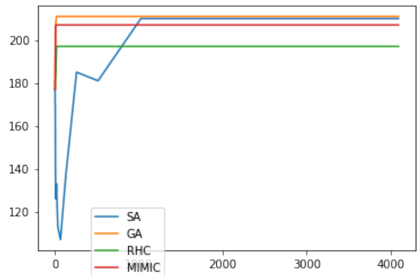

Figure 3—Iteration vs. Fitness

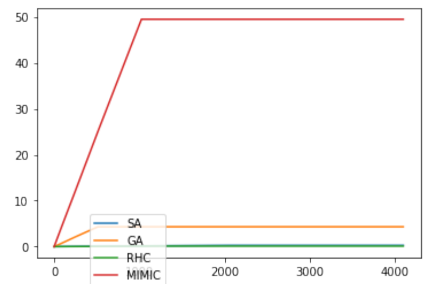

Figure 4—Iteration vs. Time

3.1 Summary

Based on seeing both the fitness and time, one can conclude GA is the best algorithm for this problem. The Genetic Algorithm quick converged to a maximal fitness quicker than other algorithms.

In addition to the fitness level, it also boasts the 3rd lowest time.

4 EIGHT QUEENS

The 8 Queens problem is to have a queen along each column of a chessboard and try to minimize the number of queens trying to attack each other.

4.1 Summary

Based on seeing both the fitness and time, one can conclude SA is the best algorithm for this problem. The Genetic Algorithm quick converged to a maximal fitness quicker than other algorithms.

In addition to the fitness level, it also boasts the lowest time.

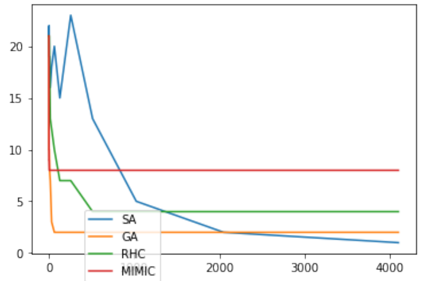

Figure 5—Iteration vs. Fitness

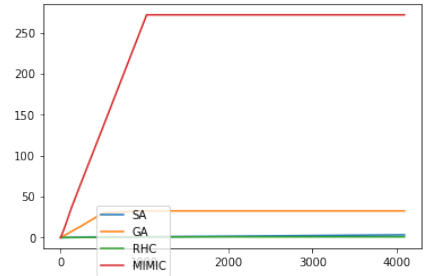

Figure 6—Iteration vs. Time

5 NEURAL NETWORK

We also trained a neural network using RHC, SA and MIMIC and obtained accuracies for all of them. We compared it against a baseline MLP SKLearn algorithm.

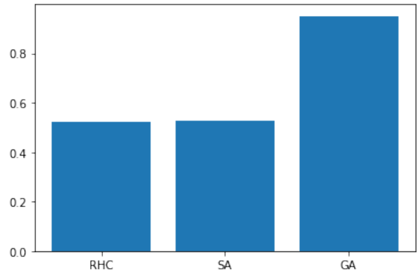

Figure 7—Test Accuracy

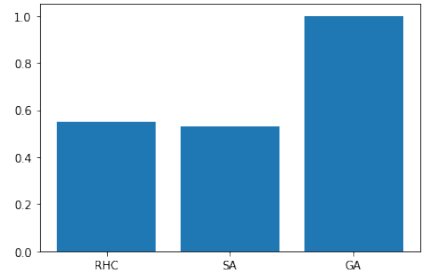

Figure 8—Train Accuracy

5.1 Summary

This shows that for a binary classification task, GA was the only one who reached 100% accuracy for train and reached in the upper 90s for test. This is close to the 100# accuracy we achieved using SKlearn's MLP Classifier.

6 REFERENCES

1. @inproceedingsknapsack2021, author = Geeksforgeeks, title = 0-1 Knapsack Problem | DP-10 - Geeksforgeeks, booktitle = https://www.geeksforgeeks.org/0-1-knapsack-problem-dp-10/), year = 2021