Algorithm Comparative Analysis for Randomized Optimization

Xiangnan He
Oct 15, 2017

I. Abstract

Four random search algorithms are analyzed in this study, randomized hill climbing, simulated annealing, genetic algorithm, and MIMIC. In the first part, the first three algorithms are used to find good weights for a neural network, which is trained on the dataset in Assignment #1 – Breast Cancer dataset, and compare with the backpropagation neural network used in Assignment #1. In the second part, the four algorithms are used to solve three optimization problems, including Knapsack Problem, Continuous Peaks Problem, and Traveling Salesman Problem. We show the circumstances for using each algorithm and how they look like in the time consumption. ABAGAIL library are used across the project.

II. Introduction

For problems that are not differentiable or continuous, random search (RS) algorithms are frequently used to get the global optima, without requiring the gradient of the problem to be optimized. We discuss four random optimization algorithms: Randomized Hill Climbing (RHC), Simulated Annealing (SA), Genetic Algorithm (GA), and Mutual Information Maximizing Input Clustering (MIMIC). In this study, ABAGAIL library is used for the analysis [1].

RS methods will maximize the fitness function, *F*, and let X be a position in the search space, the basic RS algorithm will work in the steps as follows:

1) Initial X at a randomly selected position in the search space;
2) Check if a termination criteria has been met (iteration limit or sufficient fitness), repeat:
    a. Sample a new location Y from the hypersphere of a given radius surrounding the current position X.
    b. If F(Y) > F(X) then move to the new position by setting X = Y.

1. GA

Genetic algorithm maximize the fitness function based on a natural selection process that mimics biological evolution. The algorithm modifies the population of solutions repeatedly. In each iteration, GA randomly selects individual solutions from the pool of population as parents to produce the children for the next generation/iteration. With growing iterations, the population will evolve to global optima. Children are produced by mutating and mating different parents from the population. One of the drawback for GA is that it doesn't scale with the problem complexity. When the number of elements becomes larger, the search space increases exponentially. The three parameters we studied are population, mate, and mutate.

2. RHC

Randomize hill climbing searches for global optima by randomly starting at a new position to avoid getting stuck in local optima. At each new position, it moves toward neighbors that have higher fitness value. The random start increases the chance of getting the global optima. Eventually, it stops after iteration limit or adequate fitness function has been achieved.

3. SA

Stimulated annealing comes from metallurgy where ductility in metals can be improved by heating to high temperature and then slowly cooled to keep its structure. This algorithm maximizes the fitness function values by evaluating the neighbor's fitness value. If the neighbor is better, it becomes the next point. On the other hand, if the neighbor is worse, the algorithm will calculate the probability of moving to the neighbor based on the acceptance function, which is a function of initial temperature and cooling factor. SA move to worse points with a lower probability during it exploration in the neighbor spaces. With repeating the process, SA is able to reach the global optima, and the time cost is higher.

The probability is $\exp((f(x_0) - f(x))/T)$, where f is the fitness function, x_0 is the current position, and x is the neighbor, and T is the temperature. The SA in ABAGAIL mainly uses initial temperature at 1E11 and cooling factor of 0.95.

4. MIMIC

MIMIC searches for global optima by evaluating probability density functions. It then utilizes the probability densities to build structure of the solution space at each iteration. It will generate a better solution structure and sample the points from the solution space to find global optima. There are two main components in MIMIC: 1) a randomized optimization algorithm to search for a region with high likelihood of optima; 2) an effective density estimator to capture a vast variety of structure on the input space [2].

III. Data analysis and preprocessing

1. Data analysis and preprocessing

Breast Cancer dataset from assignment #1 was chosen to perform the analysis. The weights of the neural networks are optimized with three RS algorithms, including RHC, GA, and SA. The data was previously analyzed with neural network with backprobagation algorithm as the method to optimize its weights.

There are total 569 instances, 30 attributes, and the labeling of the data is “Diagnosis” column with M = malignant and B = benign. The classification of M and B are replaced by 1 and 0. [3]

The instances are split into training set and testing set. The former contains 398 instances (70%), whereas the later contains 171 instances (30%). The results generated by the algorithms are plotted with Minitab.

IV. Results

Part I. Neural Network with RHC, SA, and GA

The score used in this study will be the accuracy. We choose to evaluate the algorithms at iterations 10, 100, 500, 1000, 2500, and 5000.

As we can see from Figure 1, the RHC algorithm score is well above 0.8 after less than 1000 iterations. The testing score is higher than training score at the beginning (< 1000 iteration), but the training score outgrow testing score very quickly at 1000 iterations. And then the difference between training and testing scores are stabilizing.

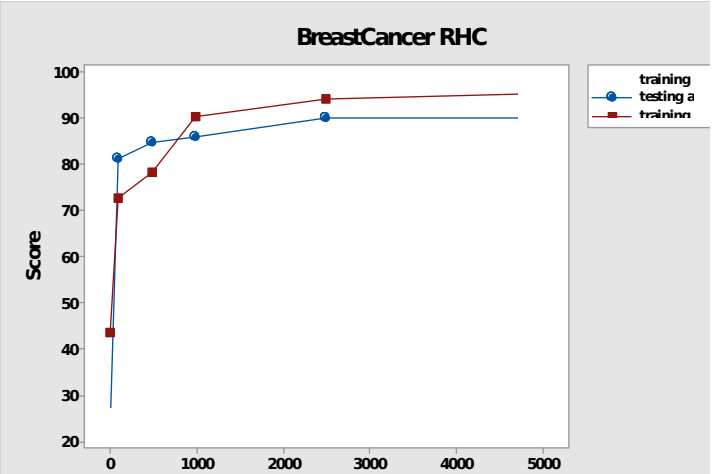

Figure 1. RHC score vs iteration plot

SA with cooling factors of {0.15, 0.25, 0.35, 0.45, 0.55, 0.65, 0.75, 0.85, 0.95} are trained and evaluated. From Figure 2 (left), the training and testing score grow together, at the beginning, it is hard to tell which one is higher, as the iteration goes, the training score is stabilizing and higher than testing score. Overall, the higher the cooling factor, the higher the score. The higher the cooling factor, the slower the temperature drops, which means the acceptance probability decrease slower, and then it is more possible to accept worse positions instead of always moving to other higher fitness points and get trapped at a local optima. From Figure 2(right), training score is plotted with different cooling factors. At the beginning, the training score is low, and then it grows to above 0.8 and the score at different cooling factors are different, therefore, there is a spread of scores when the temperature is not low enough.

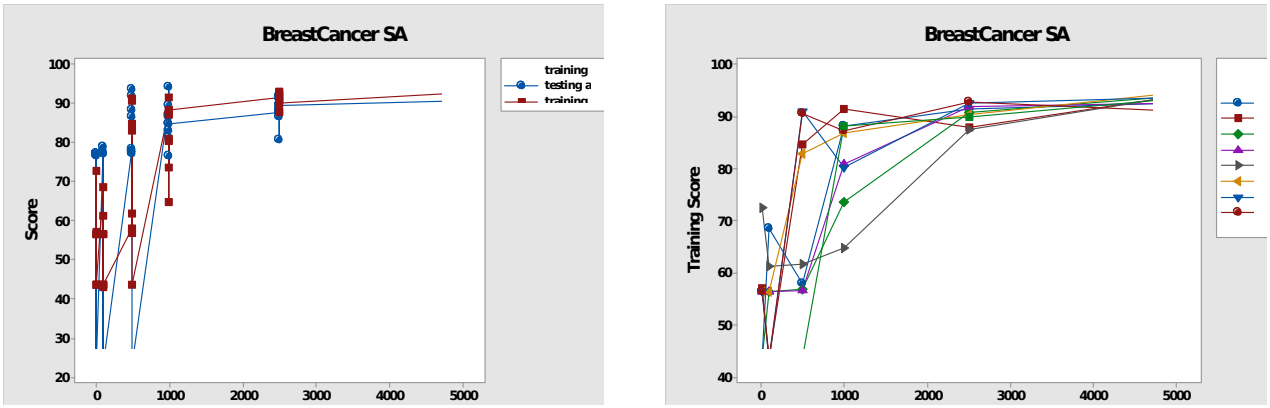

Figure 2. BreastCancer SA score vs iteration plots. (Left) SA training and testing score plot with different cooling factors, testing score labeled in blue and training in red. (Right) SA training score plot with different cooling factors.

GA with different population, mate, mutation combinations are evaluated, including (10, 5, 2), (20, 10, 5), (50, 25, 10), (100, 50, 10), (200, 100, 20), and (500, 350, 50). The population size is the survived individual number from the iteration, and mate is the crossover number of the mating between two parents. Mutate is the number of random modifications. From Fig. 3 (left), the plot contains both testing and training score, as well as the different combinations. Training score is overall higher than test score as they grow with increasing iterations. From Fig. 3(right), training score is plotted with different parameters, with lower population, mating, and mutation, the score is lower, and the 10_5_2 has very poor score compared with others.

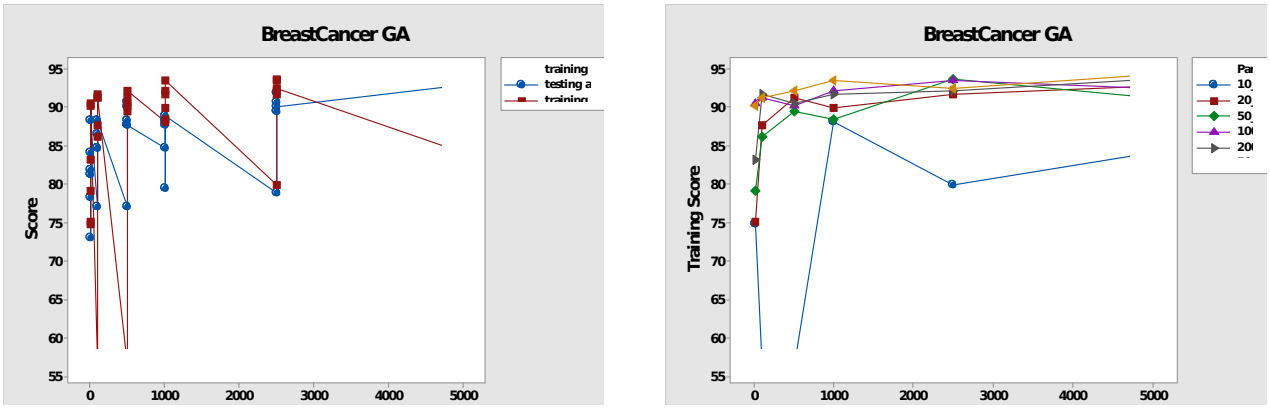

Figure 3. BreastCancer GA score vs iteration plot. (Left) GA training and testing score plot with different population-mate-mutation combinations, testing score labeled in blue and training in red. (Right) GA training score plot with different population-mate-mutation combinations.

As shown in Figure 4, the training time for GA grows very fast with increasing population, mating, and mutating parameters. And the time increases significantly with more iterations. The test time is not changing so much, so we will only focus on the training time in this study.

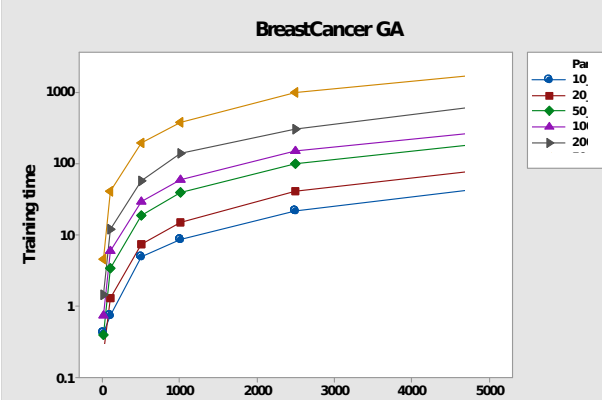

Figure 4. BreastCancer GA logarithm training time vs iteration plot.

When comparing the three algorithms, we can see Fig 5 (left) and (right), the training and testing score grows above 0.8 at 1000 iteration. SA is set at 0.95 cooling factor, GA has a population of 500, mating at 350, and mutating at 30. RHC algorithms has the highest training and testing scores at 5000 iterations. GA and SA are close on the training scores, and SA is a little higher compared with GA on the testing score. In Fig 5(right), GA costs orders higher than the time spent on RHC and SA. SA is slightly higher than RHC, due to the fact that we run different cooling factor on SA, whereas RHC doesn't have parameter change.

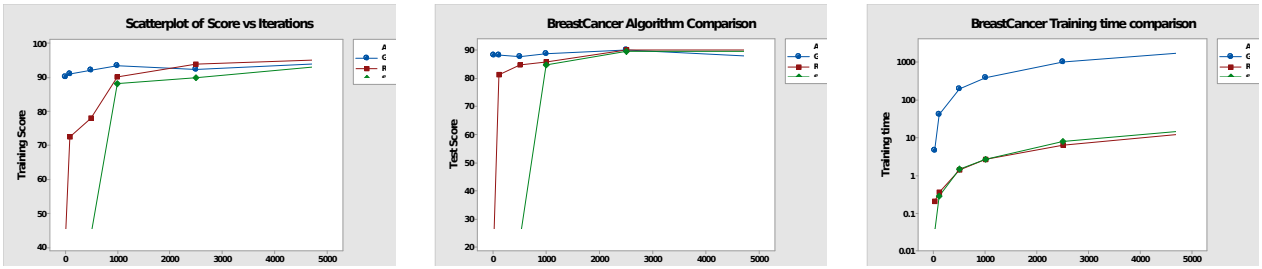

Figure 5. BreastCancer algorithm comparison plots among SA, GA, and RHC, including (Left) training score, (mid) testing score, and (right) training time vs iteration.

Sklearn MLPClassofier (backpropagation) was used in assignment #1 neural network to calculate the weights, and the test score was 0.892. The best training score achieved in this study is above 0.95, and the test score is 0.90, with RHC algorithm at 5000 iterations. This is better than the MLPClassifier, which suggesting the problem doesn't need a complex structure, instead a And in assignment 1, 7 attributes were chosen to better train the data, however, here we used 30 attributes all together, so if we choose the same 7 attributes, the score will be even better.

Table 1. BreastCancer neural network algorithm test score comparison.

Algorithms	Backprop	RHC	SA	GA
Test score	0.892	0.90	0.876	0.894

Part II. Three Optimization Problems

1) Knapsack Problem

Knapsack is a problem in combinatorial optimization. For example, to be able to maximize the capability of the container, we need to select from a set of items, which has items with different weights and value. By choosing the combination, we need to make sure the total weight is less than or equal to a limit, at the same time, the total value should be as large as possible. The Knapsack problem here is defined in ABAGAIL as follows:

Number of items: 50
Copies of each item: 5
Max weight: 100
Max volume: 100
Knapsack volume = max volume * number of items * copies * 0.4

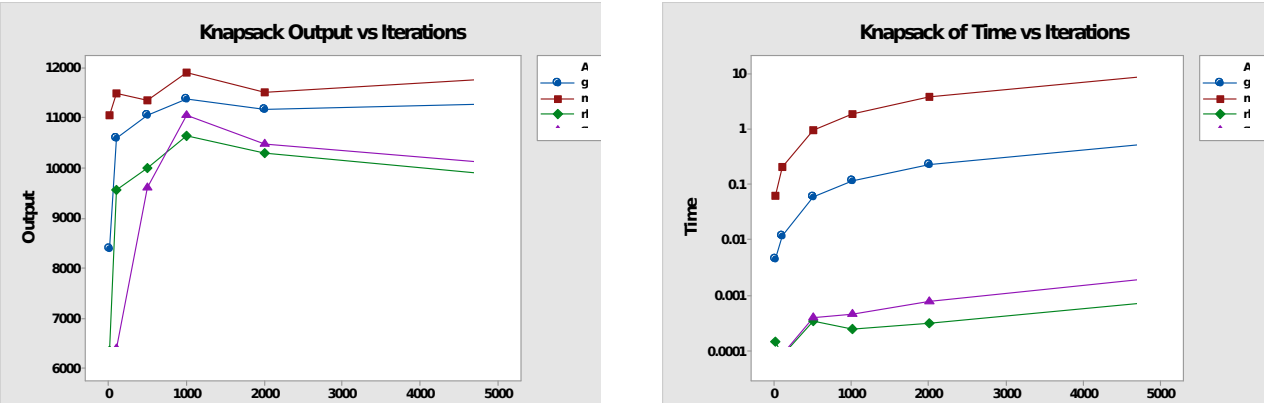

Figure 6. (Left) Knapsack output vs iterations plot. (Right) Knapsack time vs iteration plot.

From Fig. 6(left), the comparison of the four algorithms showing that, the MIMIC algorithm is better at optimizing the Knapsack problem with the highest fitness function output, then GA, RHC, and SA. RHC and SA even show a decreasing trend, due to complex structure of the problem. MIMIC and SA is well suited to the complex structure problem. Because the Knapsack requires building of complex structure to understand the how well each path performances. From Fig. 6(right), the cost of time shows, MIMIC is much higher than GA, and much higher than RHC and SA.

For MIMIC, we used the sample number and the numbers to keep (“Tokeep”) as the first two parameters in ABAGAIL MIMIC algorithm, such as 5(Tokeep)/10(sample#), 10/20, 20/40, 40/80, 80/160, 20/200, 40/200, 80/200, 160/200, and 180/200. As shown in Fig. 7, the highest fitness function output is at 20/200 and the lowest is at 5/10, due to lower sample number. When the ratio of “Tokeep” is smaller, but the sample is sufficiently large, the output will be better. The samples are run at 5000 iterations.

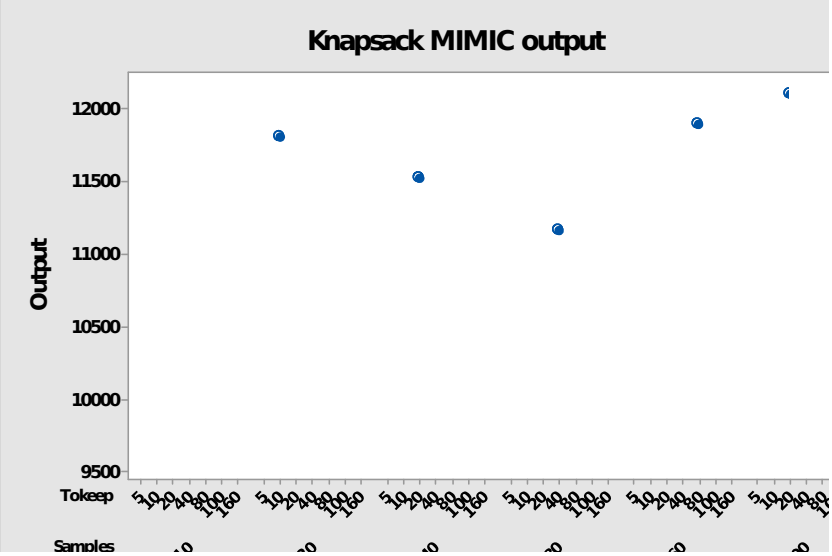

Figure 7. Knapsack MIMIC plot with different sample and keep combinations.

2) Continuous Peaks Problem

The continuous peak problem in ABAGAIL are used with N = 100 and T = 10. From Fig. 8(left), SA works best by generating the highest fitness function output compared with other algorithms, MIMIC, GA follows, and RHC is the worst. SA is run with initial temperature at 1E11 and cooling factor at 0.85. In Fig. 8(right), the time at each iteration, MIMIC and GA are orders higher than SA and RHC. MIMIC is also significantly higher than GA.

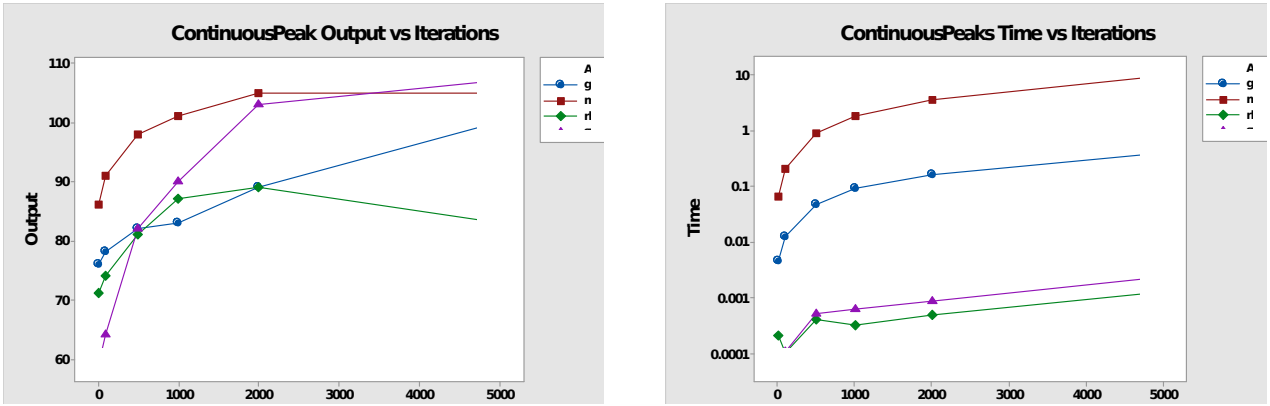

Figure 8. (Left) Continuous peak output vs iteration plot. (Right) Continuous peak time vs iteration plot.

3) Traveling Salesman Problem

Traveling salesman problem (TSP) is to solve the following problem: given a list of cities and the distances between each other, find out the shortest possible route that visits each city exactly once and returns to the origin city. The N value for TSP is defined at 40 in ABAGAIL.

From Fig. 9 (left), GA is much better than the other algorithms, although it even shows a slight decreasing trend with increasing iterations. From Fig. 9 (right), the time cost still shows the similar

pattern, MIMIC GA are much higher than SA and RHC.

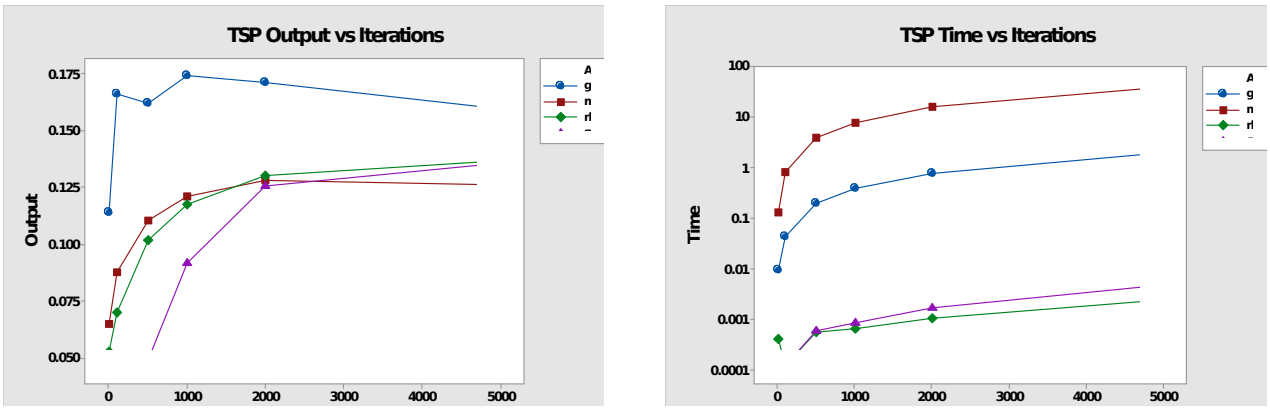

Figure 9. (Left) TSP output vs iteration plot. (Right) TSP time vs iteration plot.

From Fig. 10 (left), the fitness of GA is higher with more population combinations, and as shown on Fig. 10(right), the time cost increases linearly with population parameters: (10, 5, 2), (20, 10, 5), (50, 25, 10), (100, 50, 10), (200, 100, 20), and (500, 350, 50). The best fitness value is at 500_350_50, but the time cost is also the highest compared with other combination.

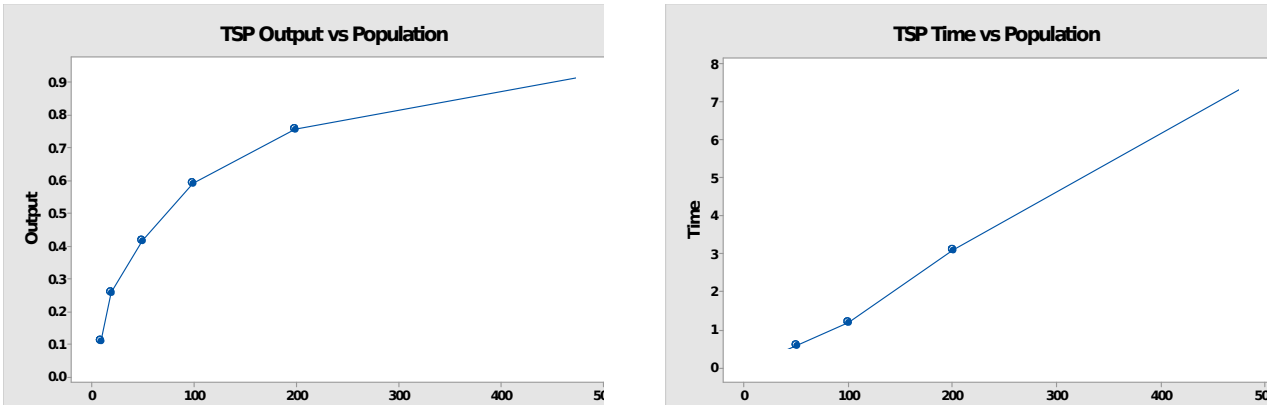

Figure 10. (Left) TSP output vs population plot. (Right) TSP time vs population plot.

From Table 2, we discuss when to use each algorithm and what is the time cost. From what we learning in this study, RHC and SA are good for simple structure problems to just find the global optima within a short time, whereas GA and MIMIC are suited on more complex structure problems that also allow higher time cost, and MIMIC can generate a dependency tree that are good for preserving the structure of the problem.

Table 2. Performance comparison between algorithms

Algorithms	When to use	Time cost
RHC	Simple structure problems	Low
SA	Simple structure problems	Low
GA	Complex structure problems	Med
MIMIC	Complex structured problems and requirement on saving the structure with dependency tree	High

V. Conclusions

The weights of the neural network for Breast Cancer dataset are optimized with three different algorithms, RHC, SA, and GA. The accuracy and time cost has been compared and analyzed. The RHC algorithm is found to be the best on the BreastCancer dataset to calculate the weights for a neural network, since it has a slightly better test score compared with the results in assignment#1. Three optimization problems, such as Knapsack, Continuous Peaks, and Traveling Salesman problems, are analyzed with four different algorithms, RHC, SA, GA, and MIMIC. MIMIC score the best on Knapsack problem, SA is best on Continuous Peaks problem, and GA is found to be the best for Traveling Salesman problem. The best performance of MIMIC on Knapsack is found with input sample number equals 200 and Tokeep at 20. The accuracy and time cost are analyzed with varying parameters for the algorithms. RHC and SA are better suited to work on problems with less requirement of the structure complexity with faster computation speed, whereas, GA and MIMIC cost more time, but they are good for complex structure problems.

VI. References

[1] Pushkar. Abagail, 2017. https://github.com/pushkar/ABAGAIL

[2] C.L. Isbell, Jr, Randomized Local Search as Successive Estimation of Probability Densities. (https://www.cc.gatech.edu/~isbell/tutorials/mimic-tutorial.pdf)

[3] This database is also available through the UW CS ftp server: ftp ftp.cs.wisc.edu cd math-prog/cpo-dataset/machine-learn/WDBC/

Also can be found on UCI Machine Learning Repository: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29