AI-AB: COURSEWORK 1

CANDIADATE NUMBER: 184521

Contents

Introduction

Hill Climbers and Genetic Algorithms are two algorithms that can be used for the optimisation of a problem in a similar way just with different approaches. Genetic algorithms are a metaheuristic algorithm that is inspired by natural selection and were first in development during the 1980's to try to harness the algorithms abilities for optimisation.(1) Hill climbing on the other hand is a heuristic search algorithm that also has uses in mathematical optimisation problems for the field of Artificial Intelligence.(2) Here we explore how both algorithms optimisation capabilities can be used to solve a simple bag problem, where when given a bag with a specific maximum weight and items that have both a weight and fitness will find the selection of items with the highest fitness that don't breach the max weight of the bag. The idea behind exploring the capabilities of each of the algorithms is to best understand which algorithm is best at optimisation through changing variables such as size of the bag, number of items, number of iterations and so on. I hypothesise that the genetic algorithm will perform better than the hill climbing algorithm with the given tasks and that higher mutation rate will give better results.

Methodology

To begin both algorithms need the same parameters to be changed to ensure we can appropriately compare the two algorithms and work out which is better at optimisation. The parameters that we have available that are both shared between the algorithms are the mutation rate, the number of iterations and population size as well as the number of items that can be flipped in them. Of these the mutation rate is probably the most important parameter to be changed as by changing it we can change how we quickly an algorithm reaches the highest value available. This will give us the ability to closely inspect the graph and more easily see the differences in speed when this parameter is changed and how it affects the overall outcome. The second parameter that can be changed is the number of iterations, this however has very little effect when it comes to the efficiency of the algorithms. Therefore, this value shall only be changed to ensure that the algorithms reaches the max possible fitness and will be the main way to measure how good and efficient an algorithm is at optimisation. This is as the algorithm that reaches the maximum value the fastest is the more efficient and therefore the better optimisation algorithm. The third value that can be changed is the population size, this value does affect the algorithm and has effects on its efficiency to. This is as in theory increasing the population should decrease the time taken to reach the maximum fitness it is just a matter of how much it effects the efficiency of doing so. The ability to change the size could also be an important variable to change as it will allow us to see if with larger problems does the speed at which the highest value is reach ed change if the value is higher or lower. This is as with the ability to have more or different number 'items' available to a 'bag' the higher or lower the max possible value will be. For simplicity though this shall be kept at the max being 20 possible 'items' per 'bag' as this will allow for a larger gap, if there is any, between the algorithms.

With these four values and their effects now clearly stated we can now progress on to the methodology of measuring the two different algorithms. The main method will be changing between a low, medium and high set of values for the mutation rate and population size parameters and then seeing at which iteration the highest value is hit. For population size these values will be 10 for low, 50 for medium and 100 for high and with mutation rate we will have values 30%, 60% and 90% for high, medium and low respectively. These can then be compared to work out which algorithm is truly the most efficient by looking at how soon the best fitness value is reached as well as how the other members of the population perform. For this to work the other parameters must be kept the same whilst only changing either the mutation rate or size of the population between the previously given high, medium and low values. Whilst this occurs the other value will be kept at its lowest setting to limit its effect on the data e.g. whilst changing population size the mutation rate will be set to 30%. This leaves the size of the ‘bags’ being kept at 20 and the iteration number to be left at 50 to allow for better comparison. This will result in the algorithms outputting iteration number vs fitness graphs that can be compared with their being four distinct categories for these graphs. These will be Genetic Algorithm’s population size and mutation rate as well as Hill Climber’s population size and mutation rate and will be run 20 times to get average values. I will also look at crossover rate, which is unique to genetic algorithms, to see if it affects the Genetic Algorithm or whether the mutation rate is more important. To do this mutation rate will be kept at 0 for the duration of the crossover experiments so it doesn’t interfere, and the crossover rate will be kept at 0 for the mutation rate experiments as well.

Pseudo Code

Population of Hill Climbers

```
for i in range(pop):
  i<-randomItem
  best<- current best
  while z not = number_of_iterations
    if i in bag
      new loop
    else
      if item+capacity<size
        bag<-bag+item
       else
        if bag+lowest_item<size
        else
          calc_fitness()
          eval_fitness()
            if new>old
              return new
              start new loop
            else
              return old
              new loop
    before new loop
      plot(best and z)
      z<-z+1
displayGraph()
```

Genetic Algorithm

```
For i in range(population_size)
  Population<-Population+RandomGenotype
For i in range(iterations)
  A<-randomInt
  B<-randomInt
  If(Population[a]>Population[b])
    Best<-A
    Worst<-B
  Else If(Population[a]<Population[b])
    Best<-B
    Worst<-A
  Rate<-randInt
  If Rate<=crossoverRate
    gene<-randInt
    if((Best[gene]=1) and (Worst[gene]=0)
      Worst[gene]=<-1
      calcWeight[Worst]
      calcFitness(worst)
    if((Best[gene]=1) and (Worst[gene]=0)
      Worst[gene]<-0
      calcWeight[Worst]
      calcFitness(worst)
  Rate<-randInt
  If Rate<=muatationRate
    Gene<-randInt
    For i in length(populationSize)
      If(worst[gene]=1)
        Worst[gene]=0
      If(worst[gene]=0)
        Worst[gene]=1
      calcFitness(worst)
  New_population<-Population+Best+Worst
```

Results

Hill Climber Population Size Averages

	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
High	27.9	32.7	15.7	94.45
Med	16.8	31.85	16.75	92.1
Low	18.85	28.8	17.1	90

Hill Climber Algorithm Mutation Rate Averages

	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
High	19.6	29.9	18.1	81.75
Med	19.5	29.75	17.6	90
Low	21.6	30.65	17.7	90

Genetic Algorithm Population Size Averages

	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
High	1	33	0.75	18.25
Med	10.5	30.3	1	19.8
Low	21.5	27.45	6.89	40

Genetic Algorithm Mutation Rate Averages

	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
High	11.65	29.8	6.85	38
Med	12.15	29.85	7.45	39.25
Low	12.15	30.45	6.35	48.5

Genetic Algorithm Crossover Rate Averages

	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
High	15.1	27.8	2.81	23.25
Med	15.1	28.45	3.58	24
Low	13.05	29.3	2.87	23

From the above data I have noticed the following trends, for the Hill climber, population size has more of an effect whereas the mutation rate has slight opposite or no effect overall with less individuals getting over 20 fitness when the mutation is applied. By opposite effect I mean that the values decrease with the increase of the rate and effect meaning how much it changes the values of the individuals. Population size obtains higher fitness's overall and manages to get the maximum value more often than with mutation though it takes longer on average to reach this value. Mutation raises the lowest fitness but not by much and those individuals that don't make it over 20 fitness get very close to making it past this threshold. Changing the population rate also gets more individuals over the 20-fitness threshold but unlike mutation unless there is a low population size those that don't make it over are usually further from making it.

For the Genetic Algorithm the population size gets the higher value results more often and quicker than the hillclimber when set to a lower level though it can be seen to be slower than the hi climber algorithm. Mutation has a very slight opposite effect or at times no effect overall with the largest effect being at a medium value though less individuals get over the 20-fitness threshold. Crossover rate also has a very little opposite and or no effect when it comes to the values of the individuals produced by the genetic algorithm. Though a lower or medium cross over rate causes the best and biggest change. Population size has less above 20 fitness where mutation rate has more above the threshold but in an opposite effect.

Discussion

In the Hill climber algorithm, it is safe to say that the population size has the largest effect on the success of the algorithm, this can also be said for the genetic algorithm. This is as in both cases the larger the population size the more individuals there are that can reach higher numbers, therefore giving the appearance and view that it acts better. This can be seen in the genetic algorithms population size test were the average iteration number where the top number reached was one e.g.

the initiation of the individual. This is as by probability there are only so many ways that an individual can initialise so by a population size of 100 it is confirmed that at least one will initialise with the maximum possible value. That is why even though the population size has the largest effect it is only due to probability.

In both the Hill Climber and the Genetic Algorithm, the Mutation Rate has a very slight opposite effect or no effect at all and less individuals get over 20 fitness. This again is probably due to probability as with a higher mutation rate the more chance there is for an item to be removed. For the Hill Climber this results in the new selection not overwriting the old as it will result in a worse fitness unless a better item fills the void. This can explain how the lowest fitness increases slightly with a higher mutation rate as there is more of a chance that a slightly better item can fill the void. Though this is the case these items are obviously not enough to get to the max possible value as from the averages the higher mutation rate causes the hillclimbers to become very close and have the similar fitness. This is as the top fitness's decrease and the lower ones increase leading to greater similarity between individuals. In the genetic algorithm there is a similar trend though the gap between the fitness's of individuals seems larger than the hill climbers though this is due to hill climbers not taking the new fitness unless its better.

The last trend that can be discussed is the crossover rate for the Genetic Algorithm, as this gets higher it as a very slight opposite effect or no effect just like the genetic algorithms mutation rate. The only effect that can be seen is at the medium cross over rate, being 60%, where the average fitness of the individual increases slightly along with the lowest fitness and the percentage above the fitness of 20. This makes around 60% crossover the best option for crossover in genetic algorithms. All if the above observations however are limited as they were all taken within 50 iterations of each of the algorithms and don't consider the possibility of a combination of settings. For example, a combination of 50% mutation and low crossover rates with a high population size in theory could return the highest possible value in less than 15 iterations or maybe even fewer.

Conclusion

In conclusion, the best use of algorithms for this problem will probably be a Genetic algorithm with a medium mutation and low crossover rate but with a high population size and a Hill Climber algorithm with a low mutation rate but large population size. These algorithms configured this way will give the best chance at returning as many high valued individuals as possible. However, this problem only requires that the maximum possible value is found and returned as quickly as possible, to this end the Genetic algorithm with a high population size is the best bet for this problem. This is as it will return at least one maximum value individual almost immediately which is all the problem requires as its simply running as a heuristic algorithm. This however goes against my original hypothesise where I stated that a genetic algorithm with just a high mutation rate would give the better results. However, if there were more components and the individuals were of a larger size this would change as a hill climber may then become better suited. This would then require a hillclimber with a low mutation rate and large population size as there is seen to be more of a general increase over time in individual's fitness's here and would also execute much faster. This is as a genetic algorithm would need a much larger population size to achieve a similar answer which would take more time to initialise and more iterations to run to get the same answer as a hillclimber. For this problem though the genetic algorithm is best.

Appendix

Bibliography

1. Mitchell, Melanie (1996). *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press.

https://www.edureka.co/blog/hill-climbing-algorithm-ai/

Genetic Algorithm; Population Size

High

No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	1	31	0	20%
2	1	33	0	24%
3	1	31	0	30%
4	1	32	2	14%
5	1	33	1	12%
6	1	33	1	24%
7	1	33	1	19%
8	1	33	0	16%
9	1	33	0	17%
10	1	32	1	20%
11	1	33	0	15%
12	1	33	1	16%
13	1	33	2	13%
14	1	33	1	14%
15	1	33	1	16%
16	1	33	0	15%
17	1	33	0	17%
18	1	33	0	19%
19	1	33	2	17%
20	1	33	2	17%

Medium

No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	3	33	1	10%
2	1	33	0	14%
3	1	34	3	16%
4	1	39	1	22%
5	25	29	1	24%
6	23	26	0	28%
7	1	36	0	16%
8	2	33	0	16%
9	1	34	0	16%
10	1	31	2	18%
11	19	36	2	24%
12	1	28	0	10%
13	40	33	0	20%
14	50	33	1	20%
15	28	33	3	22%
16	1	25	0	6%
17	9	30	0	18%
18	1	28	3	16%
19	1	33	3	22%
20	1	32	0	18%

Low

No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	4	13	9	20%
2	12	33	11	40%
3	44	33	3	50%
4	50	22	3	40%
5	23	25	0	10%
6	40	27	8	50%
7	50	37	13	80%
8	16	27	4	30%
9	23	24	1	30%
10	12	23	3	30%
11	38	31	3	30%
12	1	17	7	0%
13	1	28	5	20%
14	23	31	8	60%
15	1	23	5	20%
16	11	33	13	70%
17	29	27	19	90%
18	1	31	6	30%
19	50	34	4	40%
20	1	30	6	20%

Averages

	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
High	1	33	0.75	18.25%
Med	10.5	30.3	1	19.8%
Low	21.5	27.45	6.89	40%

Hill Climber; Population Size

High-100

No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	44	33	17	96%
2	15	33	15	95%
3	9	31	18	96%
4	46	32	16	92%
5	4	33	15	93%
6	22	33	13	94%
7	33	33	17	95%
8	9	33	13	93%
9	15	33	15	95%
10	37	33	16	94%
11	25	33	16	94%
12	33	33	17	94%
13	35	32	16	96%
14	40	33	16	95%
15	25	33	16	94%
16	37	33	14	94%
17	48	33	17	94%
18	30	33	15	94%
19	28	33	16	94%
20	23	31	16	97%

Medium-50

No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	25	30	16	94%
2	46	33	18	96%
3	17	32	17	92%
4	32	33	17	90%
5	3	29	17	92%
6	4	33	17	90%
7	16	32	18	94%
8	32	30	17	94%
9	8	32	15	92%
10	10	32	17	90%
11	8	30	15	88%
12	12	32	18	94%
13	13	33	16	92%
14	23	32	17	90%
15	17	32	19	96%
16	6	32	17	92%
17	14	33	15	88%
18	17	32	18	94%
19	26	33	16	92%
20	7	32	15	92%

Low-10

No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	9	29	19	80%
2	13	32	18	70%
3	32	29	19	90%
4	29	32	16	70%
5	15	32	18	90%
6	7	27	20	100%
7	18	32	16	80%
8	26	31	21	100%
9	37	26	17	80%
10	14	26	21	100%
11	38	29	19	90%
12	34	30	17	90%
13	2	26	17	70%
14	4	28	21	100%
15	18	27	20	100%
16	13	30	19	90%
17	15	27	20	100%
18	18	24	20	100%
19	6	29	22	100%
20	29	30	20	100%

Averages

	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
High	27.9	32.7	15.7	94.45
Med	16.8	31.85	16.75	92.1
Low	18.85	28.8	17.1	90

Genetic Algorithm; Crossover Rate

High

No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	13	27	3	40%
2	48	28	3	30%
3	14	30	4	30%
4	8	30	1	15%
5	31	35	5	30%
6	18	24	3	15%
7	1	25	0	20%
8	5	27	3	15%
9	13	30	0	25%
10	14	27	3	25%
11	23	28	2	35%
12	14	29	0	20%
13	12	22	1	5%
14	31	21	3	5%
15	16	33	0	40%
16	23	22	3	10%
17	5	36	3	40%
18	11	26	4	20%
19	1	29	3	30%
20	1	27	1	15%

Medium

No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	33	40	4	60%
2	8	26	0	20%
3	1	28	3	25%
4	23	34	0	35%
5	37	28	0	15%
6	32	22	0	10%
7	12	29	0	35%
8	36	42	4	35%
9	22	25	9	20%
10	31	24	0	5%
11	1	22	3	10%
12	1	30	3	20%
13	14	31	0	25%
14	1	26	2	15%
15	1	23	3	25%
16	36	33	3	35%
17	1	22	0	10%
18	1	26	3	25%
19	1	28	3	30%
20	10	30	3	25%

Low

No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	17	26	3	15%
2	1	31	3	25%
3	3	33	7	25%
4	16	36	3	20%
5	20	29	3	40%
6	1	20	0	5%
7	18	35	3	45%
8	1	31	1	35%
9	33	33	3	15%
10	1	32	0	15%
11	35	24	0	20%
12	1	30	0	15%
13	33	28	3	15%
14	34	31	0	25%
15	1	29	3	30%
16	1	24	4	10%
17	13	28	3	25%
18	1	28	1	25%
19	1	23	2	15%
20	30	35	1	40%

Averages

	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
High	15.1	27.8	2.81	23.25
Med	15.1	28.45	3.58	24
Low	13.05	29.3	2.87	23

Hill Climber; Mutation rate

High-90%

No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	41	32	17	80%
2	34	28	20	100%
3	18	30	19	95%
4	26	29	17	90%
5	8	28	19	95%
6	13	32	19	90%
7	6	31	17	95%
8	13	33	18	95%
9	32	28	17	95%
10	30	30	18	95%
11	3	28	18	90%
12	13	31	18	85%
13	27	33	16	80%
14	32	30	18	95%
15	4	30	17	70%
16	31	32	18	80%
17	17	33	19	85%
18	7	31	19	95%
19	15	30	20	100%
20	22	29	18	95%

Medium-60%

No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	39	30	16	85%
2	8	28	18	90%
3	16	33	20	100%
4	37	33	17	90%
5	35	32	16	90%
6	16	28	18	90%
7	3	28	20	100%
8	14	32	14	80%
9	4	28	18	90%
10	26	28	21	100%
11	7	30	18	90%
12	48	31	18	95%
13	7	29	16	85%
14	47	33	16	80%
15	20	32	18	95%
16	17	32	18	90%
17	6	30	18	85%
18	3	29	18	95%
19	29	31	17	85%
20	8	18	17	85%

Low-30%

No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	3	32	17	80%
2	23	32	19	90%
3	41	30	16	95%
4	40	33	16	85%
5	47	29	17	90%
6	39	32	18	95%
7	32	33	21	100%
8	9	29	18	90%
9	23	32	16	85%
10	3	30	17	95%
11	3	26	17	85%
12	8	29	17	80%
13	42	29	18	90%
14	3	32	18	85%
15	22	30	17	85%
16	28	32	18	90%
17	8	28	18	95%
18	15	32	20	100%
19	35	30	18	95%
20	8	33	18	90%

Averages

	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
High	19.6	29.9	18.1	81.75
Med	19.5	29.75	17.6	90
Low	21.6	30.65	17.7	90

Genetic Algorithm; Mutation Rate

High-90%

No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	12	32	8	35%
2	1	33	9	30%
3	1	33	10	35%
4	1	30	6	35%
5	1	29	8	35%
6	26	26	9	35%
7	1	27	10	45%
8	47	30	4	55%
9	49	26	4	40%
10	48	31	6	45%
11	1	32	5	50%
12	1	30	6	60%
13	1	28	12	50%
14	1	30	3	35%
15	10	32	12	45%
16	28	33	4	25%
17	1	29	4	30%
18	1	20	2	25%
19	1	33	10	45%
20	1	32	5	40%

Medium-60%

No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	16	21	13	35%
2	31	30	9	45%
3	1	29	9	45%
4	18	26	7	35%
5	1	33	6	40%
6	8	26	9	30%
7	1	32	3	40%
8	24	33	6	70%
9	1	33	5	40%
10	1	30	6	40%
11	40	33	10	40%
12	1	28	5	50%
13	45	31	9	20%
14	1	31	10	35%
15	1	30	6	35%
16	49	30	1	25%
17	1	28	9	40%
18	1	30	8	40%
19	1	32	2	50%
20	1	31	7	30%

Low-30%

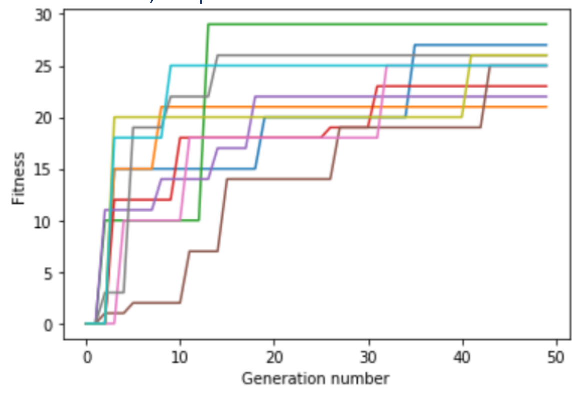
No.	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
1	1	29	2	75%
2	1	30	5	65%
3	1	28	4	55%
4	43	33	3	40%
5	29	28	7	55%
6	28	28	7	50%
7	26	31	10	50%
8	1	32	7	35%
9	1	33	4	30%
10	23	32	8	55%
11	36	33	10	45%
12	9	29	9	35%
13	1	29	5	55%
14	1	32	7	45%
15	1	26	3	50%
16	1	31	10	50%
17	13	33	3	40%
18	4	28	8	45%
19	22	33	7	55%
20	1	31	8	40%

Averages

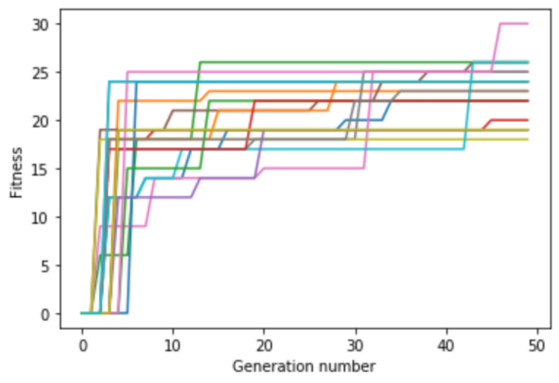
	Iteration No. top fitness	Highest Fitness	Lowest Fitness	% above 20
High	11.65	29.8	6.85	38
Med	12.15	29.85	7.45	39.25
Low	12.15	30.45	6.35	48.5

Example Graphs

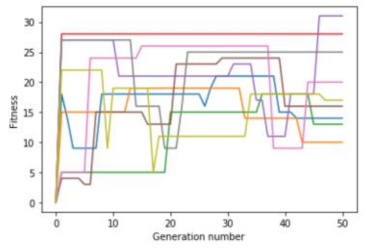
Hill Climber; Population Size



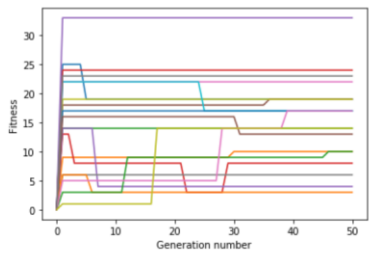
Hill Climber; Mutation rate



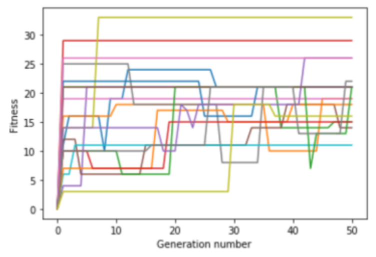
Genetic Algorithm; Population Size



Genetic Algorithm; Crossover Rate



Genetic Algorithm; Mutation Rate


Fitness
Generation number