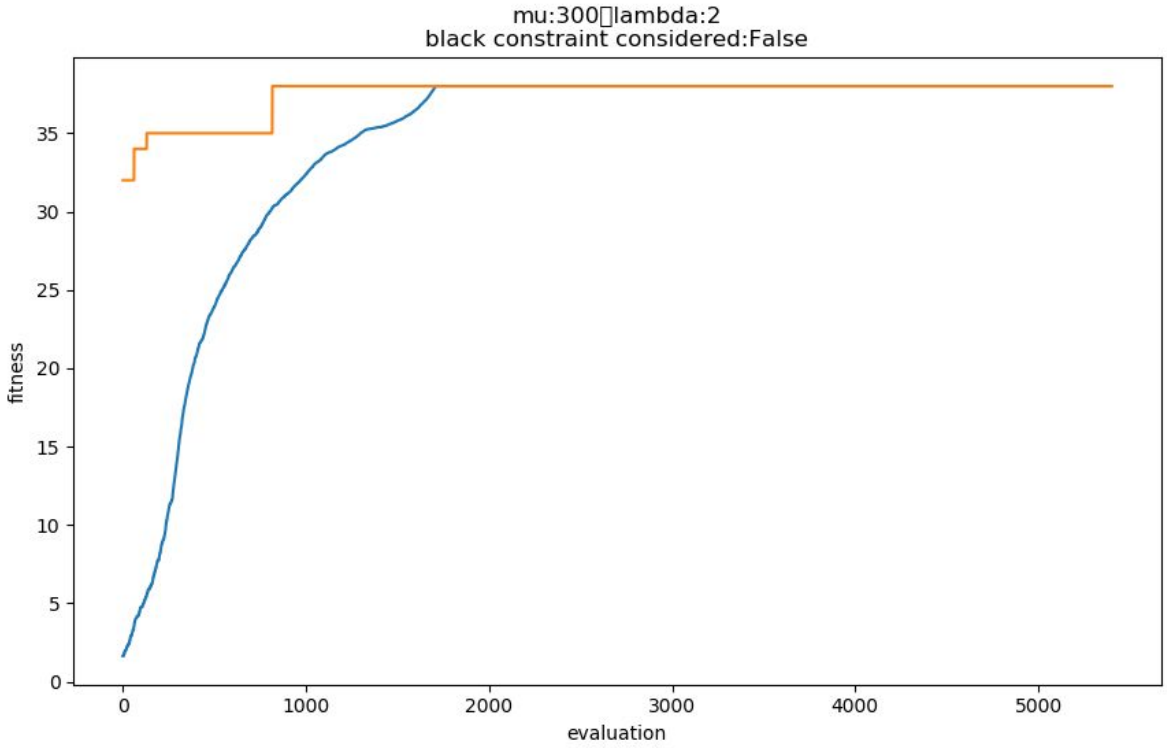
In this assignment, I have implemented the (mu+lambda) evolutionary algorithm. In which mu random solutions will be generated at first, and after that the fittest ones are selected from the pool to mate and give birth to the next generation.

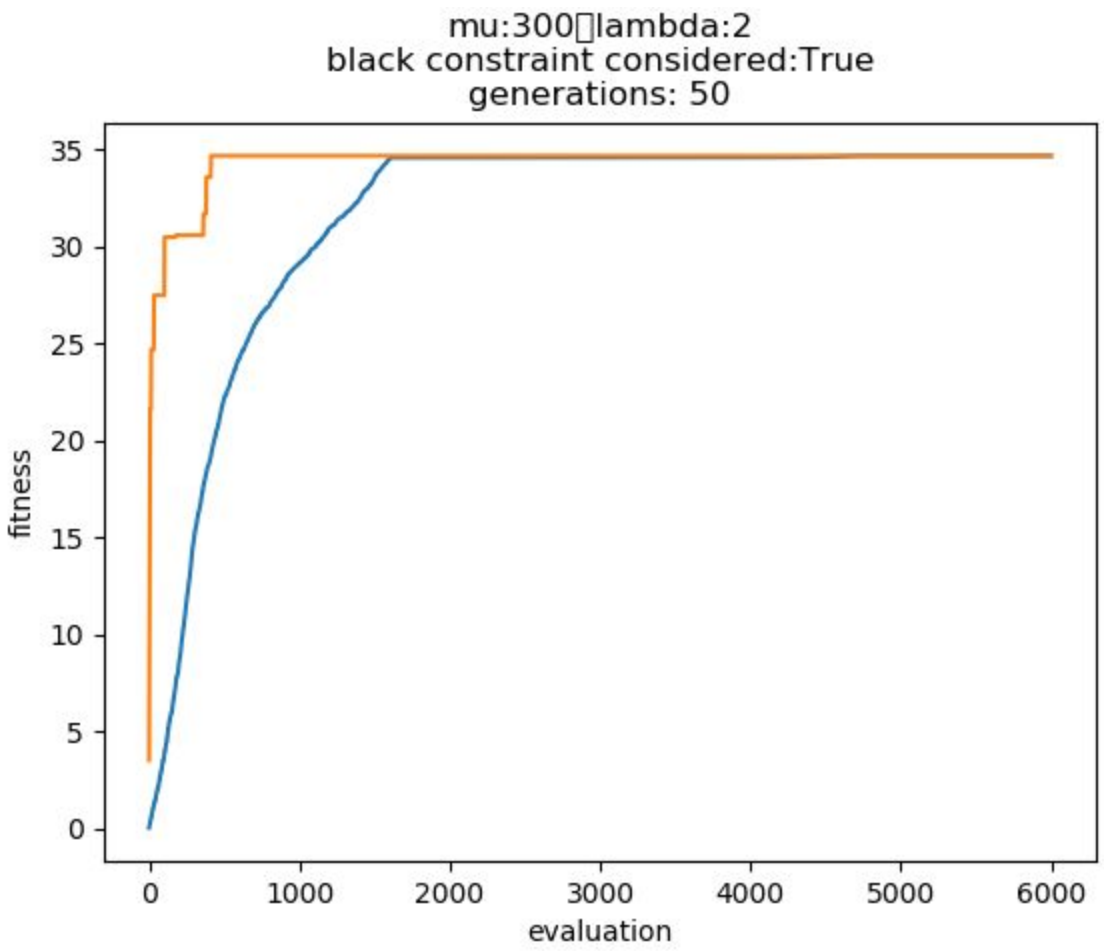
To explain in more detail, the size of my pool is mu, and lambda new individuals are born at each generation and then the less fit ones are killed so that the size of the pool remains the same. For parent selection, two parents will be selected each time, uniformly at random. Then the top horizontal half of the solution board of the first parent is concatenated to the lower horizontal half of the second parent, in order to make the baby.

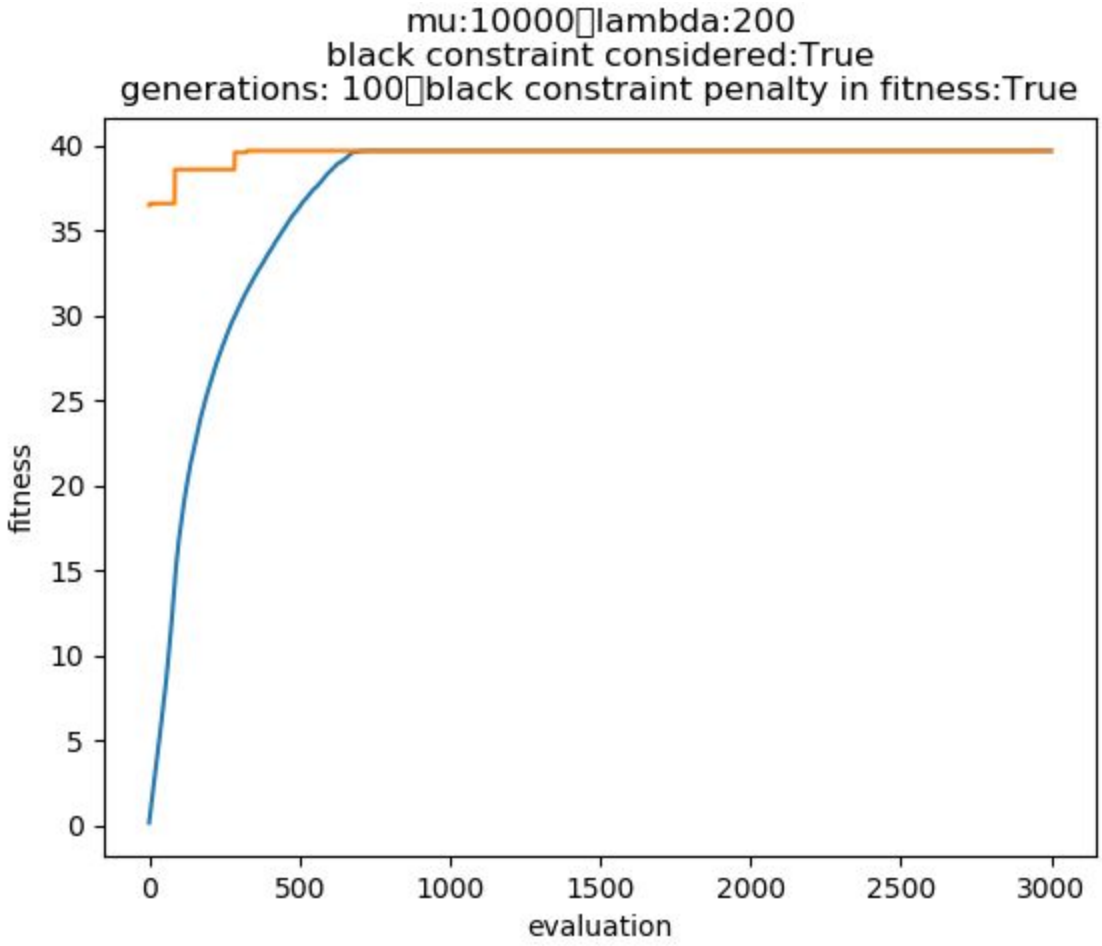
I have experimented with lots of different mu and lambda, and noticed that in order to keep the mating pool diverse enough, the surviving population should be large enough to include some less fit individuals. Otherwise, it would consist of the same looking individuals that will not produce better offspring, as if they are exactly similar, their child would be exactly the same as them.

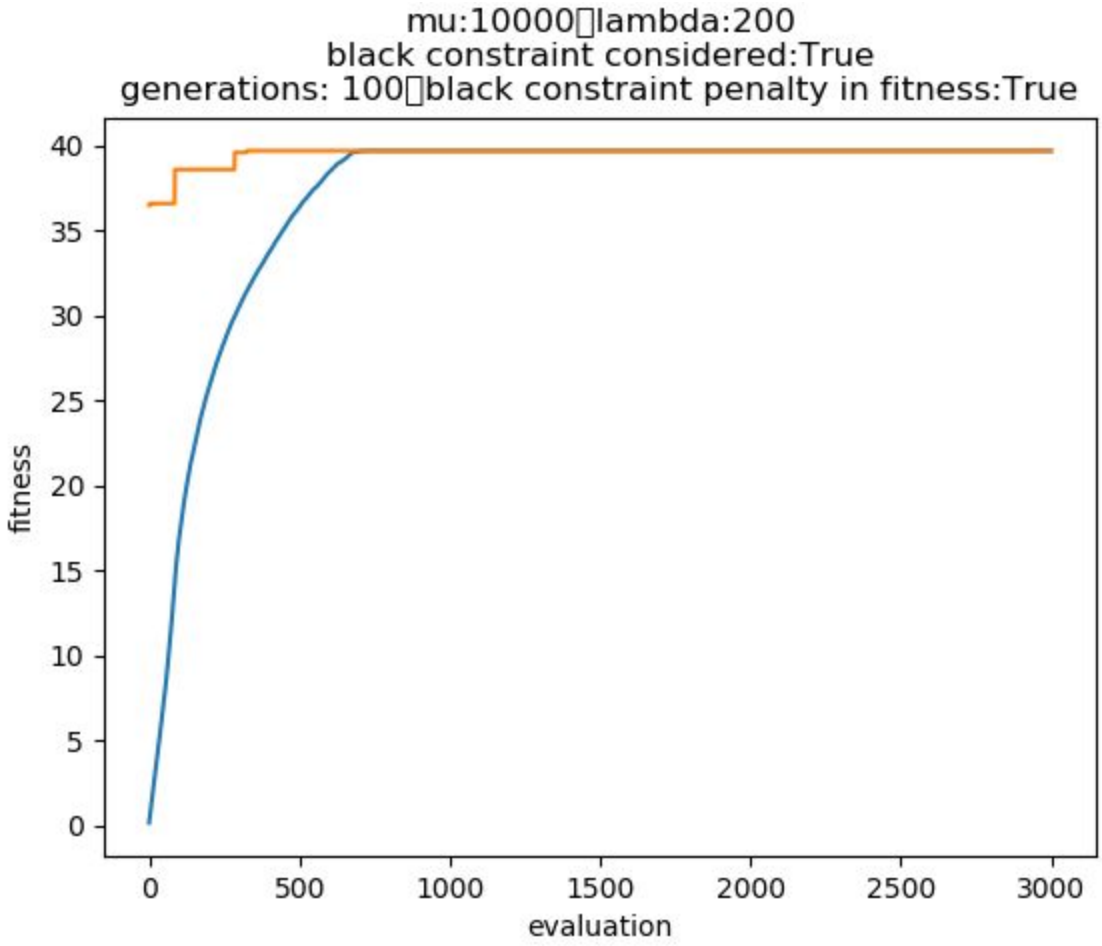
- Something interesting that I have found was that rather than giving 0 fitness value to the solutions that violate the black adjacent bulb constraint, by just penalizing them in proportion with the constraints they have violated, we would be able to get much better results. For instance for problem a2, in a situation where I could not get any valid results after 100000 runs, I got a solution that only violated one of the black block constraints. I penalize the solutions that violate the black constraints, by 0.1 on each constraint violated. This is done by setting the semi_black_constraint to True, in the configuration.
- Below are the average fitness(blue) and best fitness(orange) plots. Average fitness has less variance due to the variance law.

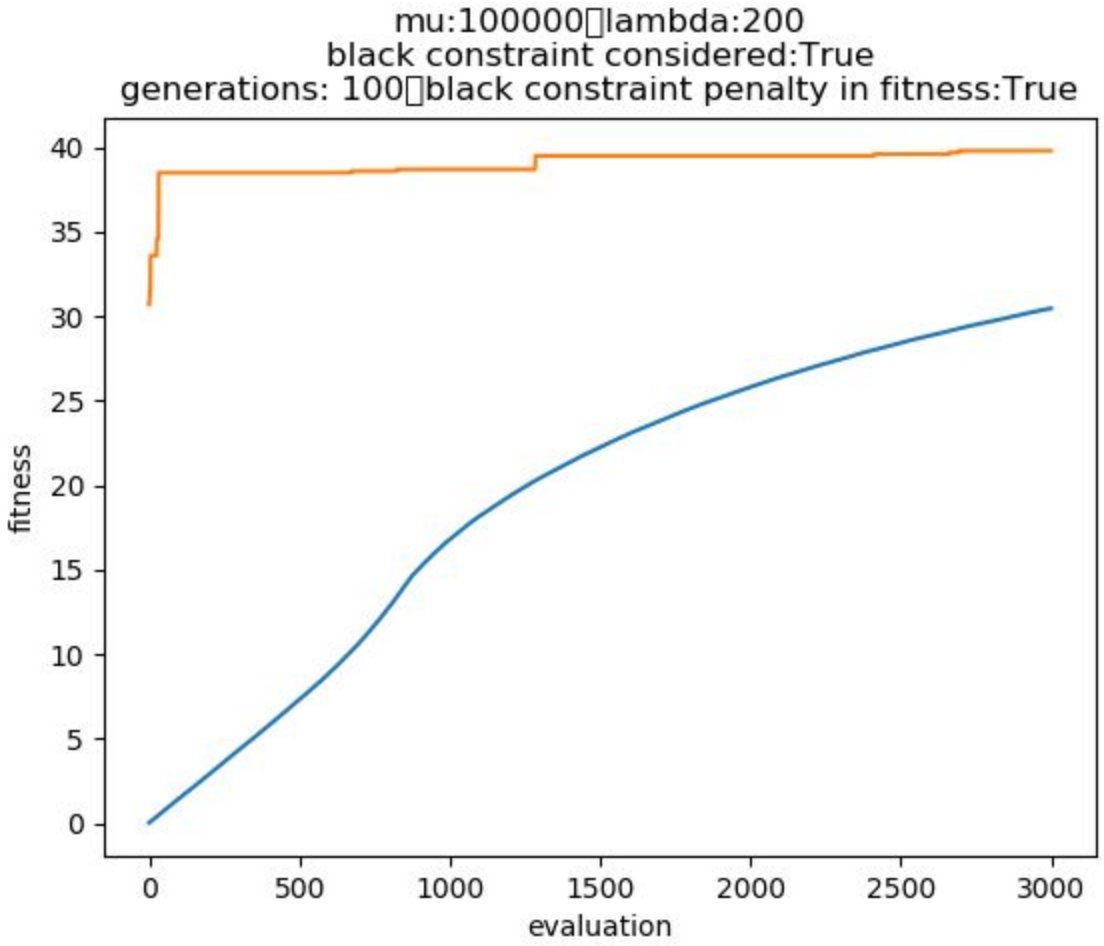
  For problem a2:


mu:300□lambda:2
black constraint considered:False
fitness
evaluation
0
5
10
15
20
25
30
35
0
1000
2000
3000
4000
5000

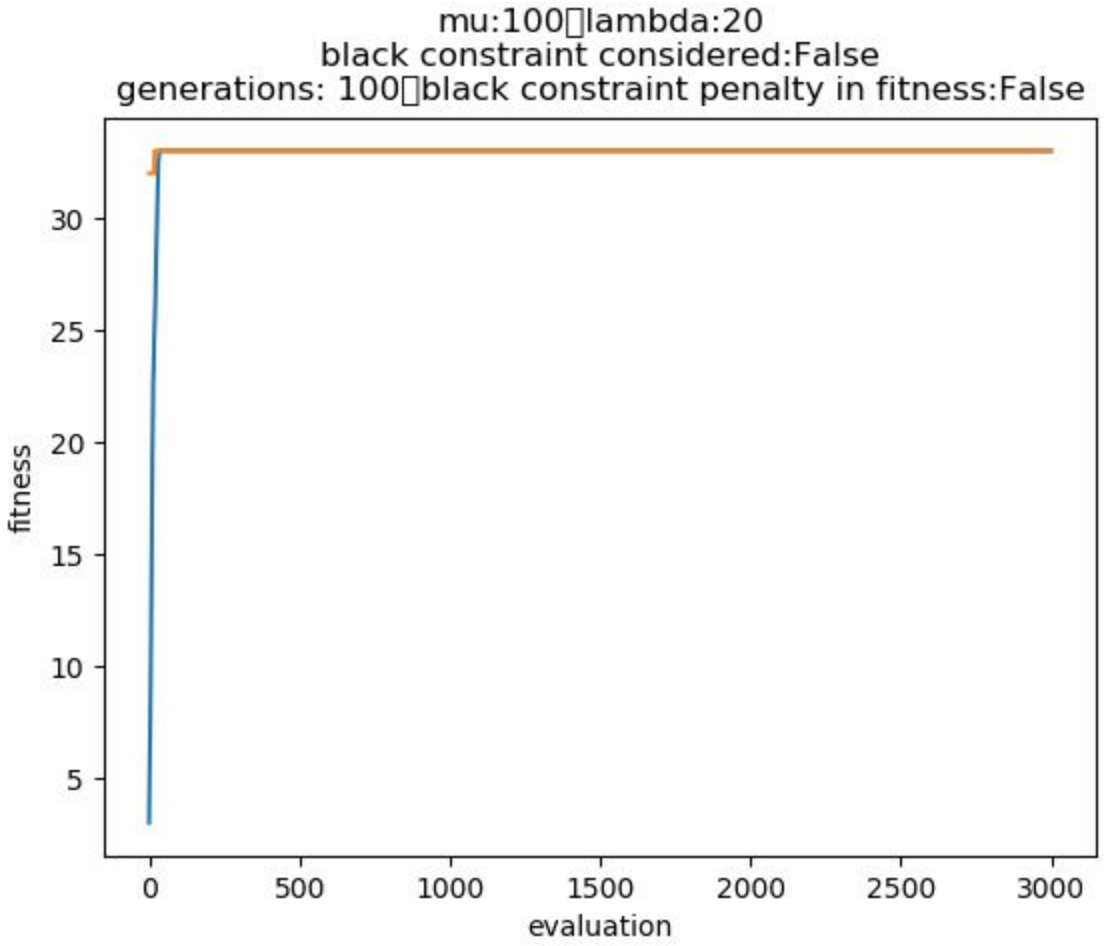

mu:300□lambda:2
black constraint considered:True
generations: 50
fitness
evaluation
0
5
10
15
20
25
30
35
0
1000
2000
3000
4000
5000
6000


mu:10000□lambda:200
black constraint considered:True
generations: 100□black constraint penalty in fitness:True
fitness
evaluation
0
5
10
15
20
25
30
35
40
0
500
1000
1500
2000
2500
3000

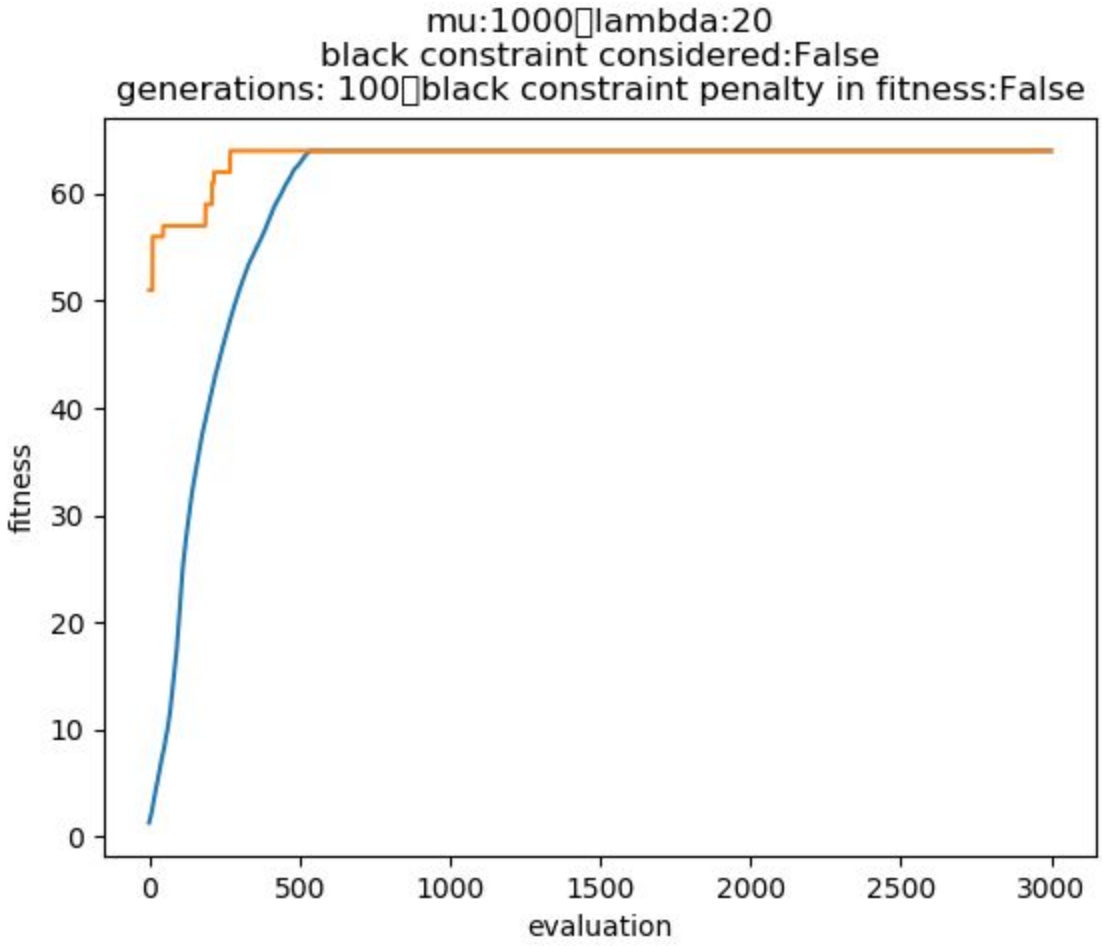

mu:10000□lambda:200
black constraint considered:True
generations: 100□black constraint penalty in fitness:True
fitness
evaluation
0
5
10
15
20
25
30
35
40
0
500
1000
1500
2000
2500
3000


mu:100000□lambda:200
black constraint considered:True
generations: 100□black constraint penalty in fitness:True
fitness
evaluation
0
5
10
15
20
25
30
35
40
0
500
1000
1500
2000
2500
3000

Problem a1:


mu:100□lambda:20
black constraint considered:False
generations: 100□black constraint penalty in fitness:False
fitness
evaluation
30
25
20
15
10
5
0
500
1000
1500
2000
2500
3000

Problem a3:


mu:1000□lambda:20
black constraint considered:False
generations: 100□black constraint penalty in fitness:False
fitness
evaluation
0
10
20
30
40
50
60
0
500
1000
1500
2000
2500
3000