Solving The Travelling Salesman Problem by Genetic Algorithm

Background

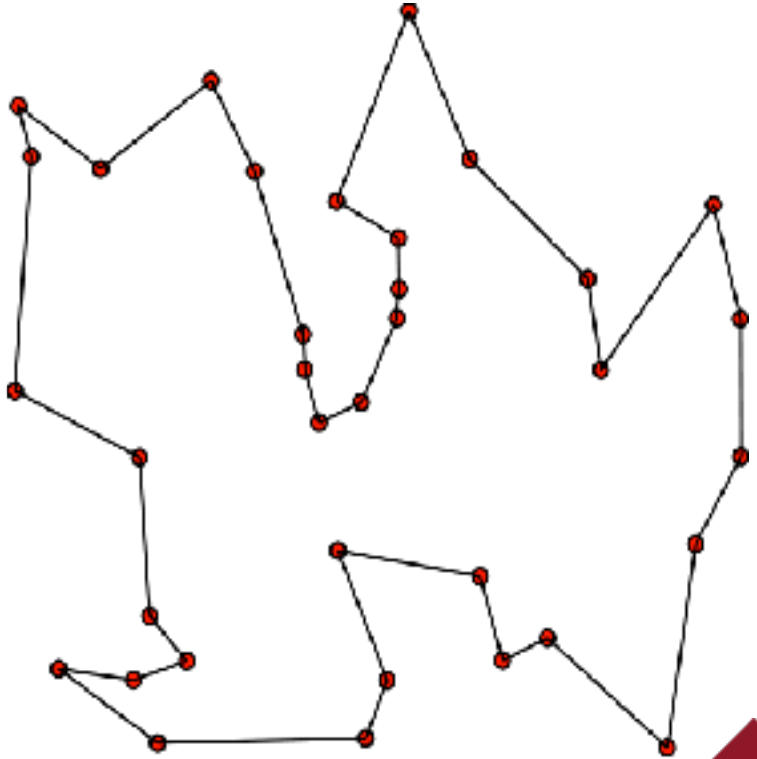
The travelling salesman problem (TSP)

"*Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?*"

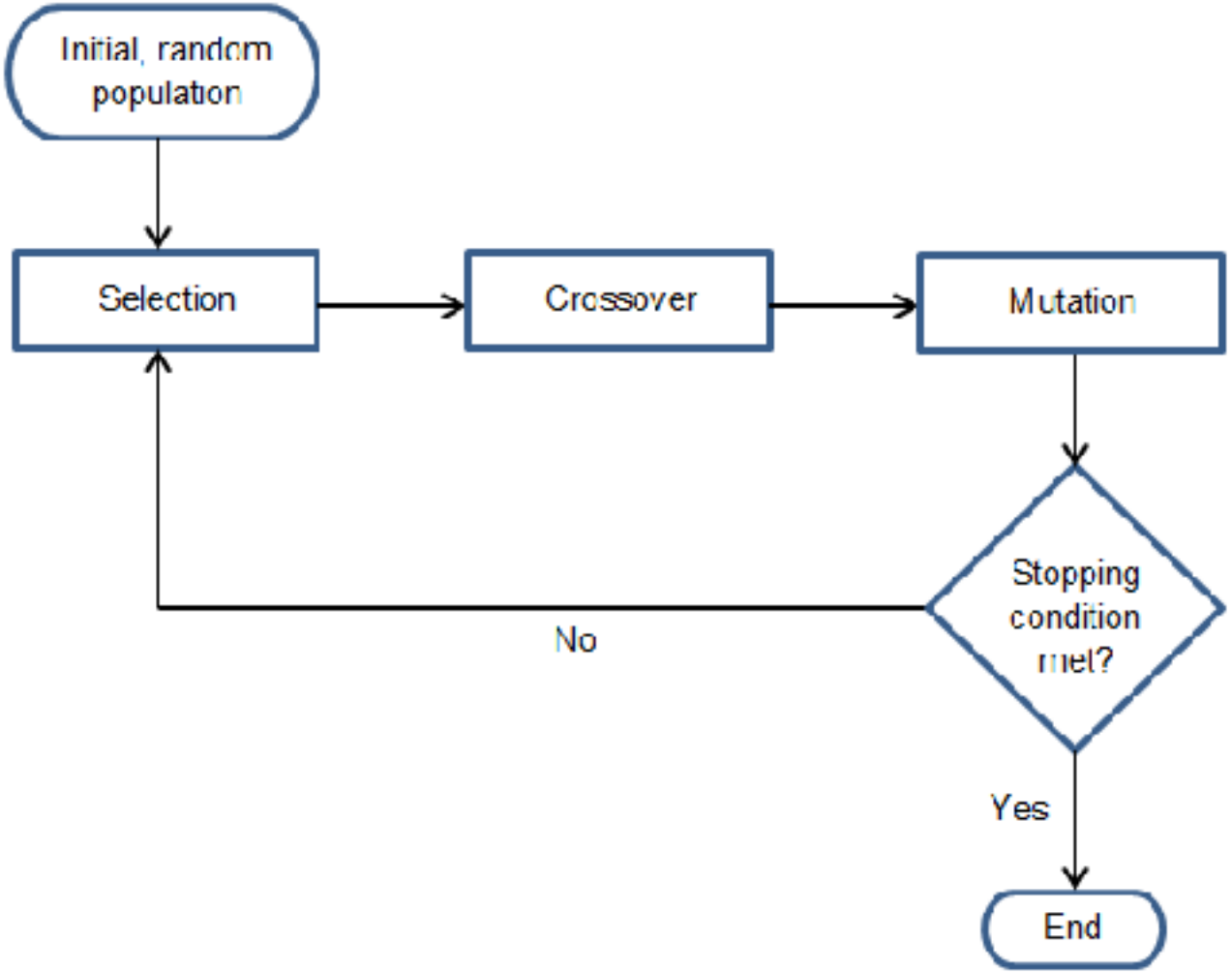
Background

```
NAME: berlin52
TYPE: TSP
COMMENT: 52 locations in Berlin (Groetschel)
DIMENSION: 52
EDGE_WEIGHT_TYPE: EUC_2D
NODE_COORD_SECTION
1 565.0 575.0
2 25.0 185.0
3 345.0 750.0
4 945.0 685.0
5 845.0 655.0
6 880.0 660.0
7 25.0 230.0
8 525.0 1000.0
9 580.0 1175.0
10 650.0 1130.0
11 1605.0 620.0
12 1220.0 580.0
13 1465.0 200.0
14 1530.0 5.0
15 845.0 680.0
16 725.0 370.0
17 145.0 665.0
18 415.0 635.0
19 510.0 875.0
20 560.0 365.0
21 300.0 465.0
22 520.0 585.0
23 480.0 415.0
24 835.0 625.0
25 975.0 580.0
```

Berlin 52



Genetic Algorithm



Current Work

Selection

1, Roulette Selection

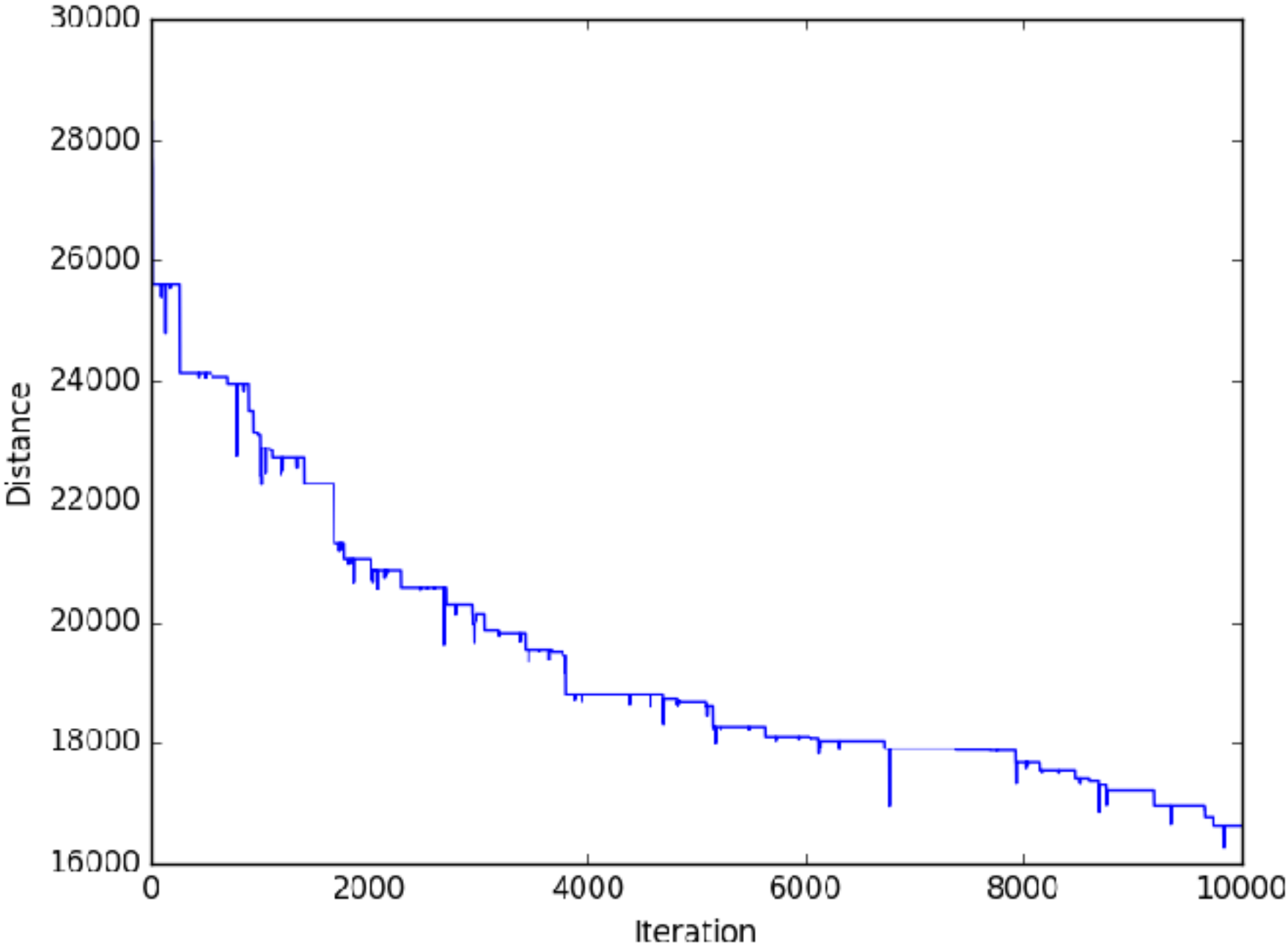
2, Elite Selection

Roulette Selection

Naïve Version:

$$p_i = \frac{S_i}{\sum_{i=0}^{n} S_i}$$

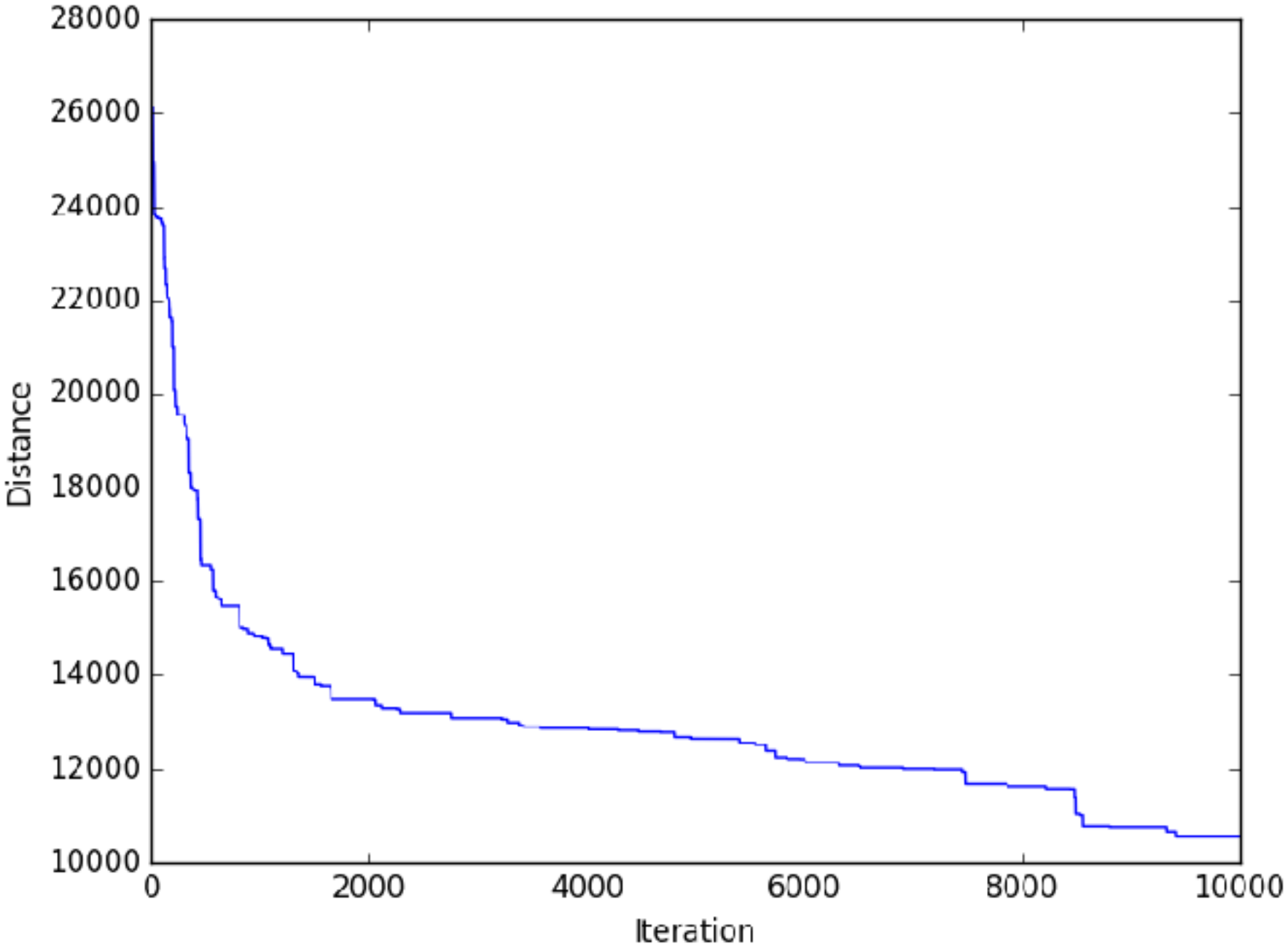
Reorder p_i list, shortest candidate choice has the biggest probability to be chosen.


30000
28000
26000
24000
22000
20000
18000
16000
Distance
0
2000
4000
6000
8000
10000
Iteration

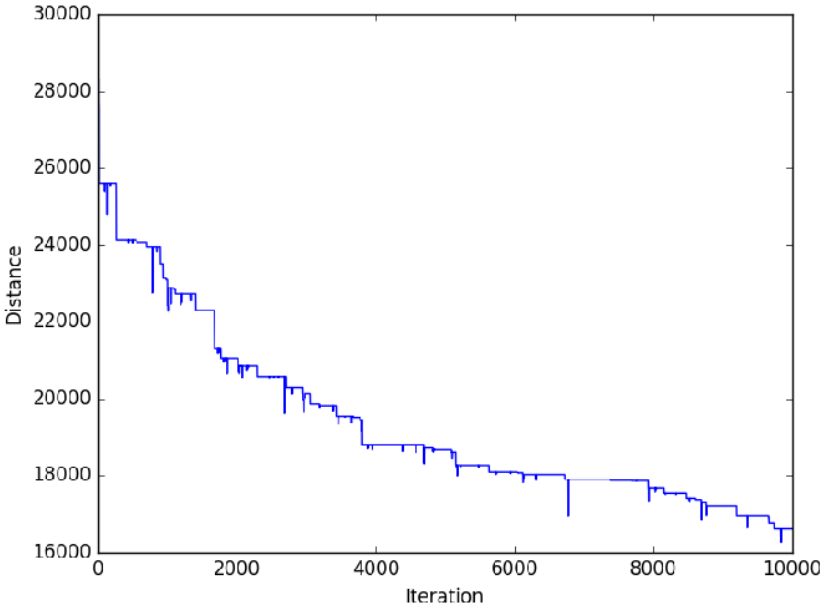
Roulette Selection

Advance Version:

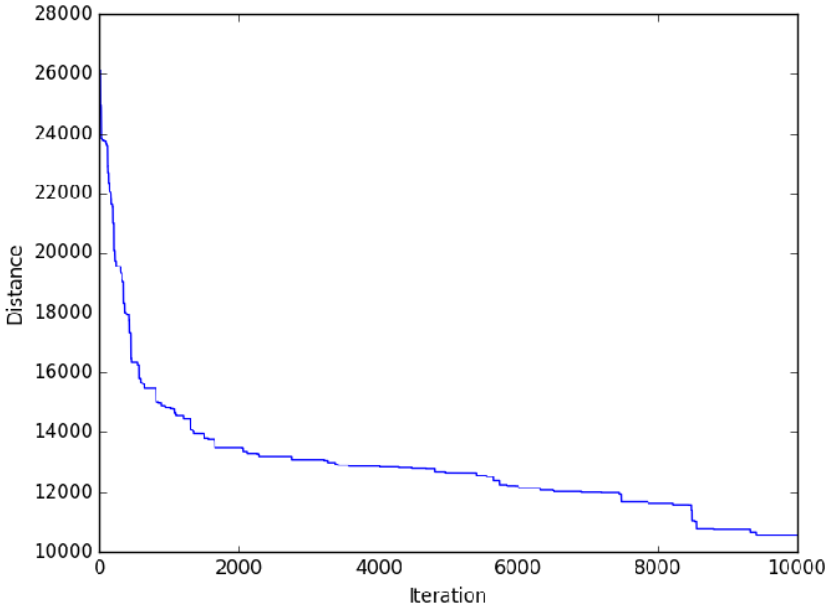
$$p_i = \frac{S'_{max} - S'_i}{\sum_{i=0}^{n}(S'_{max} - S'_i)}$$


28000
26000
24000
22000
20000
18000
16000
14000
12000
10000
Distance
0
2000
4000
6000
8000
10000
Iteration

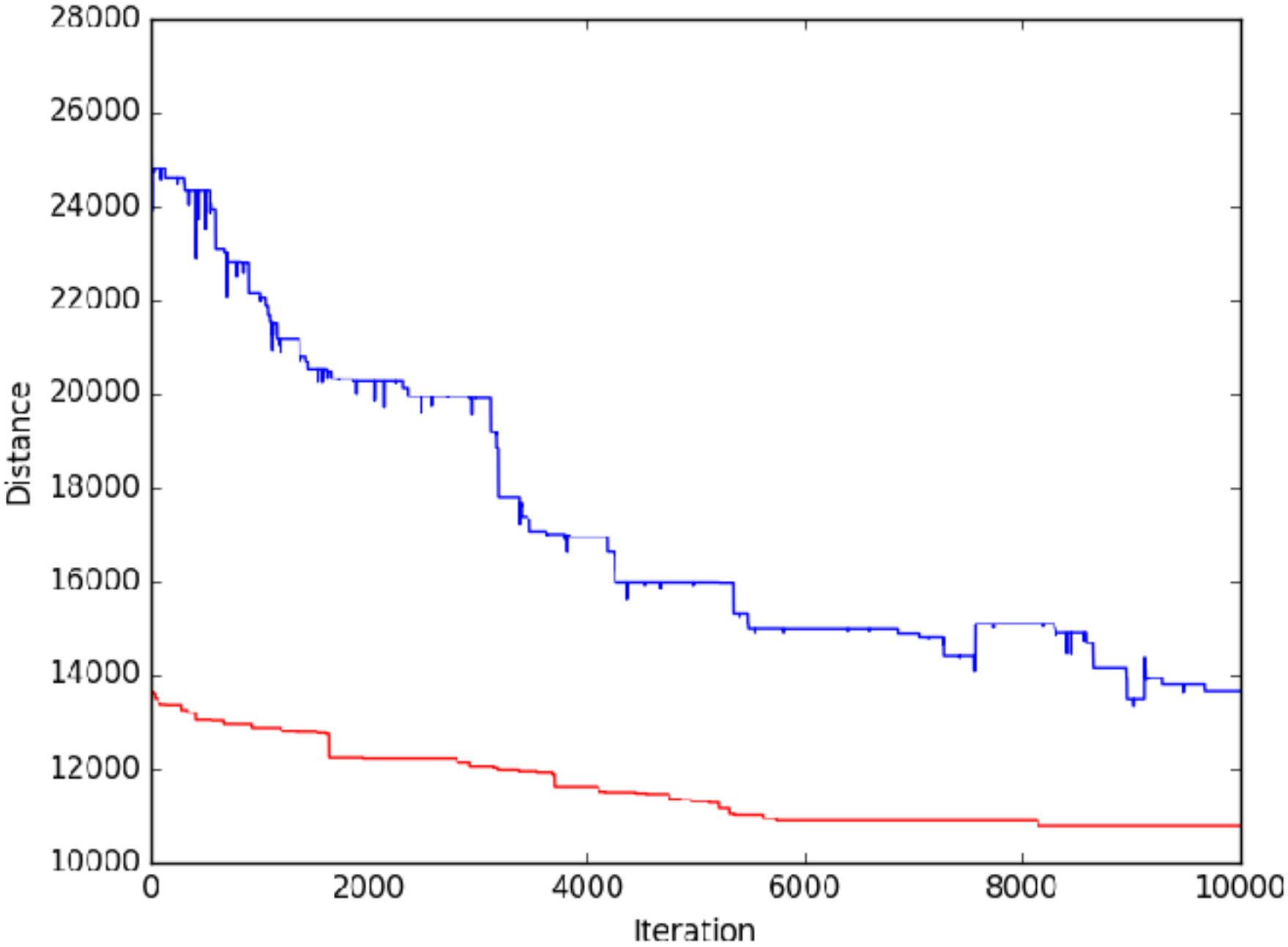
Roulette Selection



Naïve Version



Advance Version


28000
26000
24000
22000
20000
18000
16000
14000
12000
10000
Distance
0
2000
4000
6000
8000
10000
Iteration

Mutation

A bug during programming hint me:

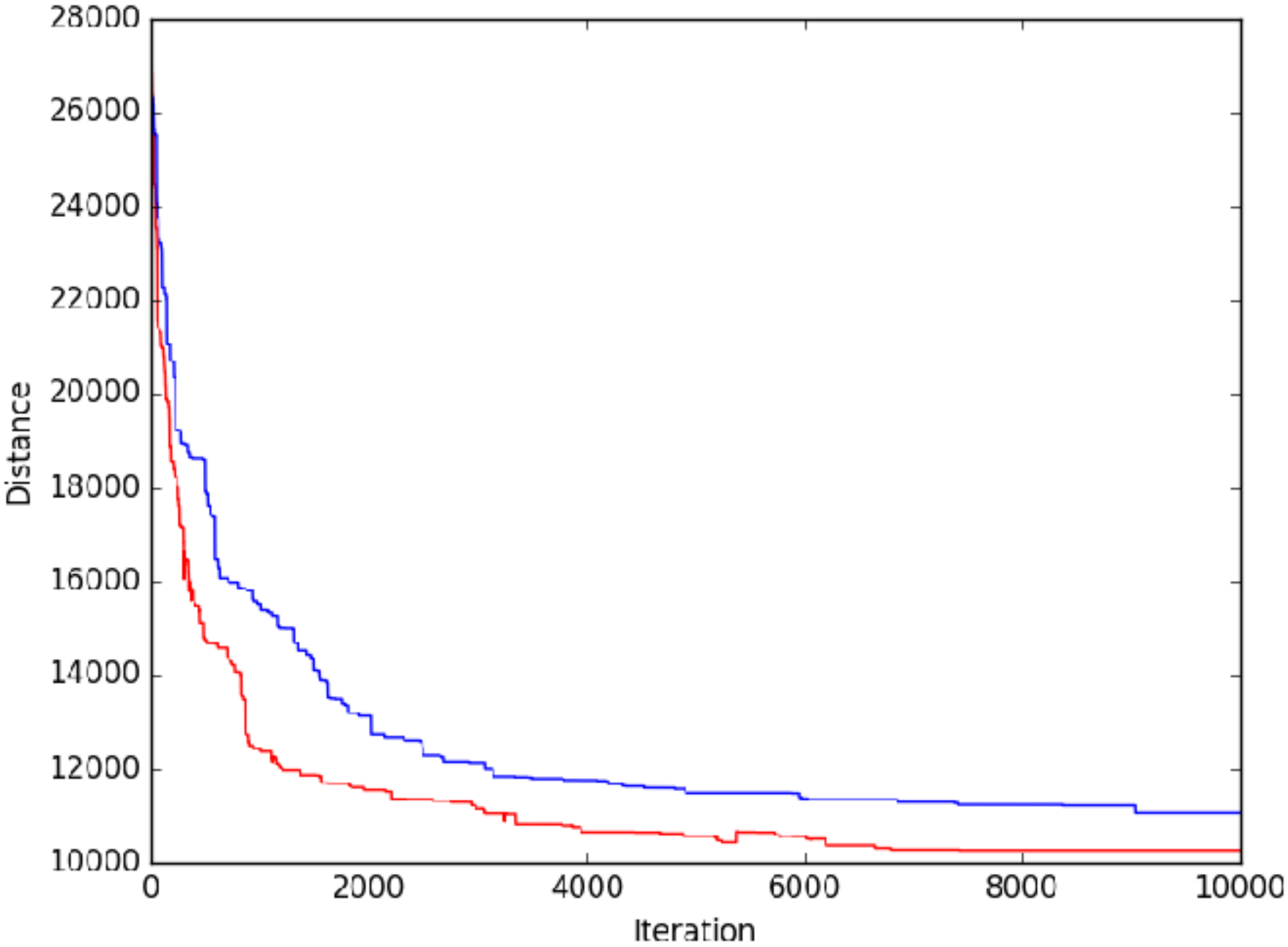
$$S'_{max} = S'_i$$

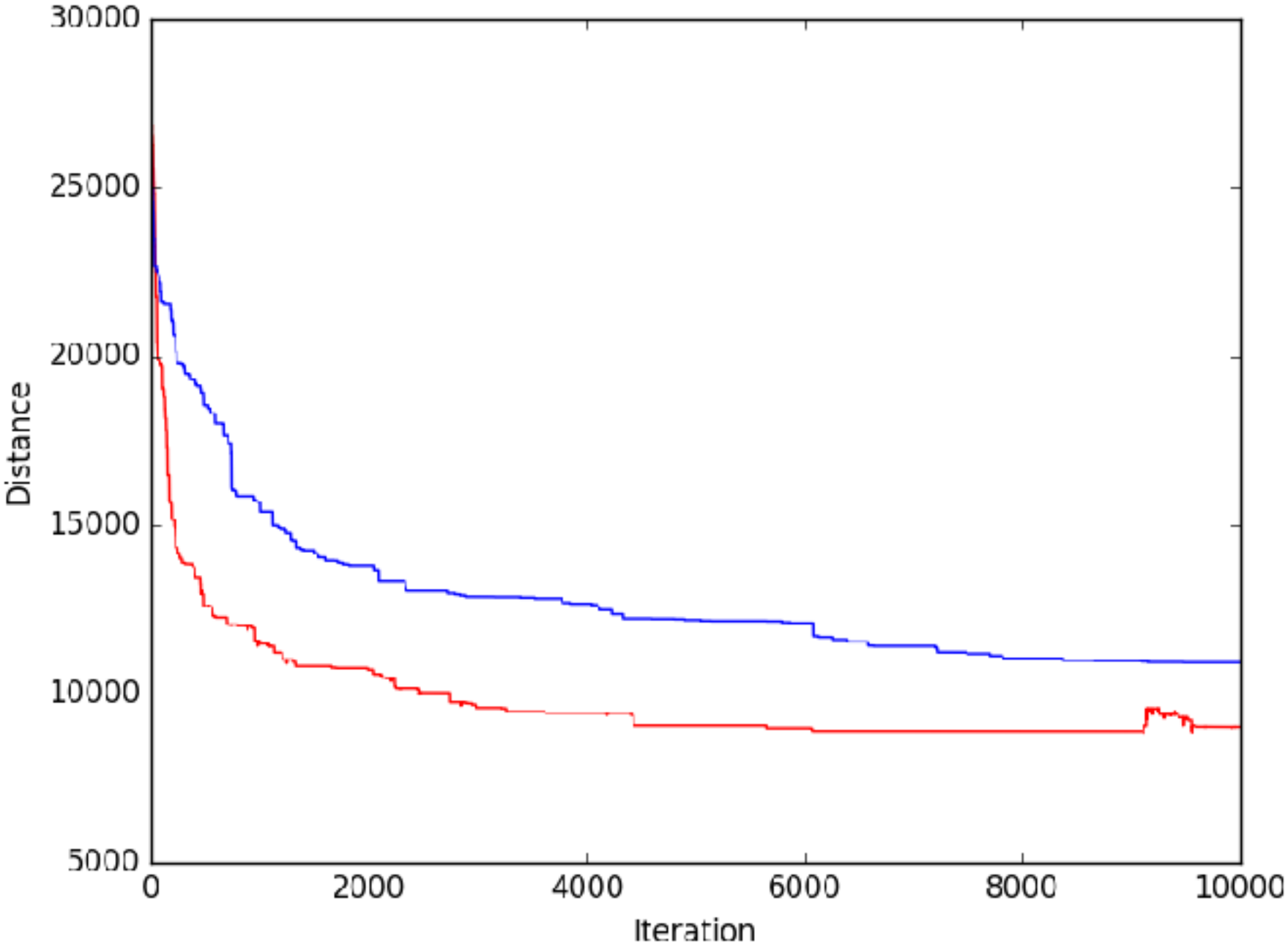
$$\sum_{i=0}^{n} (S'_{max} - S'_i) = 0$$

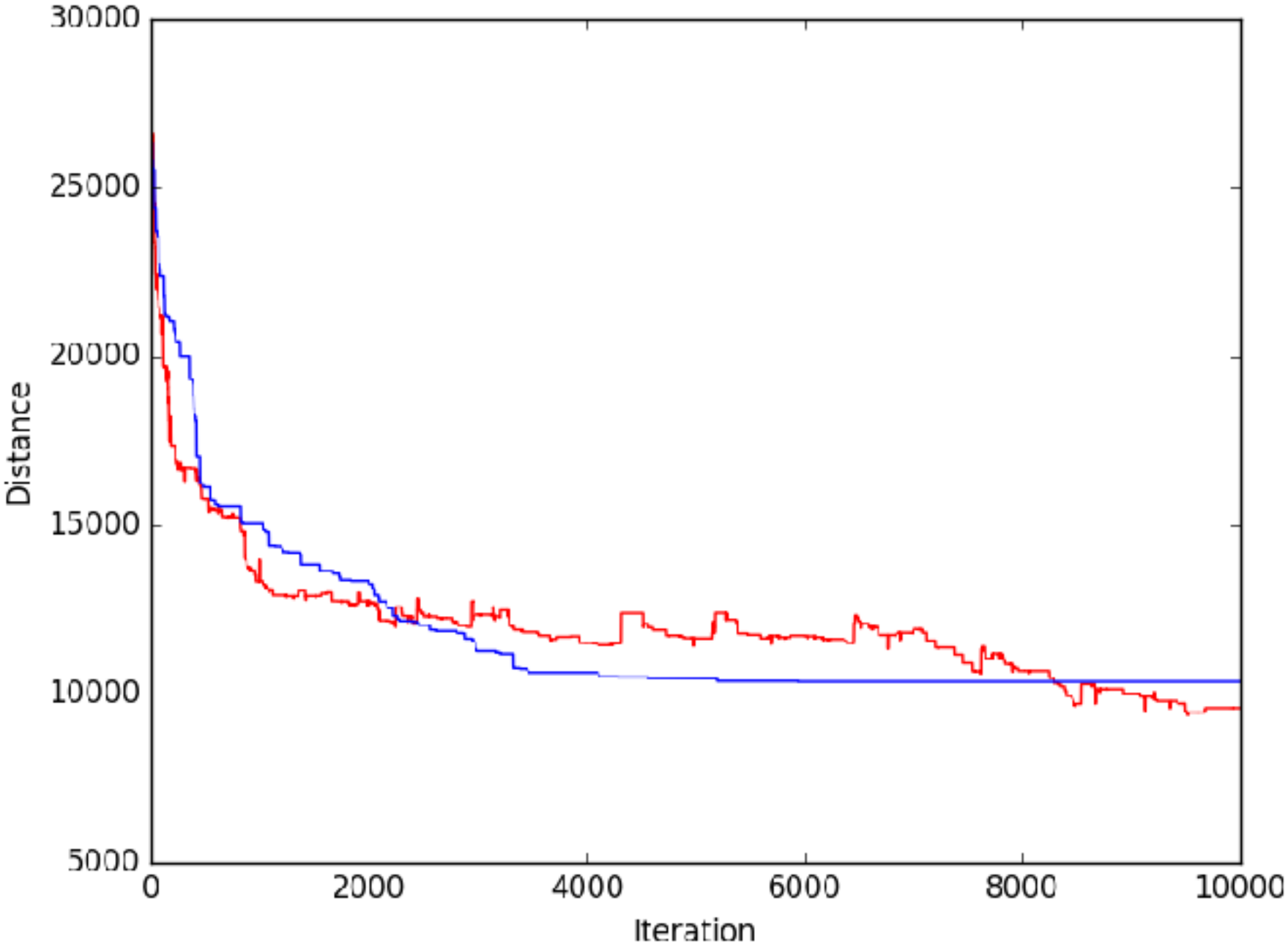
Mutation

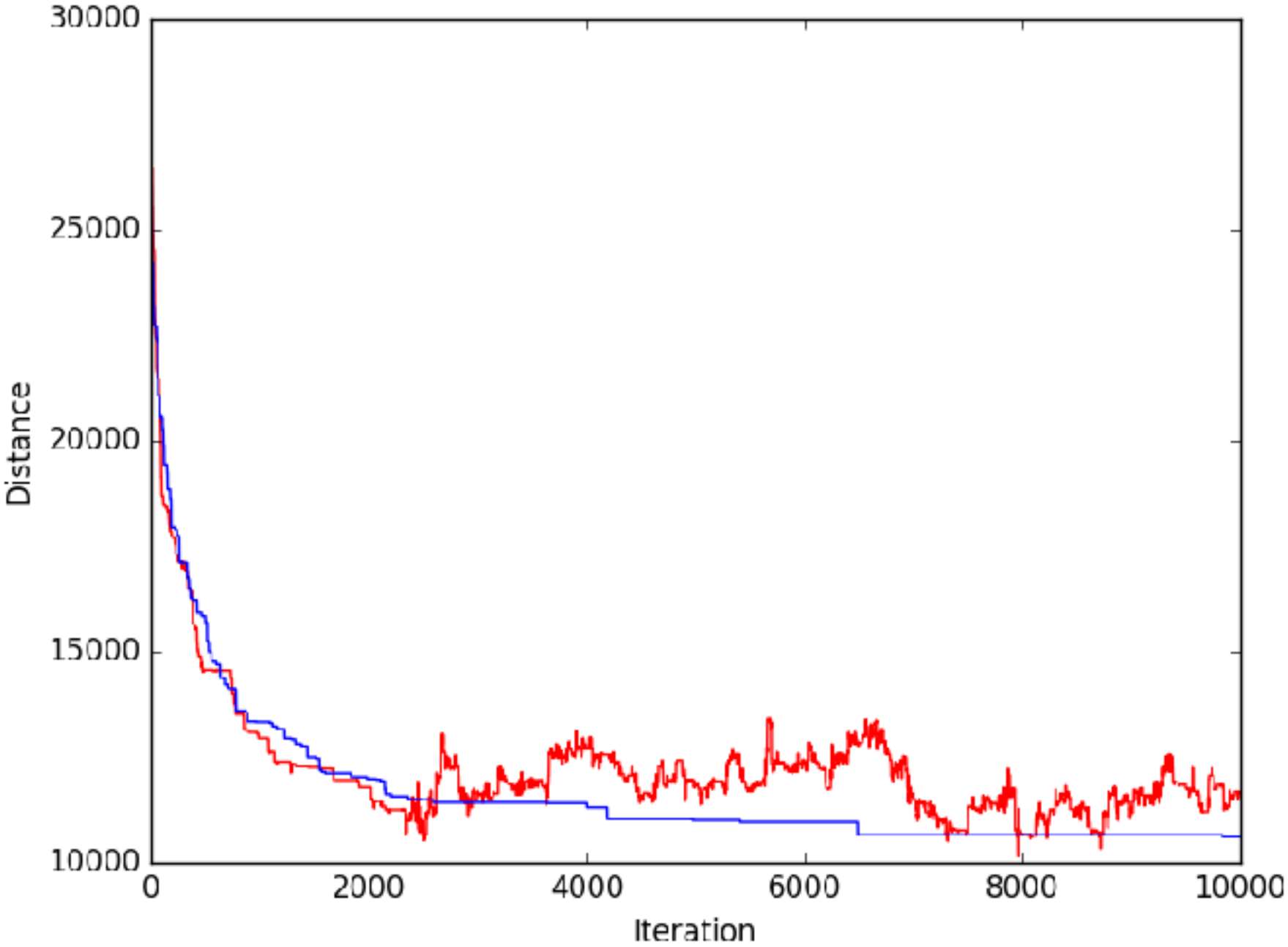
When $\sum_{i=0}^{n} (S'_{max} - S'_i) < a$

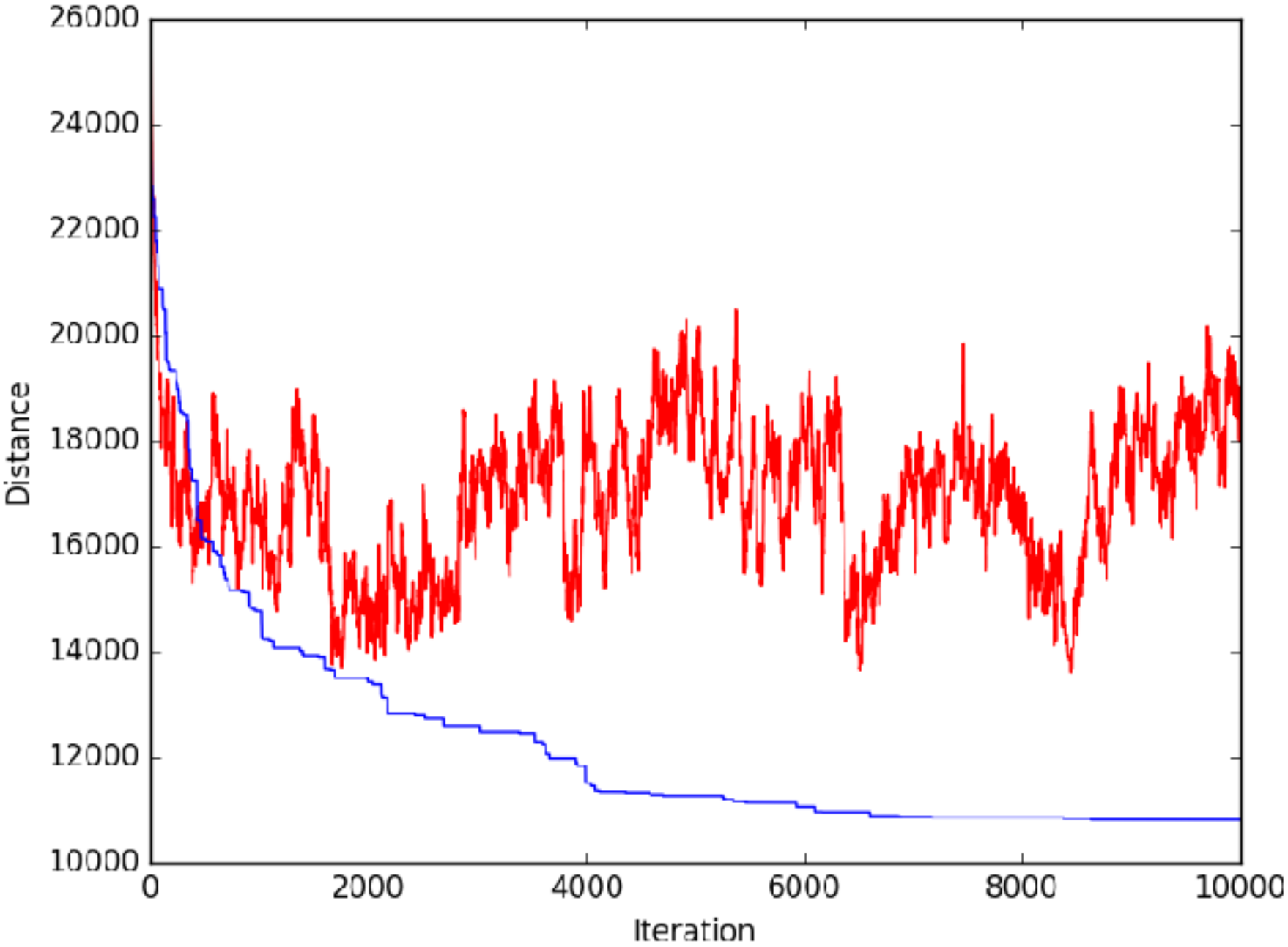
```
mutation_rate = mutation_rate*b  // b>1
```


28000
26000
24000
22000
20000
18000
16000
14000
12000
10000
Distance
0
2000
4000
6000
8000
10000
Iteration


30000
25000
20000
15000
10000
5000
Distance
0
2000
4000
6000
8000
10000
Iteration

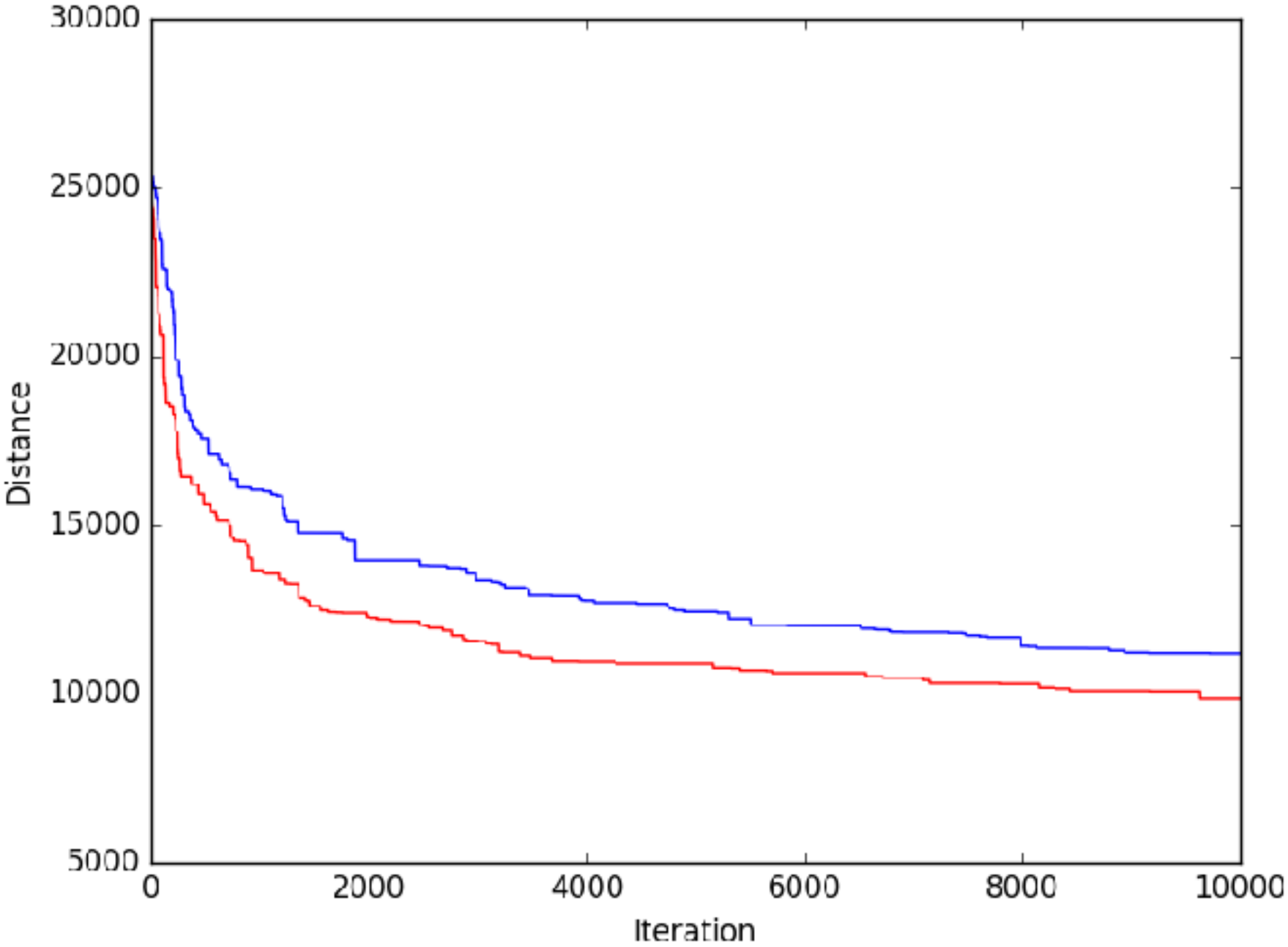

30000
25000
20000
15000
10000
5000
Distance
0
2000
4000
6000
8000
10000
Iteration


30000
25000
20000
15000
10000
Distance
0
2000
4000
6000
8000
10000
Iteration


26000
24000
22000
20000
18000
16000
14000
12000
10000
Distance
0
2000
4000
6000
8000
10000
Iteration

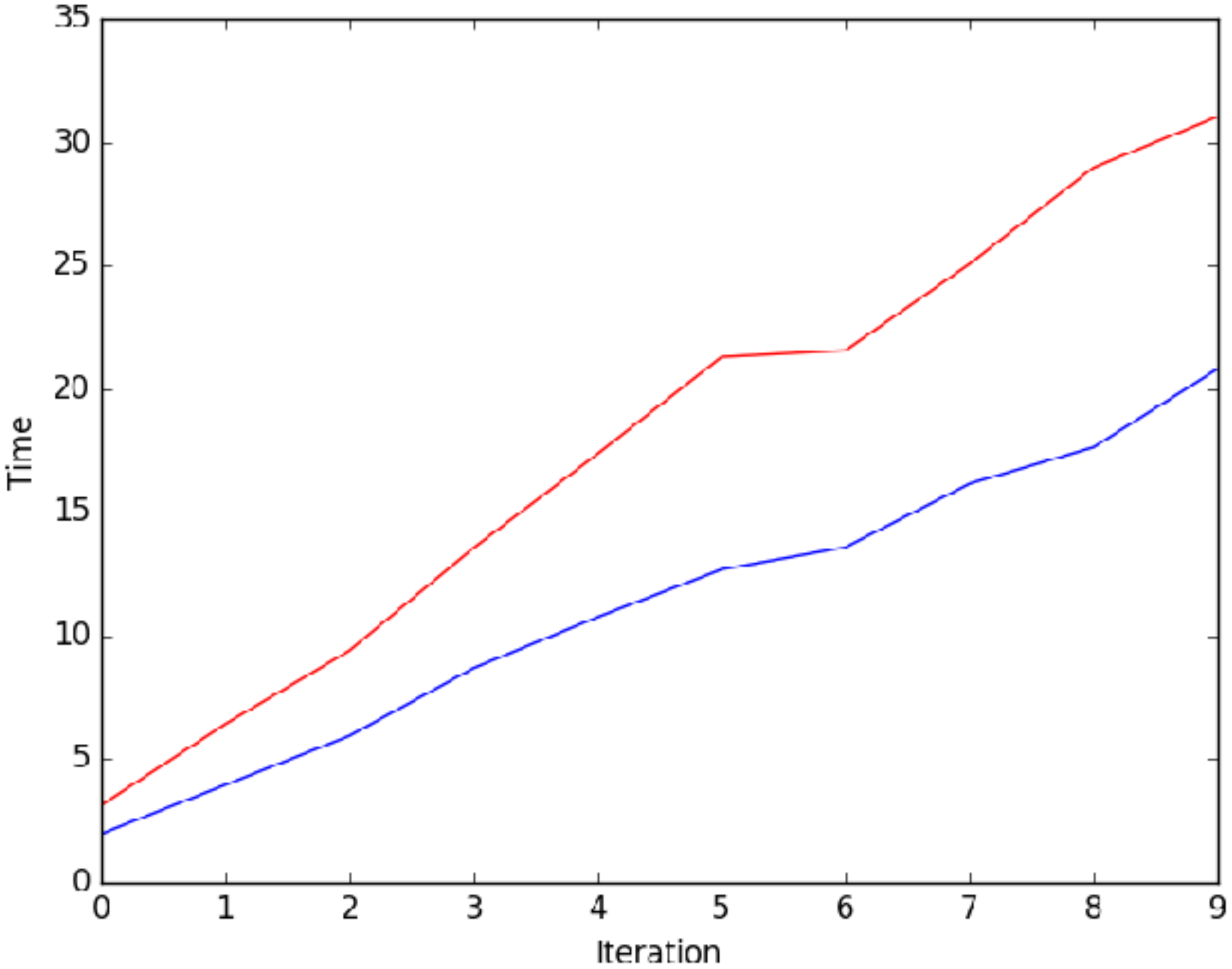
Elite Selection

Use the best choice in candidate list to crossover with other choices


30000
25000
20000
15000
10000
5000
Distance
0
2000
4000
6000
8000
10000
Iteration

Other Improvement

Build up a Distance Matrix instead of calculating sum distance every time.


35
30
25
20
15
10
5
0
Time
0
1
2
3
4
5
6
7
8
9
Iteration

Future Work

Future Work

1, Use graph theory to improve the selection method

2, Compare with ACO

Thank you for listening!