Probabilistic Algorithms Project

Comparing heuristics for TSP

Laurent Hayez

January 9, 2017

1 Introduction

The Traveling Salesman Problem (TSP) is an old problem, having written sources as old as almost 300 years. The problem consists of a person needing to visit n cities, but with constraints. The constraints are that the person must visit each of the n cities only once, and must do it in an optimal way, i.e., the total traveled distance must be the smallest possible.

This problem is hard to solve deterministically. Indeed, to find the optimal solution, we would need to test every possible path in the graph composed of the n cities, and the paths between them. If there are n cities, the number of paths to test is $n!$, and the computations become impractical when n is as small as 20.

This hardness justifies the need to use heuristics, and probabilistic algorithms to solve the problem. Of course, there is no guarantee that we will find the optimal path, but the heuristics return good enough paths. The goal of this project is to compare a few heuristics that can be used to solve the TSP. We will compare the best solutions found by each algorithms, the differences for the routes, the performance of the algorithms, and pairwise statistical comparison of the algorithms.

2 Comparison of loss values

Denote by $\mathcal{L}$ the sequence $\{L(\sigma_1^*), \ldots, L(\sigma_m^*)\}$ of the best solutions generated after calling each implemented algorithms $m = 30$ times. We will present tables showing the minimum of $\mathcal{L}$, the maximum, the average and the 95% condidence interval for this average for each implemented algorithms.

Table 1: Construction heuristics

Algorithms	min($\mathcal{L}$)	max($\mathcal{L}$)	mean($\mathcal{L}$)	Confidence interval at 95%
best insertion	1470.1821	1563.2285	1516.5628	[1506.6866, 1526.4391]
shortest edge	1584.0766	1753.079	1671.8119	[1656.668, 1686.9558]

We notice that the shortest edge heuristic is not good compared to the best insertion heuristic. The worst solution found by the best insertion heuristic is still better than the best solution found by the shortest edge heuristic. After multiple runs, the results do not come from unlucky runs, as every run resulted in approximately the same values. It could be either because there is a mistake in the implementation of the shortest edge heuristic, or the heuristic is not good enough. The approach is similar to the nearest neighbor

Table 2: Greedy local search

Algorithms	min($\mathcal{L}$)	max($\mathcal{L}$)	mean($\mathcal{L}$)	Confidence interval at 95%
swap	1459.2188	1557.0646	1506.8777	[1497.346, 1516.4094]
translation	1443.1734	1513.6743	1483.3338	[1476.954, 1489.7137]
inversion	1441.9932	1538.6809	1486.538	[1477.6003, 1495.4757]
mixed	1417.9906	1497.824	1457.8618	[1451.398, 1464.3255]

Table 3: Simulated annealing with Metropolis criterion

Algorithms	min($\mathcal{L}$)	max($\mathcal{L}$)	mean($\mathcal{L}$)	Confidence interval at 95%
swap	1480.4799	1570.6594	1524.4871	[1515.8524, 1533.1217]
translation	1496.309	1593.6313	1526.9437	[1517.9352, 1535.9522]
inversion	1482.4386	1548.8955	1517.5976	[1510.4491, 1524.746]
mixed	1484.3271	1570.1874	1520.7171	[1512.8956,1528.5386]

Table 4: Simulated annealing with heat bath condition

Algorithms	min($\mathcal{L}$)	max($\mathcal{L}$)	mean($\mathcal{L}$)	Confidence interval at 95%
swap	1475.0552	1592.2256	1530.0634	[1520.7394, 1539.3875]
translation	1482.6547	1557.8635	1525.6696	[1518.2635, 1533.0756]
inversion	1473.0003	1567.3859	1516.8026	[1509.7888, 1523.8164]
mixed	1474.1733	1576.7916	1513.6764	[1504.5419, 1522.8109]

heuristic, and we know for a fact that the nearest neighbor approach does not necessarily yield good approximations, and we can make the paths arbitrarily longer than the optimal one.

The improvement heuristics (greedy local search and simulated annealing) all had their first solution coming from the best insertion construction heuristic. We see for the greedy local search that the swap approach has the worst results, but we still see an improvement compared to just using the best insertion heuristic. The translation and inversion moves yield similar results on average. We see that the mixed approach provides the best improvement, as it reduces the path length the most.

The Simulated Annealing algorithm with both conditions has approximately the same results, even though it seems that the heat bath condition with mixed small moves yields the best results. The greedy local search returns better results than the SA algorithm.

3 Trace of best solution

We show the trace of the best tour found after 30 rounds of each algorithms. For the improvement algorithms and the SA algorithm, the first tour is constructed with the best insertion algorithm. The traces are shown in Figures 1 to 4.

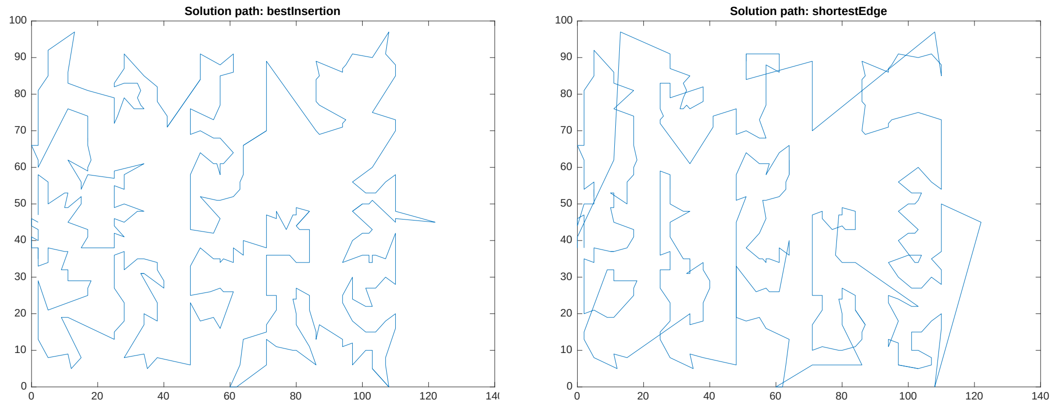

Figure 1: Solution path for best insertion and shortest edge heuristics

4 Performance plots

We now present the performance plots, which show the evolution of the path length over the different iterations. For the greedy local search, the path length is represented as a function of the number of steps of the algorithms, and for the simulated annealing algorithms, the path length is a function of the temperature. The performance plots are represented in Figures 5 to 7.

On plot 5, we see that for each of the small moves, the gain in path length is not that huge, but the translation and inversion moves seem to give better results than the swap move. The biggest gain is obtained by using the mixed moves, as we saw in Section 2.

On the other hand, for the SA algorithm, due to the fact that the temperature is very hot at the beginning, we accept worse solutions, with a big path length, but the path length decreases very fast near its optimal value. The best result seem to be achieved when using the inversion small moves, as the stopping condition is met when the temperature is almost 10^{10} times bigger, which is a huge improvement. In this case, using translations as small moves seems to be the worst case, as we need to decrease the temperature a lot before meeting the stopping criterion.

If we use the metropolis criterion, using mixed small moves seems to be a good idea, as the result are not so bad, and mixed moves are versatile. When using the heat bath condition, the mixed moves are not the best choice, but this may come from an unlucky run.

5 Pairwise comparison of algorithms

In this section, we will compare the algorithms we used by making statistical tests. We will use the following statistical test. We denote by T the statistic

$$T = \frac{\overline{x} - \overline{y}}{\sqrt{\frac{\sigma_X^2}{n_X} + \frac{\sigma_Y^2}{n_Y}}},$$

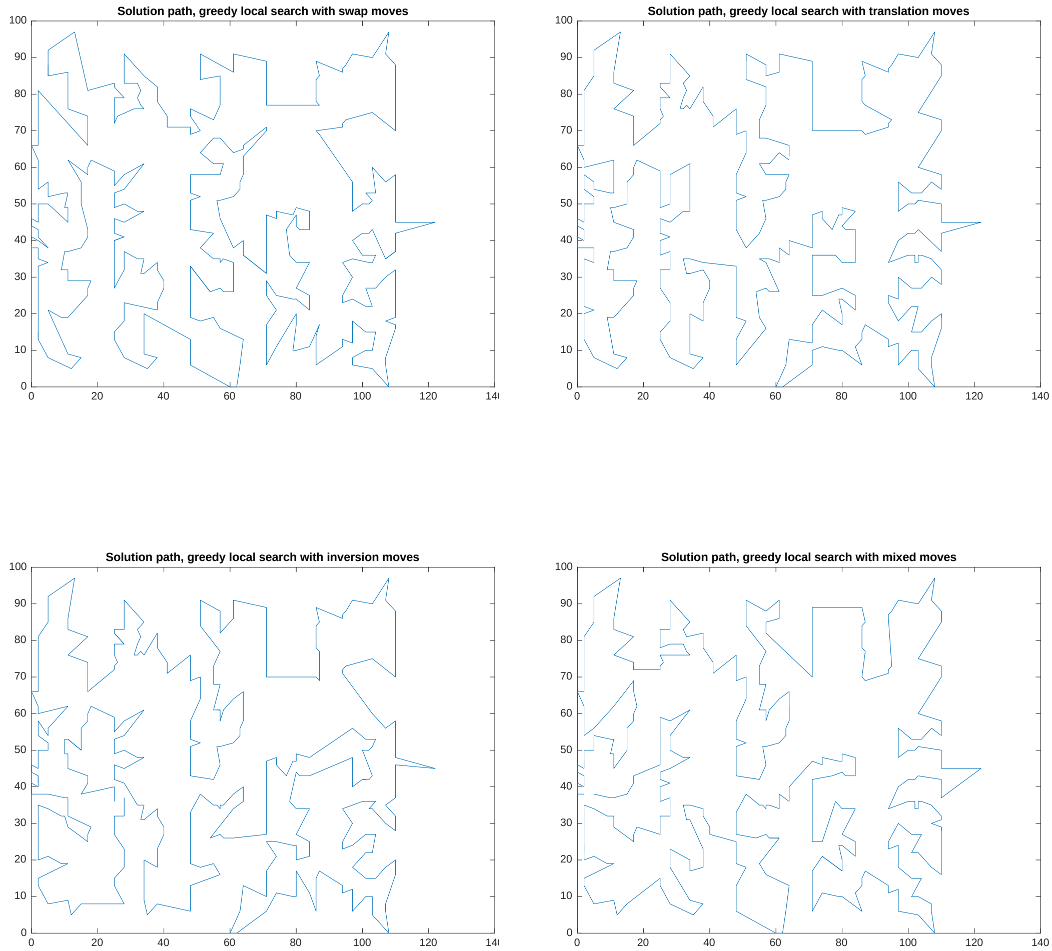

Figure 2: Solution path greedy local search heuristics

where T follows a Student distribution (t_m) with m degrees of freedom, where

$$m = \frac{\left(\frac{\sigma_X^2}{n_X} + \frac{\sigma_Y^2}{n_Y}\right)^2}{\frac{(\sigma_X^2/n_X)^2}{n_X+1} + \frac{(\sigma_Y^2/n_Y)^2}{n_Y+1}} - 2.$$

The acceptance interval is $[-\beta, \beta]$ where

$$\beta = t_m^{\alpha/2}\sqrt{\frac{\sigma_X^2}{n_X} + \frac{\sigma_Y^2}{n_Y}}.$$

In our case we choose $\alpha = 0.5$.

The results are shown in Table 5.

We see that the simulated annealing achieves more or less the same performance whether we use the metropolis condition, or the heat bath criterion. However, the best insertion heuristic is more powerful than the shortest edge heuristic.

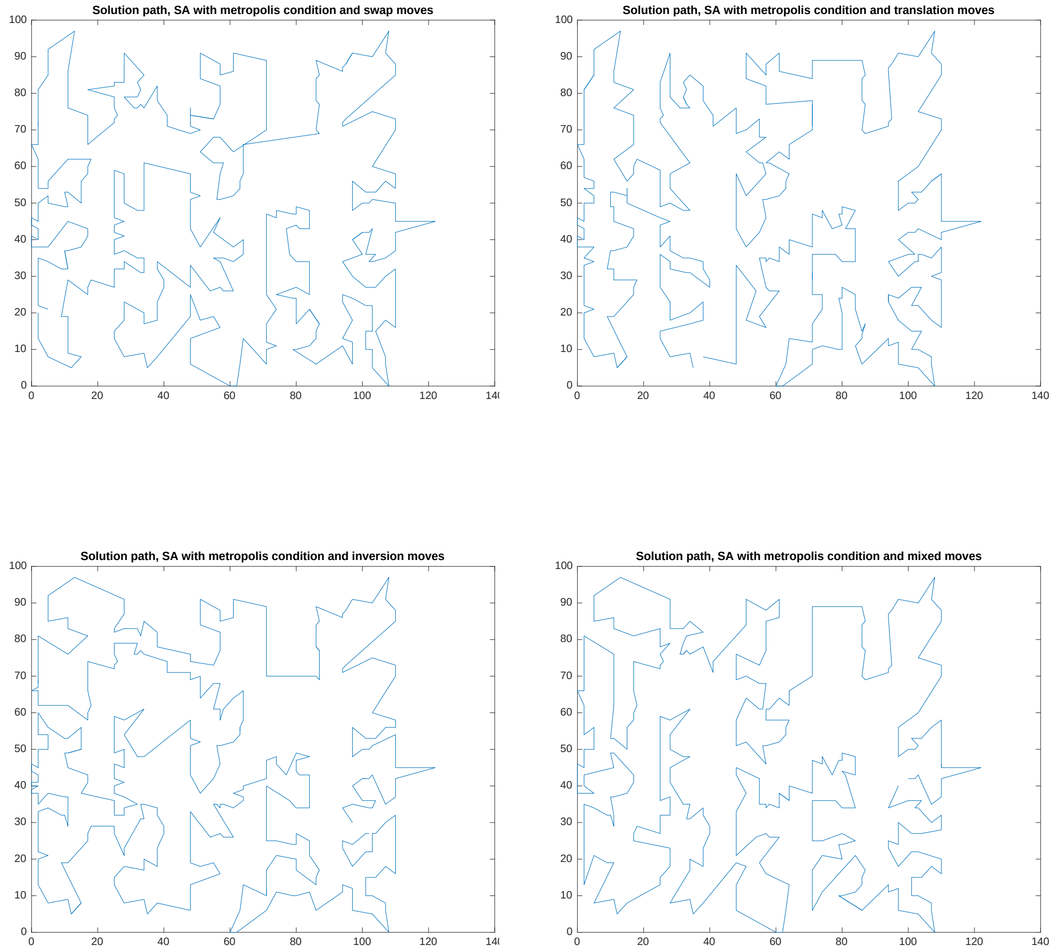

Figure 3: Solution path SA algorithm with metropolis criterion

Table 5: Pairwise comparison

Algorithms	μ_X	μ_Y	T	β	Best algorithm
best insertion vs shortest edge	1521.2787	1673.6961	-19.3326	5.3528	Best insertion
metropolis vs heat bath	1526.7489	1521.7953	0.80944	4.1537	Equivalent

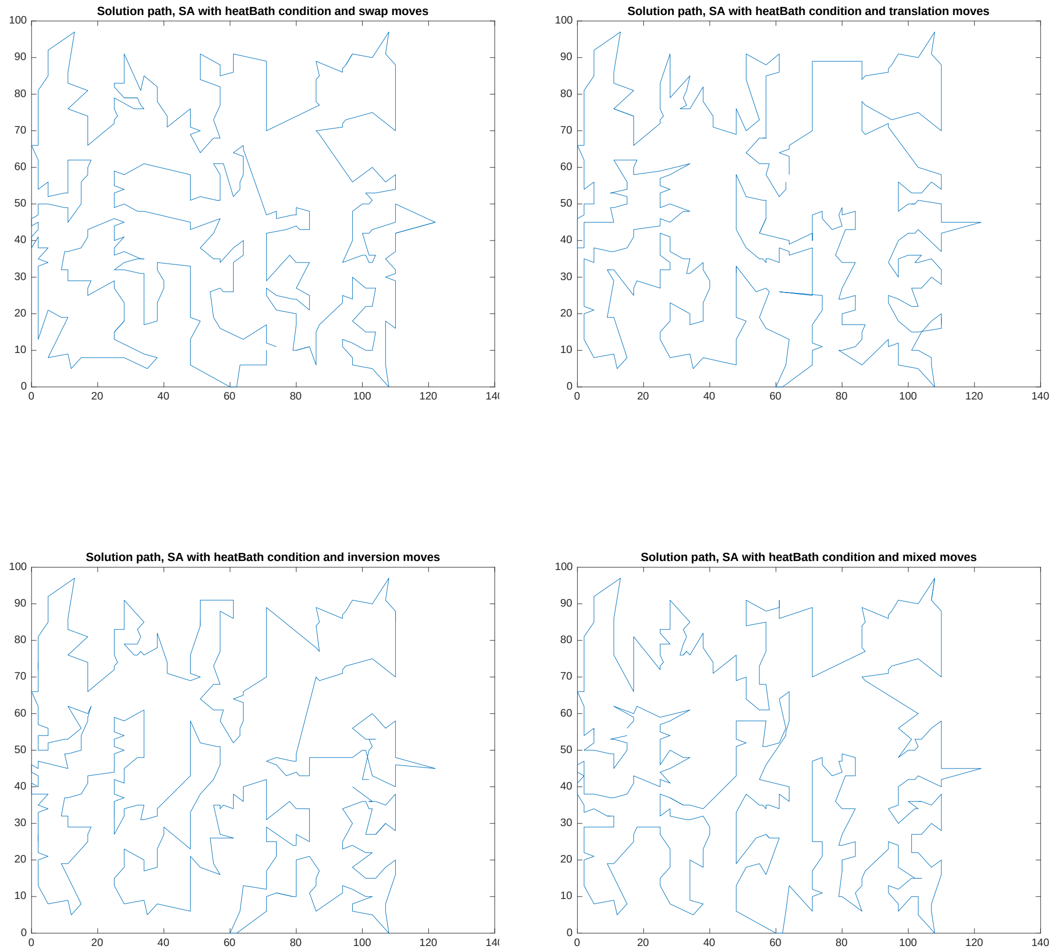

Figure 4: Solution path SA algorithm with heat bath criterion

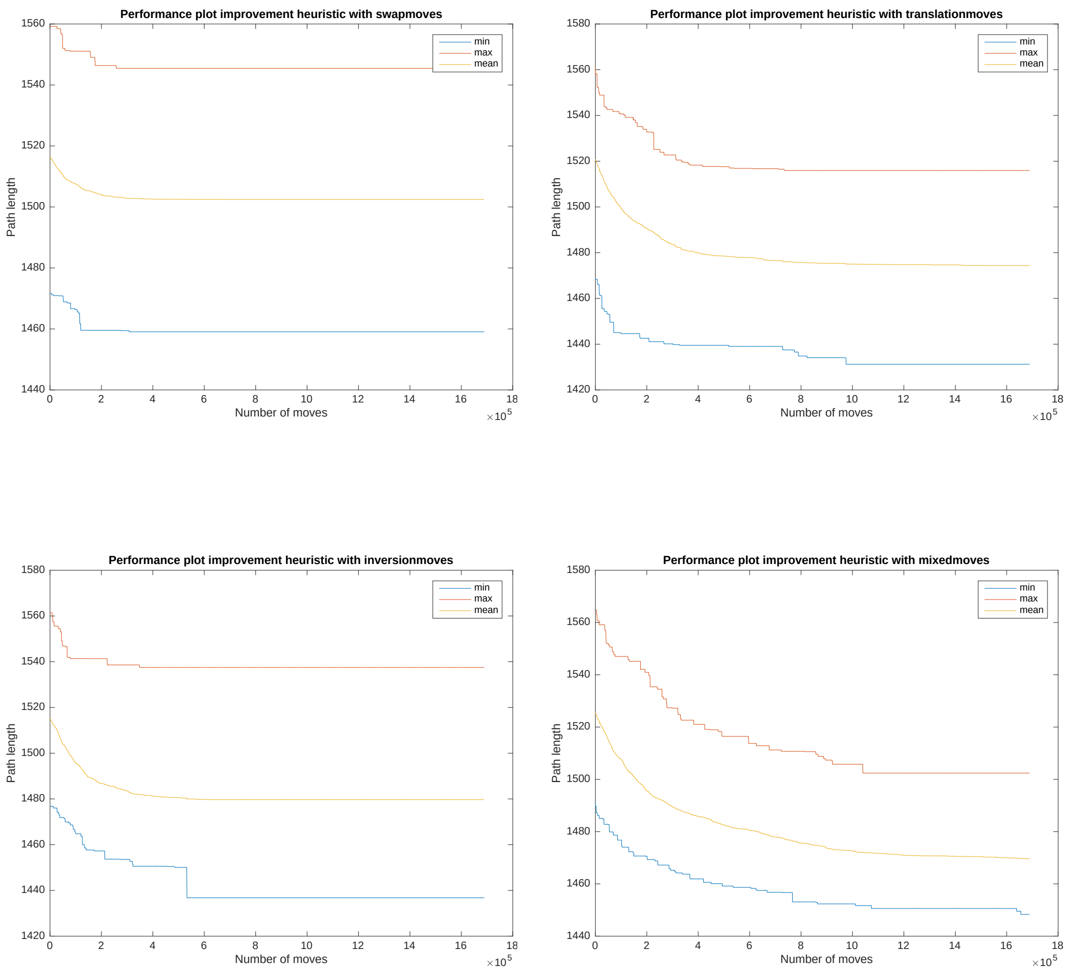

Figure 5: Performance plots for the greedy local search

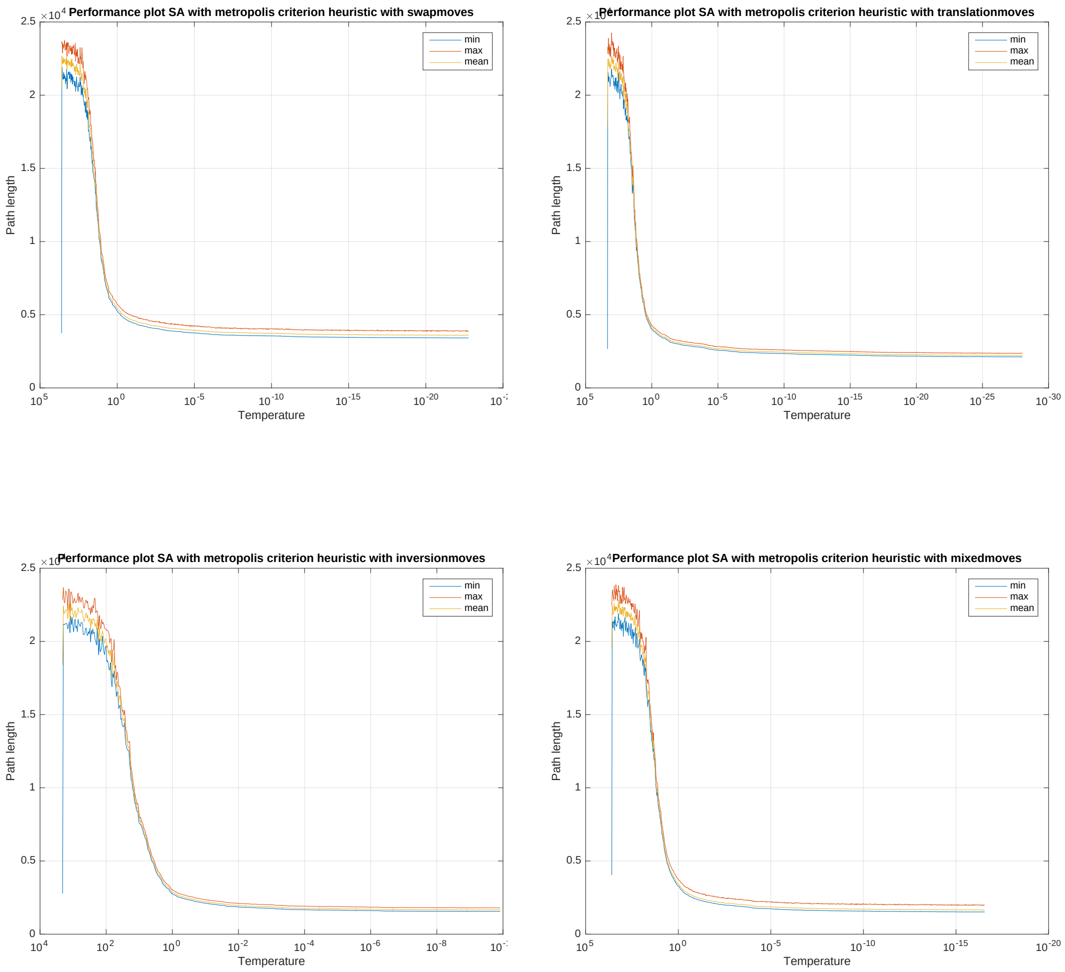

Figure 6: Performance plots for the SA algorithm with the metropolis criterion

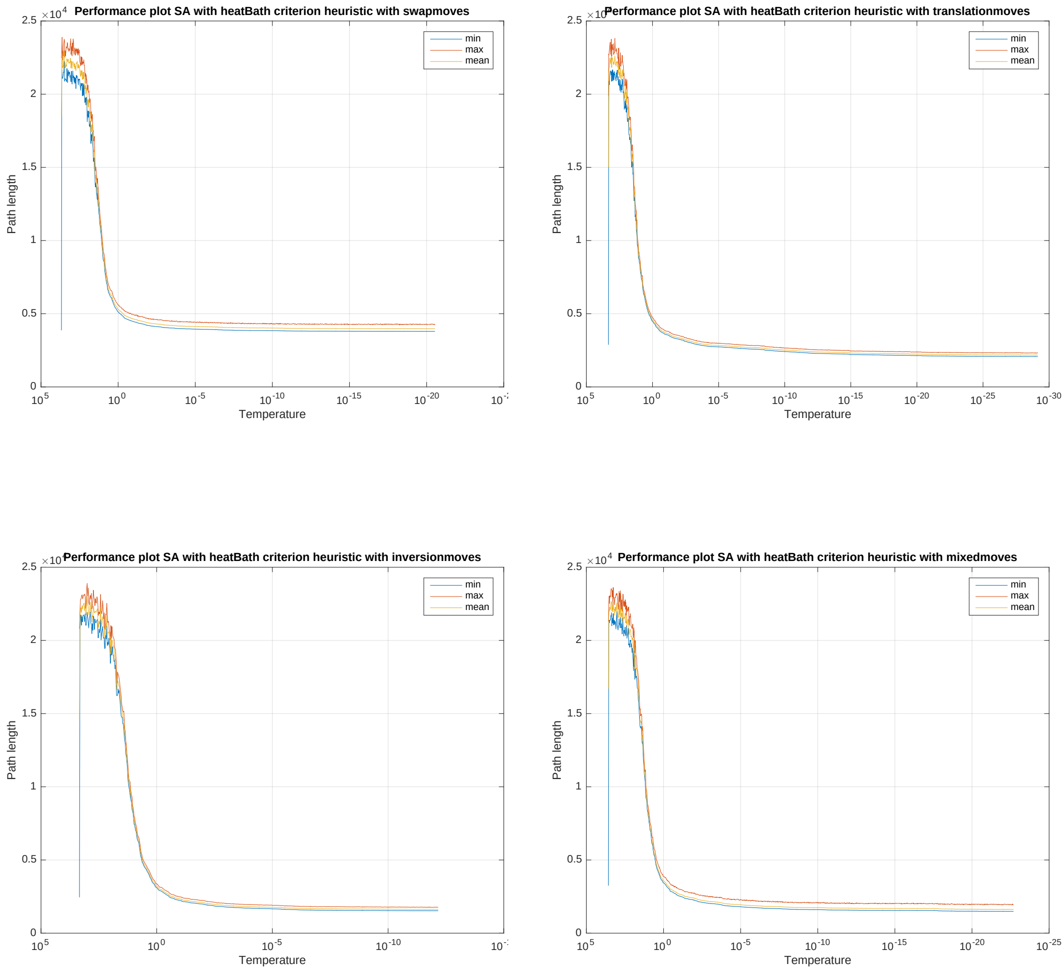

Figure 7: Performance plots for the SA algorithm with the heat bath condition