Evolutionary Computation: The Optimised Odyssey

2512308

February 28, 2023

1 Rubric

1.1 Task

Implement Simulated Annealing (SA) and Genetic Algorithm (GA) to solve the Odyssey of Ulysses 22 cities Travelling Salesman Problem (TSP).

1. Create an executable Simulated Annealing (SA) and Genetic Algorithm (GA) in Matlab or another language. For GA, choose an appropriate encoding scheme.
2. Execute a maximum of 100 trial runs for each algorithm to tune the parameters (literature search the appropriate parameter ranges). Record parameters and performance.
3. Execute 30 independent runs with 10000 fitness (objective function) evaluations for each algorithm (SA: 10000 interactions; GAs: depends on the population size and the number of maximum generations). Record the average distance and standard deviation from the results over the 30 runs for each algorithm.
4. Compare the results for these two algorithms statistically using a Wilcoxon signed-rank test (WST).

1.2 Report

1. Introduction: justify SA and GA design decisions, e.g., the encoding scheme for GA, and explain these algorithms by using a flowchart and pseudo-code.
2. Discuss the parameters and how they impacted the performance of the algorithms.
3. You should also list all the average results and standard deviations obtained from the 30 runs of the two algorithms
4. Discuss how you compare the results obtained by SA and GA statistically.

1.3 Marking Scheme (25 points)

1. Correct implementation of SA (6 marks)
2. Correct implementation of GA (6 marks)
3. Report: Satisfied requirements 6 and 7 of a well-presented, well-explained report, including critical reflection and discussion of results (13 marks)

1.4 Submission

Submit a single PDF report and a separate source code file (.m, .py, .c, not notebook). If necessary, you may use .tar or .tgz or .targz (if possible, avoid rar). Also, submit compiled binary executable (if applicable). Submit the report and code separately.

2 Simulated Annealing

2.1 Introduction

Definition ∼ *The random perturbation of feasible solutions to combinatorial optimisations. Sub-optimal perturbations are accepted randomly as a decreasing function of current time or fitness to move from local to global optima. (Rutenbar, 1989)*. Below is a general SA algorithm:

Algorithm 1: General SA

```
For {$x_k$, neighbour($x_k$) = $x'_k$} ∈ $\mathcal{X}$, objective function $E()$, and step size $temp(k)$:
Input: $\mathcal{X}, E(x), temp()$
Output: $x_{k_{max}}$
$x = x_0$;
$x_0 = x_{best}$;
while $k < k_{max}$ do
    $T \leftarrow temp(k)$;
    Compute $E(x_k)$;
    Compute $E(\text{neighbour}(x_k))$;
    if $P(E(x_k), E(x'_k), T) > rand(0,1)$ then
        $x_k \leftarrow x'_k$;
    if $E(x_k) > E(x_{best})$ then
        $x_{best} \leftarrow x_k$
    $k \leftarrow k + 1$
return $x_{k_{max}}$
```

2.2 Parameters

The subsequent discussion seeks to solve the Travelling Salesperson Problem (TSP) via SA. To render algorithm 1 executable, one must define its free parameters. For ease of exposition, table 1 displays the justification for each parameter:

Parameter Value	Explanation	Reference
1. Solution space $\mathcal{X}$	The assignment TSP includes 22 cities. A complete graph has $\frac{n(n-1)}{2}$ edges and $(n-1)!$ Hamiltonian cycles (HCs). The graph Ulysis22TSP forms has 231 edges and 21! HCs. The solution space is the set of all HCs.	Goddard, 2009, Diestel, 2017
2. Objective function $E(x)$	$E(x)$ is the Euclidean distance of HC 'x'.	Černý, 1985
3. Initial solution x_0	Choose a pseudo-random HC (permutation of nodes 1-22)	Kirkpatrick et al., 1983
4. Initial temperature T_0	$T_0 = \Delta E_{max}, T_0 = 20$	Kirkpatrick et al., 1983
5. Cooling schedule - *temp()*	$T_{t+1} = \alpha T_t$, where $\alpha = 0.95$	Kirkpatrick et al., 1983
6. Acceptance Threshold	Accept if $rand(0,1) < P(E) = e^{(\frac{-E}{T})}$	Kirkpatrick et al., 1983
7. Neighbourhood function - *neighbour()*	For random indices $i, j \in [1, 22]_{\mathbb{Z}}$, s.t $i \neq j$, swap the ith and jth nodes in the HC.	No ref.
8. Stopping condition k_{max}	$k_{max} = 10{,}000$	(See Rubric)

Table 1: SA Parameters

2.3 Results

Run	Winning Tour Length (3DP)
1.	94.225
2.	93.548
3.	80.456
4.	92.156
5.	78.818
6.	76.378
7.	106.730
8.	79.876
9.	76.140
10.	78.616
11.	79.079
12.	91.068
13.	79.667
14.	85.650
15.	77.677
16.	94.235
17.	96.178
18.	78.722
19.	77.845
20.	79.481
21.	91.477
22.	96.051
23.	94.310
24.	76.448
25.	76.701
26.	79.941
27.	81.101
28.	87.008
29.	92.739
30.	92.716
Mean	85.501
SD	8.295

Table 2: SA Results

2.3.1 Sample Statistics

Table 2 displays the results of the simulated annealing algorithm. The sample average length is 85.501 with standard deviation 8.295. The minimum winning HC is 76.140, the maximum 106.730. Thus, the winning HC range is 30.590. The sample skewness is 0.636 indicating a small positive skew. The sample kurtosis is -0.578; the sample is platykurtic relative to the normal distribution.

2.3.2 Solution Sensitivity

Parameters initialised according to table 1 produced the results in table 2. When one alters parameters, one changes the HC solutions. This section considers the sensitivity of HC solutions to each parameter:

Initial solution x_0: The initial solution is a random permutation of nodes. The solutions in table 2, all have random initial solutions. Given the range (30.590), the initial solution may affect the optimal HC length. Yet, the random nature of perturbations at each SA step may also give rise to these differences.

Initial temperature T_0: An initial temperature increase from 20 to 2000 produced a Wilcoxon signed-rank test (WST) p-value of 0.558 between samples, suggesting the samples were not significantly different. The run-time decreased from 24.8 seconds to 24.2 seconds. This decrease in run-time is likely due to the randomness in the acceptance criterion at each step. Finally, the (min, max, range) tuple changed from (76.140, 106.730, 30.59) to (75.910, 100.970, 25.06). One must conduct further tests to verify whether the reduction in the range and outlying solutions is significant.

Cooling schedule: Scaling the cooling rate from 0.95 to 0.475 increases the rate of temperature decay. The sample mean HC length differs (76.140 to 88.513) and the WST p-value is 0.078647 indicating a significant difference at the 10% level. Faster cooling reduces the central tendency of HC lengths. The run-time reduced (24.8 to 24.1). This change may arise as the SA algorithm takes fewer sub-optimal steps and tends more quickly to local optima. *review: max iterations is the same

Maximum Iterations The maximum iterations parameter has the most notable effect on run-time. For example, scaling iterations from 10,000 to 1,000 reduced the run-time from 24.2 seconds to 2.42 seconds. Scaling the iterations from 10,000 to 20,000 increased the run-time to 48.2 seconds. Interestingly, the WST p-value between the 10,000 and 20,000 iteration samples was 0.571, suggesting no significant difference. The algorithm may need a much larger run-time with higher iterations to obtain significantly better HCs, or the algorithm may be approaching the graph's optimal HC length.

3 Evolutionary Algorithms

Definition $\sim$ *evolutionary algorithms are algorithms that perform optimization or learning tasks with the ability to evolve. They have three main characteristics: they are population-based, fitness-oriented, and variation-driven. (Yu and Gen, 2010)*. Below is a general evolutionary algorithm:

Algorithm 2: General Evolutionary Algorithm

```
For $X_0 \in X$ the initial population of solutions and fitness function $f()$:
Input: $X_0, f(x)$
Output: $x* \in X$
$t = 0$;
while $terminationflag! = true$ do
    selection: $X_t^c \in X_t$;
    variation: $X_t^c \rightarrow V(X_t^c)$;
    fitness calculation: $f(V(x_{it}^c))$;
    reproduction: least fit replaced by most fit offspring: $X_{t+1} : X_t^l \leftarrow X_t^{c*}$;
    $t \leftarrow t + 1$;
return $x_{best} : f(x_{best}) > f(x'_{best})$
```

3.1 Genetic Algorithms

Definition $\sim$ *genetic algorithms are a sub-class of evolutionary algorithms. Candidate solutions have a genotype to which mutation and crossover operators apply (Yu and Gen, 2010). The genotype (typically a bit-string encoding) is the defining feature of genetic algorithms (FrontlineSolvers, 2018)*. Below is a general genetic algorithm:

Algorithm 3: General Genetic Algorithm

```
For $X_0 \in \mathcal{X}$ the initial population of solutions and fitness function $f()$:
Input: $X_0, f(x)$
Output: $x* \in X$
$t = 0$;
while $terminationflag! = true$ do
    selection: $X_t^c \in X_t$;
    variation: $X_t^c \rightarrow V(X_t^c)$;
    /* During variation, mutation and crossover operators change the genotype (e.g.  bit string), not
       the candidate's solution-space representation (phenotype).                                    */
    fitness calculation: $f(V(x_{it}^c))$;
    reproduction: least fit replaced by most fit offspring: $X_{t+1} : X_t^l \leftarrow X_t^{c*}$;
    $t \leftarrow t + 1$;
return $x_{best} : f(x_{best}) > f(x'_{best})$
```

3.2 Parameters

The subsequent discussion seeks to solve the TSP via GA. To render algorithm 3 executable, one must define its free parameters. For ease of exposition, table 3 displays the justification for each parameter:

Parameter Value	Explanation	Reference
1. Solution space $\mathcal{X} = 21!HCs$	(See Table 1)	Goddard, 2009, Diestel, 2017
2. Objective function $f(x) = \|\|x\|\|_2$	(See Table 1)	Černỳ, 1985
3. Initial population $\|X_0\| = 100$	Generate 100 pseudo-random HCs. The genotype encoding is all 21! permutations of $[1, 22]_{\mathbb{Z}}$.	Deng et al., 2021
4. Selection $\beta = 0.2$	Rank HCs in X_0 by fitness, selecting the $\beta \cdot \|X_0\|$ fittest, $\beta \in [0, 1]$. Pair fit candidates in fitness order.	Deng et al., 2021
5. Crossover $\gamma = 1$	Crossover a fit candidate pair if $rand(0, 1) < \gamma$, $\gamma \in [0, 1]$. X_t^{c*}: For random cut point (i) in parent1. Offspring1 takes the permutation in parent1, parent1[1:i]. Fill indices [i:22] with unvisited nodes in parent2 appearance order. Offspring2 takes the permutation in parent2, parent2[1:i]. Fill indices [i:22] with unvisited nodes in parent1 appearance order. Replace the $\|X_t^{c*}\|$ lowest scoring routes with X_t^{c*}	Deng et al., 2021
6. Mutation rate $\epsilon = 0.02$	Mutate an offspring if $rand(0, 1) < \epsilon \in [0, 1]$. For random indices $i, j \in [1, 22]_{\mathbb{Z}}$, s.t $i \neq j$, swap the ith and jth nodes in the HC.	Deng et al., 2021
7. Generations $gen \leq 490.196$ (490)	The 10,000 max. interaction constraint restricts the generation parameter: $\mathbb{E}(interactions) = [\beta \cdot \gamma \cdot \|X_0\|(1 + \epsilon)] \cdot gen \leq 10,000$	(See Rubric)

Table 3: GA Parameters

3.3 Results

Run	Winning Tour Length (3DP)
1.	97.520
2.	104.351
3.	98.287
4.	104.744
5.	107.166
6.	95.916
7.	118.683
8.	95.499
9.	104.784
10.	115.082
11.	107.198
12.	102.272
13.	94.011
14.	104.675
15.	111.835
16.	110.402
17.	115.633
18.	85.696
19.	107.998
20.	99.816
21.	109.701
22.	105.700
23.	123.115
24.	107.458
25.	109.359
26.	100.417
27.	91.732
28.	102.706
29.	113.204
30.	107.281
Mean	105.074
SD	8.197

Table 4: GA Results

3.3.1 Sample Statistics

Table 4 displays the genetic algorithm results. The sample average length is 105.074 with a standard deviation of 8.197. The minimum winning HC is 85.696, the maximum 123.115. Thus, the winning HC range is 37.419. The sample skewness is 0.603, indicating a positive skew. The sample kurtosis is 3.007, indicating the sample is leptokurtic relative to the normal distribution.

3.3.2 Solution Sensitivity

Parameters initialised according to table 3 produced the results in table 4. When one alters parameters, one changes the HC solutions. This section considers the sensitivity of HC solutions to each parameter:

Initial population X_0: The initial population differs for all 30 runs in table 4. Given the range (37.419), different initial populations produce a range of winning HCs. This relation may be coincidental or causal. Further tests can reveal whether result differences are statistically significant.

Selection β The selection rate (β) was initialised (^{i}) at 0.2. Changing β, one observes:

β	Mean, SD, Skew., Kurt.	Min., Max., Range	Run-time
0.1	105.714, 8.413, -0.119., 2.783.	92.951, 126.700, 33.749	12.6
0.2^i	105.074, 8.198, -0.119, 3.007	85.696, 123.115, 37.419	14.8
0.4	105.574, 8.395, 0.218, 2.142	90.494, 121.495, 31.000	19.7

Table 5: β Sensitivity

The WST p-value for the initial '0.2' sample against the '0.1' and '0.4' samples are 0.910 and 0.861 respectively. This result is notable because differing selection rates did not produce significantly different samples. The sample mean and SD of winning HC lengths is robust to the listed changes in the selection rate. Only the maximum length reduced consistently as the selection rate increased. Though, hypothesis testing is necessary to test the statistical significance of trends in these samples. The run-time increased as the number of crossovers/mutations increased. This change reflects the increasing time complexity of the algorithm.

Crossover rate γ The initial crossover rate (γ) is 1. Reducing γ, one observes:

γ	Mean, SD, Skew., Kurt.	Min., Max., Range	Run-time
0.25	122.878, 9.131, -0.366, 2.660	101.063, 138.135, 37.072	12.7
0.5	112.641, 8.876, 0.137, 2.511	94.170, 130.530, 36.358	12.9
1^i	105.074, 8.198, -0.1187, 3.007	85.696, 123.115, 37.419	14.8

Table 6: γ Sensitivity

The WST p-value for the '1' sample against the '0.5' sample was 0.006, for '1' against '0.25' the p-value was 0.000. These results suggest the alternate samples differed significantly from the initial sample. Again, as the number of crossovers (and thus mutations) increased, so did the run-time. The crossover rate (unlike the selection rate) noticeably improved the HC result. the min., max., and mean HC lengths all reduced by a large proportion of the range. This result is intuitive. Increasing the crossover rate among fit candidates raises competitive pressure. Selecting less fit individuals for crossover weakens competitive pressure.

Mutation Rate The mutation parameter (ϵ) was initialised at 0.2. Changing ϵ, one observes:

ϵ	Mean, SD, Skew., Kurt.	Min., Max., Range	Run-time
0.01	106.617, 6.782, -0.462, 2.861	91.466, 119.331, 27.864	15.3
0.02^i	105.074, 8.198, -0.1187, 3.007	85.696, 123.115, 37.419	14.8
0.1	105.240, 8.209, -0.067, 2.271	88.436, 122.200, 33.763	15.5

Table 7: ϵ Sensitivity

The WST p-value for the '0.02' sample against the '0.01' sample is 0.236. The p-value for the '0.02' sample against the '0.1' sample is 0.926. These samples did not differ significantly. This may be because the mutation rate is low throughout (< 0.1) and thus has a negligible effect on solutions.

Generations The number of generations (gen) was initialised at 490. Changing gen, one observes:

gen	Mean, SD, Skew., Kurt.	Min., Max., Range	Run-time
123	120.392, 8.668, -0.995, 4.237	94.163, 133.261, 39.097	3.8
245	110.211, 7.848, -0.718, 3.375	88.382, 122.580, 34.198	7.5
490^i	105.074, 8.198, -0.1187, 3.007	85.696, 123.115, 37.419	14.8

Table 8: gen Sensitivity

The WST p-value for the '490' sample against the '123' sample is 0.000. The p-value for the '490' sample against the '123' sample is 0.003. Both alternate samples were significantly different from the intial sample.

The run time increases as the number of generations increases. The GA performance increases with more generations; the mean HC lengths and minimum values decreased in each subsequent sample. This is likely due to the increased HC evolution. The algorithm has longer to cultivate short HCs.

All SA and GA sample results arise from random starting permutations and interactions. Thus, constant parameters can generate differing HC samples. The apparent differences displayed above need repeating and hypothesis testing before one can distinguish notable trends from statistical noise.

4 SA-GA Comparison

Statistic	Value	f_W (3DP)
W	0	0.000
$\alpha(5\%)$	137	0.050

Table 9: Wilcoxon Signed Rank Test (WST)

4.1 Performance: Rank Difference

Table 9 shows the results for the WST (one-sided). A significant difference exists between the SA and GA samples. The SA algorithm produces a shorter route with higher frequency, thus $x_{SA}^{(i)} - x_{GA}^{(i)}$ is negative in all 30 runs. The WST gives insufficient evidence to accept H_0, suggesting

$$w = min\{|sum(R_{-}^{(i)})|, |sum(R_{+}^{(i)})|\} < 0$$

The comparatively longer routes in the GA arise because of the exploration-exploitation trade-off. The GA algorithm causes large perturbations to each permutation during crossover, thus exploring the solution space. Yet, these large changes can discard efficient sub-sections of a permutation. As an efficient route forms, it is selected and randomly disturbed. A population tends towards more efficient routes but the GA generated only 490 generations, so this aggregate behaviour did not converge to globally optimal routes. The GA did not prioritise exploitation.

The SA prioritised exploitation. The SA algorithm conducted 10,000 iterations of a single permutation. Further, perturbations were accepted based on their efficiency. Thus, the SA took a more deliberate approach to optimising the HC. The two algorithms may generate similar solution if the GA generations increase, or the routes are better preserved in each crossover. For example, one could:

1. Increase the number of generations (e.g. from 490 to 10,000).
2. Alter the crossover rule (e.g. preserve node ordering in permutation-sections when mixing two parents).

4.2 Performance: Variation

The sample SD is similar for SA and GA (8.295, 8.197). The large GA population covers a larger subset of the solution space at each iteration, yet the crossover rate is high and the mutation rate low. Given, the number of generations, the algorithm may converge to a local optimum HC neighbourhood. The GA algorithm could resemble colonial Sino-European freight travelling via the silk road or around the cape of Africa until the creation of the Suez canal. The algorithm may exploit locally optimum HCs, without exploring the optimal SA HCs if SA HCs are a small subset of the solution space.

References

Černý, Vladimır (1985). "Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm". In: *Journal of optimization theory and applications* 45, pp. 41–51.

Deng, Yanlan, Xiong, Juxia, and Wang, Qiuhong (2021). "A hybrid cellular genetic algorithm for the traveling salesman problem". In: *Mathematical Problems in Engineering* 2021, pp. 1–16.

Diestel, Reinhard (2017). *Graph Theory*. 5th. Berlin: Springer. ISBN: 978-3-662-53621-6.

FrontlineSolvers, R (2018). *Genetic Algorithms and Evolutionary Algorithms-Introduction*.

Goddard, Wayne (2009). *Discrete Math for Computing*. Second Edition. Sudbury, MA: Jones and Bartlett Publishers. ISBN: 9780763741495.

Kirkpatrick, Scott, Gelatt Jr, C Daniel, and Vecchi, Mario P (1983). "Optimization by simulated annealing". In: *science* 220.4598, pp. 671–680.

Rutenbar, Rob A (1989). "Simulated annealing algorithms: An overview". In: *IEEE Circuits and Devices magazine* 5.1, pp. 19–26.

Sussex, University of (2005). *Table of critical values for the Wilcoxon test*. URL: http://users.sussex.ac.uk/~grahamh/RM1web/WilcoxonTable2005.pdf.

Yu, Xinjie and Gen, Mitsuo (2010). *Introduction to evolutionary algorithms*. Springer Science & Business Media.

5 Appendices

5.1 Wilcoxon-signed Rank Test

The Wilcoxon signed-rank test (WST) is a non-parametric hypothesis test, with paired samples (s_1,s_2) of size n, the null and alternative hypotheses are:

1. H_0: median difference score $m_{sample} = 0$

2. H_1: median difference score $m_{sample} > 0$

5.1.1 Test Statistic

To calculate the test statistic:

1. Take the pairwise difference $x_1^{(i)} - x_2^{(i)}$

2. Order the differences based on their absolute value $|x_1^{(i)} - x_2^{(i)}|$

3. Rank the ordered differences 1 to n, to find $R^{(i)}$

4. Apply $\text{sgn}(x_1^{(i)} - x_2^{(i)})$ to $R^{(i)}$

5. Test statistic $w = min\{|sum(R_-^{(i)})|, |sum(R_+^{(i)})|\}$

5.1.2 Result

Compare the WST test statistic to its probability density function $f_W(w) = P(W \leq w)$ to determine the significance of W (Sussex, 2005). For a pre-defined threshold α:

$$Result = \begin{cases} Accept H_0 & \text{if } f_w(w) > f_W(\alpha) \\ Reject H_0 & otherwise \end{cases} \tag{1}$$