INFO 6205 Spring 2023 Project

Traveling Salesman

Team Details:

Member 1**: Dinesh Shekhawat (NUID - 002776484)**
Member 2**: Dhruv Patel (NUID - 002928881)**
Member 3**: Harshil Shah (NUID - 002780887)**
GitHub Repo - https://github.com/pateldhruvr/INFO6205-Final-Project/

Aim

Given a list of geographical points represented as latitudes & longitudes, generate the shortest possible route that visits each point exactly once and returns to the point of origin. Use Christofides to generate a candidate solution and apply 4 optimization strategies - two tactical and two strategic techniques to get the shortest possible tour.

Approach

1. Get the lower bound estimate using MST Cost and further get more improved lower bound using max One-tree cost
2. Build a candidate tour using Christofides Algorithm
3. Use candidate tour obtained from Christofides and apply the following optimization strategies:-
    a. 2-opt. Heuristic (Tactical)- Randomly swap two nodes for a specified time interval. If an improvement is found in the tour cost, update the tour and continue. Once the time is exhausted, switch the algorithm by looking for improvements by swapping two adjacent nodes in the tour. Update the tour if an improvement is found in the tour distance.

b. 3-opt. Heuristic (Tactical)- Same as 2-opt, but check all the combinations obtained from the selected three nodes and choose the one that gives the least tour distance during each swap.
c. Ant Colony Optimization (Strategic)- Simulate the behavior of ants by leaving pheromone trails over short paths, thus biasing ant movement towards areas of the problem space that have high pheromone concentrations, leading to a more optimized tour.
d. Simulated Annealing (Strategic)- Explore the solution space that gets ignored due to the local minimum trap using 2opt/3opt, which may lead to a more optimized solution
e. Mix and Match (Strategic + Tactical)- Apply different optimization techniques on top of each other to get the best possible shortest tour
    i. E.g., Applying 3-opt. Heuristic on top of the Ant Colony Optimization

---

Program

1. Christofides
    a. Data Structures
        i. Heap (Min Indexed Priority Queue)[2]
        ii. Stack
        iii. HashMap
        iv. LinkedList
        v. Graph
    b. Classes
        i. HaversineDistanceUtil {}
        ii. Point(String id, double latitude, double longitude) {}
        iii. Edge(Point from, Point to) {}
        iv. PrimsMST(List<Point> nodes) {}[3]
        v. OneTree(int n) {}
        vi. KolmogorovWeightedPerfectMatchingImpl implements PerfectMatchingSolverService {}[4][5]
        vii. FluerysAlgorithm(int numOfVertices){}[6]
        viii. Christofides(List<Point> points) {}
    c. Algorithm[1]
        i. Create a minimum spanning tree T of G

ii. Let O be the set of vertices with odd degree in T. By the handshaking lemma, O has an even number of vertices.
iii. Find a minimum-weight perfect matching M in the induced subgraph given by the vertices from O
iv. Combine the edges of M and T to form a connected multigraph H in which each vertex has even degree
v. Form an Eulerian circuit in H
vi. Make the circuit found in previous step into a Hamiltonian circuit by skipping repeated vertices (shortcutting)

d. Invariants
   i. Complete graph - All the vertices in the graph should be connected to each other
   ii. The triangle inequality - the distance between any two nodes is always less than or equal to the sum of the distances between those nodes and any other node
   iii. The graph must have an even number of odd-degree nodes, or else a perfect matching cannot be found

2. Ant Colony Optimization

a. Data Structures
   i. 2D Array
b. Classes
   i. AntColonyOptimization(
      double[][] graph,
      int[] christofidesTour,
      int numberOfAnts,
      double phermoneExponent,
      double heuristicExponent,
      double phermoneEvaporationRate,
      double phermoneDepositFactor,
      int numberOfIterations,
      int maxImprovementIterations) {}
c. Algorithm
   i. Initialize the parameters:
      - N: number of ants
      - α: trail factor
      - β: heuristic factor

- ρ: evaporation rate
- Q: pheromone deposit factor
- t_0: initial pheromone trail
- tau: pheromone trail matrix
- h: heuristic information matrix
- s: current solution (Christofides's solution)

ii. Initialize the pheromone trail matrix tau and the heuristic information matrix h.

iii. Repeat for each iteration:
- Generate solutions using the following steps:
  - Place each ant at a random starting point
  - Move the ant to the next city based on the probabilities calculated using the pheromone trail matrix and the heuristic information matrix
  - Update the pheromone trail matrix using the Q value and the length of the tour
- Update the pheromone trail matrix using the following steps:
  - Evaporate the pheromone trail using the evaporation rate
  - Add the pheromone deposit to the edges visited by the ants
- Update the best solution found so far
- Terminate if the stopping criterion is met

iv. Return the best solution found.

d. Invariants

i. Parameter settings: ACO has several parameters that need to be set before the algorithm can be run. These include the number of ants, the pheromone update rate, and the importance of the heuristic information.

ii. Pheromone trail: ACO uses a pheromone trail to communicate information between the ants. The pheromone trail is updated by the ants based on the quality of the solutions they find.

iii. Ant behavior: The behavior of the ants is a key aspect of ACO. Each ant follows a set of rules to decide which path to take in the search space. These rules are often based on the pheromone trail and the heuristic information.

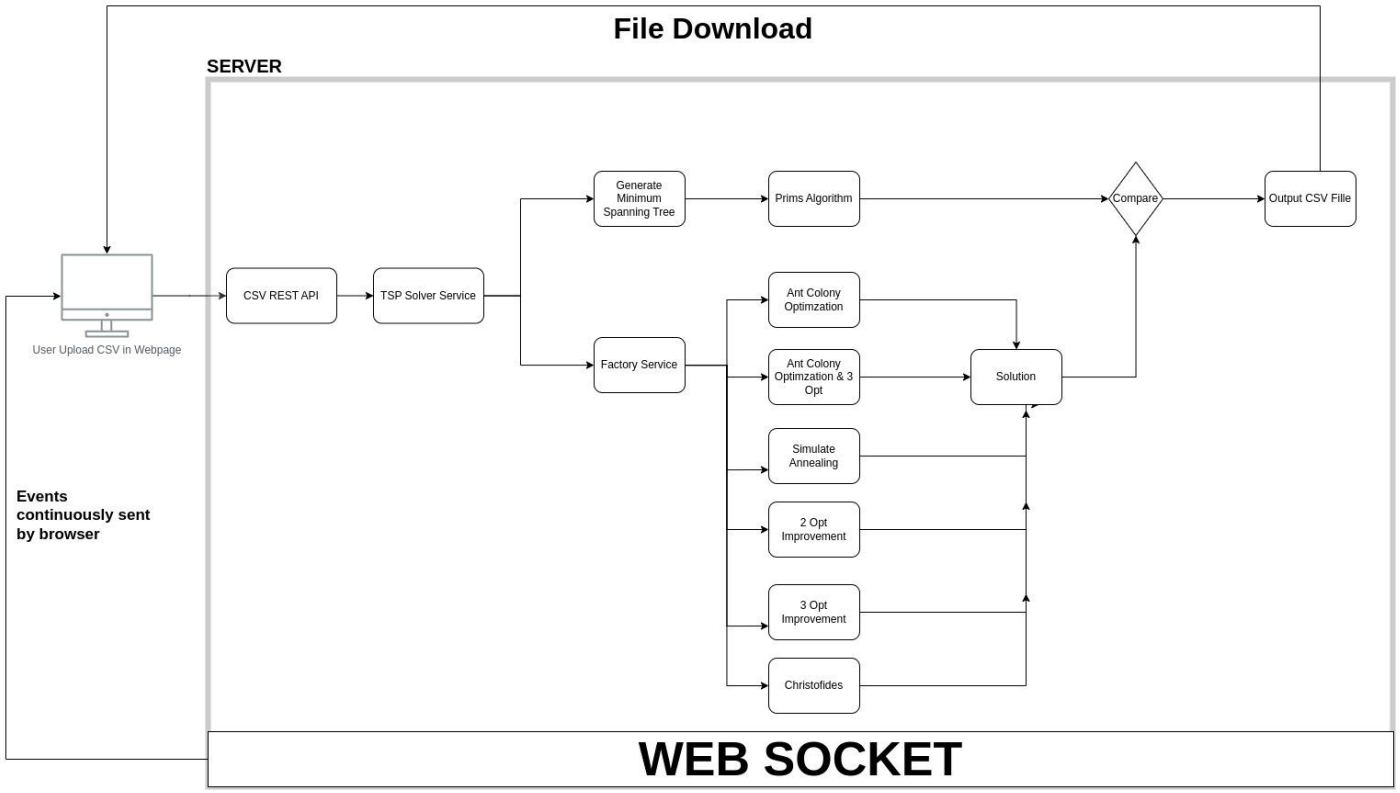
3. Simulated Annealing

a. Data Structures
   i. 2D Array
b. Classes
   i. SimulatedAnnealingOptimization(
      int[] christofidesTour,
      double[][] distanceMatrix,
      int maxIteration,
      double startingTemperature,
      double finalTemperature,
      double coolingRate) {}

c. Algorithm[7]
   i. current ← problem.INITIAL-STATE
   ii. for t = 1 to MAX_ITERATIONS do
      - T ← schedule(t)
      - if T = 1 then return current
      - next ← a randomly selected successor of current
      - $\Delta E \leftarrow$ VALUE(next) - VALUE(current)
      - if $\Delta E > 0$ then current ← next
      - else current ← next only with probability $e^{\Delta E/T}$

d. Invariants
   i. Temperature Schedule: The temperature schedule used in the Simulated Annealing algorithm must satisfy certain properties, such as decreasing over time and converging to zero. The schedule must be carefully chosen to balance exploration and exploitation of the search space.
   ii. Probability Distribution: The probability distribution used to accept or reject candidate solutions must be valid, meaning that it must sum to 1 and be non-negative for all inputs.

Flowchart (inc. UI Flow)

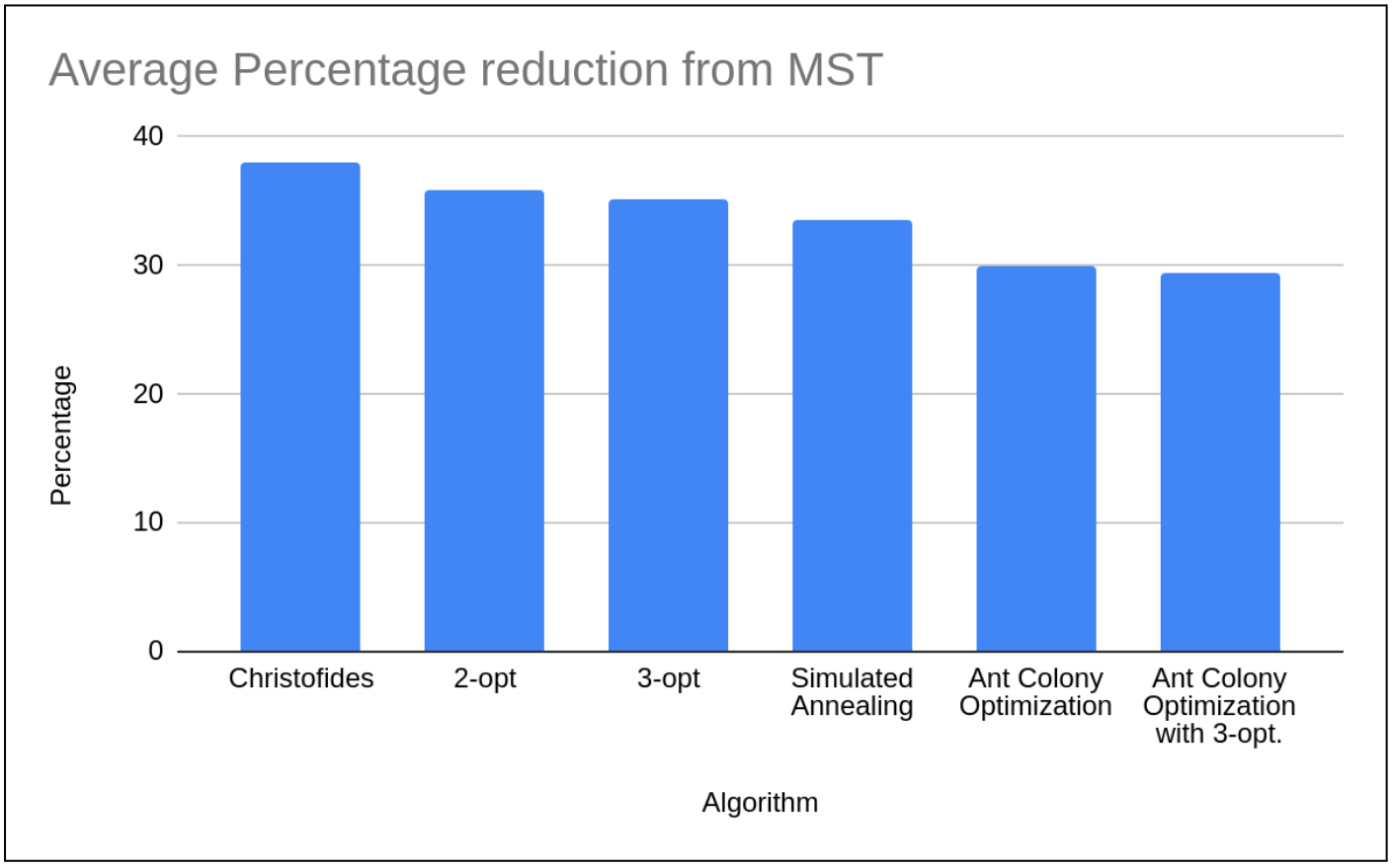


Observations & Graphical Analysis

- ## Graphical Analysis

Algorithm	Nodes	Cost	Percentage	Runtime (ms)
Christofides	10	51466.569	53.939	4
	50	28622892.3	46.418	10
	156	384244.499	29.944	15
	245	195454570.1	32.232	54
	585	656222.601	27.837	75
2-opt	10	50650.536	51.498	111
	50	27411308	40.22	337
	156	379004.883	28.172	1610
	245	194727939.6	31.74	1496
	585	655799.379	27.755	6143
3-opt	10	50650.536	51.498	188
	50	27242219.72	39.355	737

	156	377587.237	27.692	3821
	245	192086629.5	29.954	3558
	585	654442.295	27.491	14143
Simulated Annealing	10	50650.536	51.498	567
	50	27411308	40.22	811
	156	368291.923	24.549	1424
	245	188951041.8	27.832	2362
	585	634882.739	23.68	6349
Ant Colony Optimization	10	50650.536	51.498	14
	50	26566055.19	35.897	73
	156	356322.389	20.501	422
	245	181349274.9	22.689	1050
	585	611917.589	19.206	7614
Ant Colony Optimization with 3-opt.	10	50650.536	51.498	942
	50	26362601.77	34.856	3908
	156	353942.702	19.696	12418
	245	180878561	22.371	19644
	585	609944.678	18.822	54009



- **Observations**

1. Minimum Spanning Tree: Prim's Eager Evaluation was selected as the lazy version and Kruskal's ran out of space for large number of input
2. Christofides: Minimum Weighted Perfect Matching had the major impact on the solution that Christofides gave. If Nearest Neighbour was selected instead of Edmond's Blossom V for perfect matching, the tour cost would go from ~660000 to ~840000 for the dataset 'info6205.spring2023.teamproject.csv'
3. Optimizations: Although Simulated Annealing explored the solution space which was ignored by the local minimum, the results produced were not as optimized as compared to a simple 2opt. / 3opt. Randomly swapping nodes instead of adjacent gave more good results in 2opt./3opt. Therefore, a strategy was created in the project which used the best of both which swapped the nodes randomly first and then check for improvements in adjacent nodes
4. Best Results: Ant Colony Optimization performed the best compared to rest. However, applying a 3opt. on top of ACO gave even better results.

Result and Mathematical Analysis

[Note: The results are based on the final dataset given which consists of 585 points and the distance metric is meters]

1. Christofides:-
    a. Eager Prim's Minimum Spanning tree
        - Cost: 513326.0953990727
        - Mathematical Analysis: O(m log n), where m = number of edges and n = vertices in the graph
    b. Weighted Minimum Perfect matching using Edmond's Kolmogrov V
        - Number of vertices with odd degree provided: 232
        - Cost: 186659.84865451863
        - Mathematical Analysis: $O(n^2 \log n)$[5], n = number of vertices
    c. Eulerian Tour
        - Mathematical Analysis: $O(m^2 + mn)$, where m = number of edges and n = number of vertices
    d. Hamilton Tour
        - Cost: 656222.6012803589

- Mathematical Analysis: O(m), where m = number of edge in the Eulerian Tour

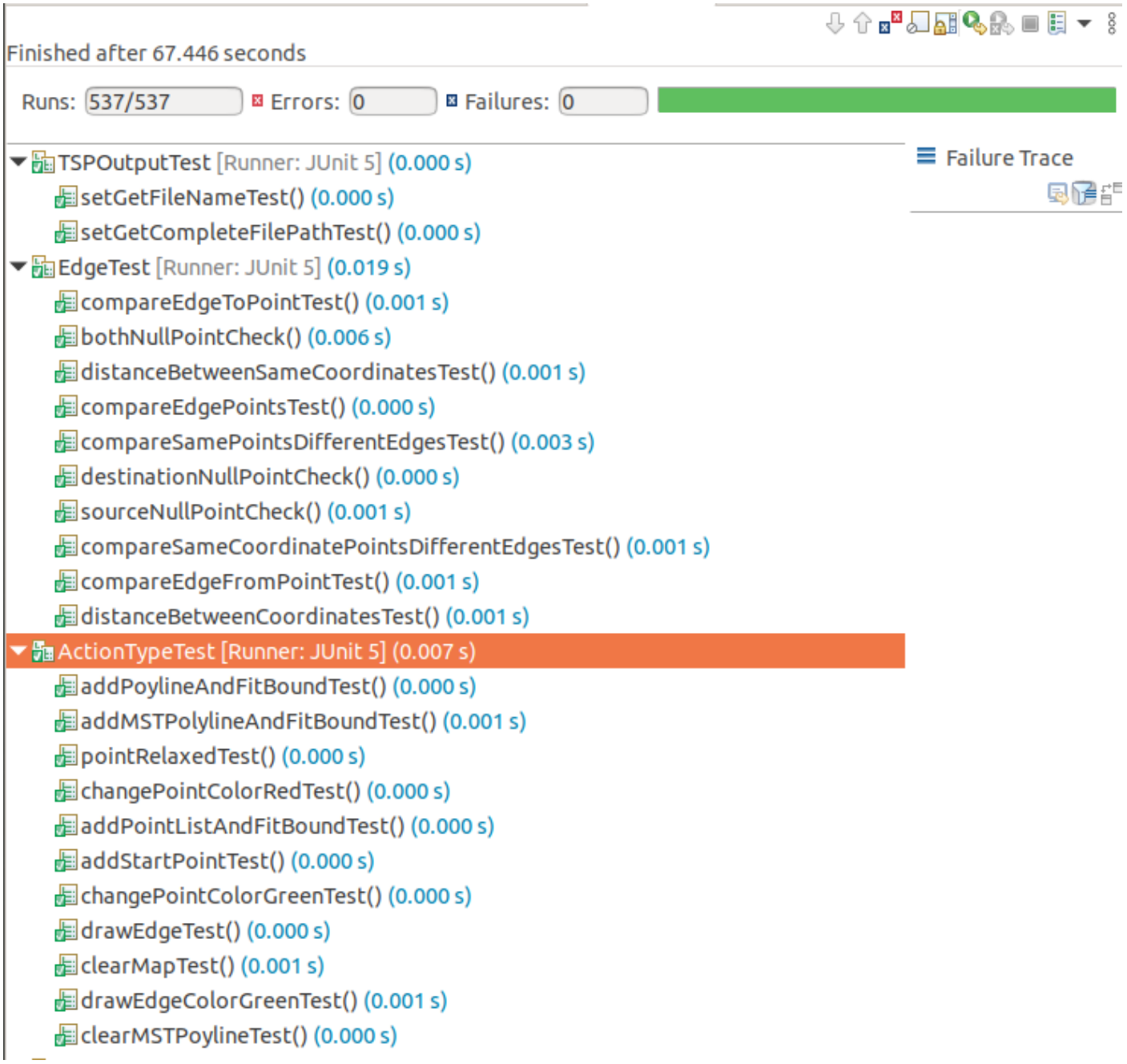
e. Final Result of Christofides
   - Cost: 656222.6012803589
   - Percentage (compared to MST) : 27.83737416859644
   - Mathematical Analysis: $O(n^2 \log n)$, where n is the number of vertices in the graph.

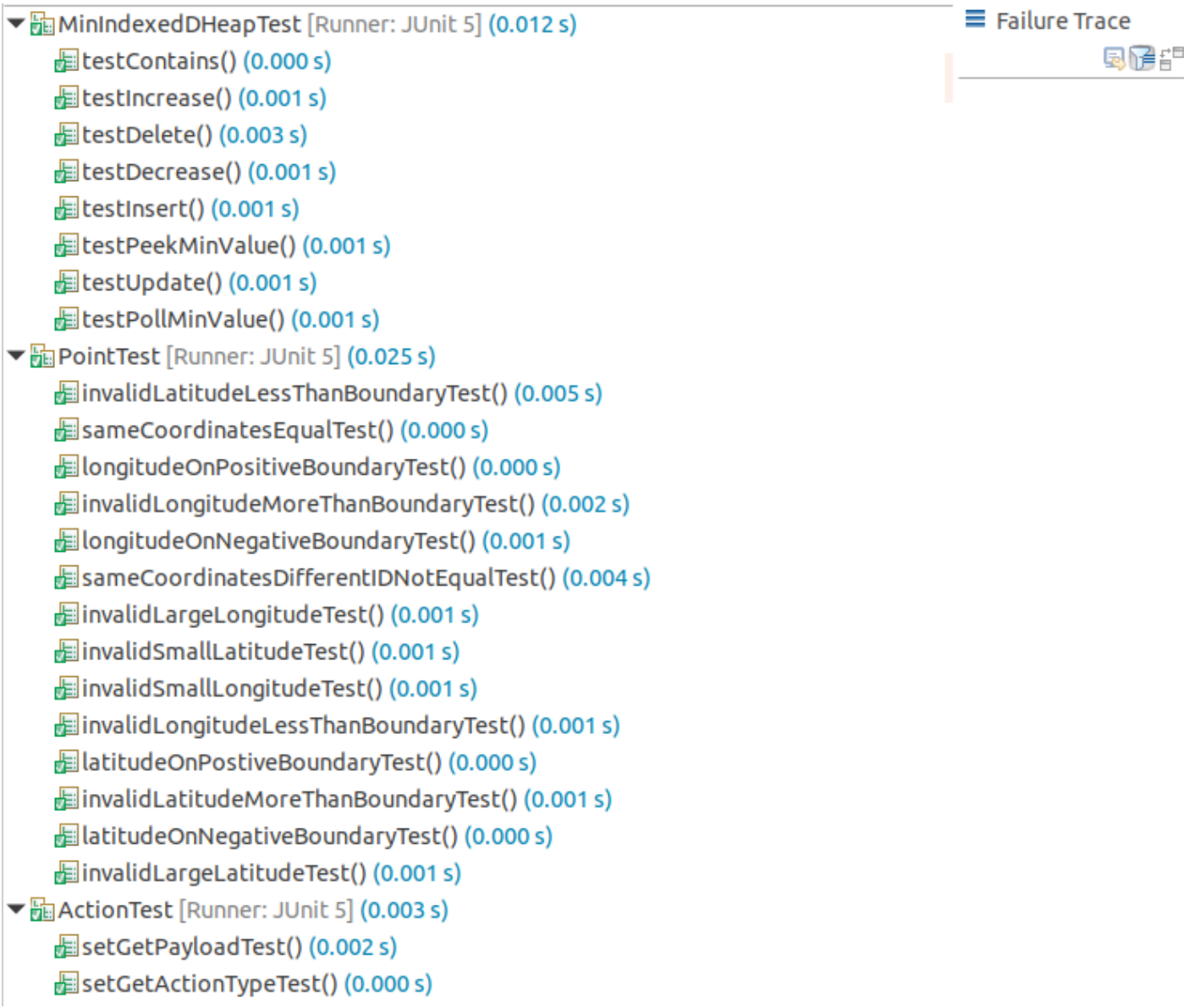
2. Ant Colony Optimization
   - Parameters: number of Ants=10, pheromone exponent=1.0, heuristic exponent=2.0, pheromone evaporationRate=0.1, pheromone deposit factor=1.0, number of iterations=20, max improvement iterations=1000
   - Cost (Range based because of the random factor): ~610687.476 to ~627603.3040357905
   - Percentage (compared to MST) : ~18.967 to ~22.26210778313848
   - Mathematical Analysis: ~O(iter * ant * (n + m) * h), where iter is the number of iterations, ant is the number of ants, h is the complexity of the heuristic function, and (n + m) is the size of the problem instance.
3. Simulated Annealing
   - Parameters: max iteration: 1000000, starting temperature: 1000.0, final temperature: 1.0, cooling rate: 0.9995
   - Cost (Range based because of the random factor): ~635281.2267126825 to ~635975.710984002
   - Percentage (compared to MST) : ~23.757828095374506 to ~23.893119146724537
   - Mathematical Analysis: $O(kn^2)$, where k is the number of iterations or cooling steps and n is the number of points
4. 2opt. (Selecting the strategy that gave the best answer)
   - Strategy: 3 (Randomly swap points and check for improvements for a particular time interval and then swap adjacent points until improvements are found)
   - Parameters: strategy: 3, budget: 1000000
   - Cost (approximate because of the random factor): ~635601.3673416032
   - Percentage (compared to MST) : ~23.820194032308173

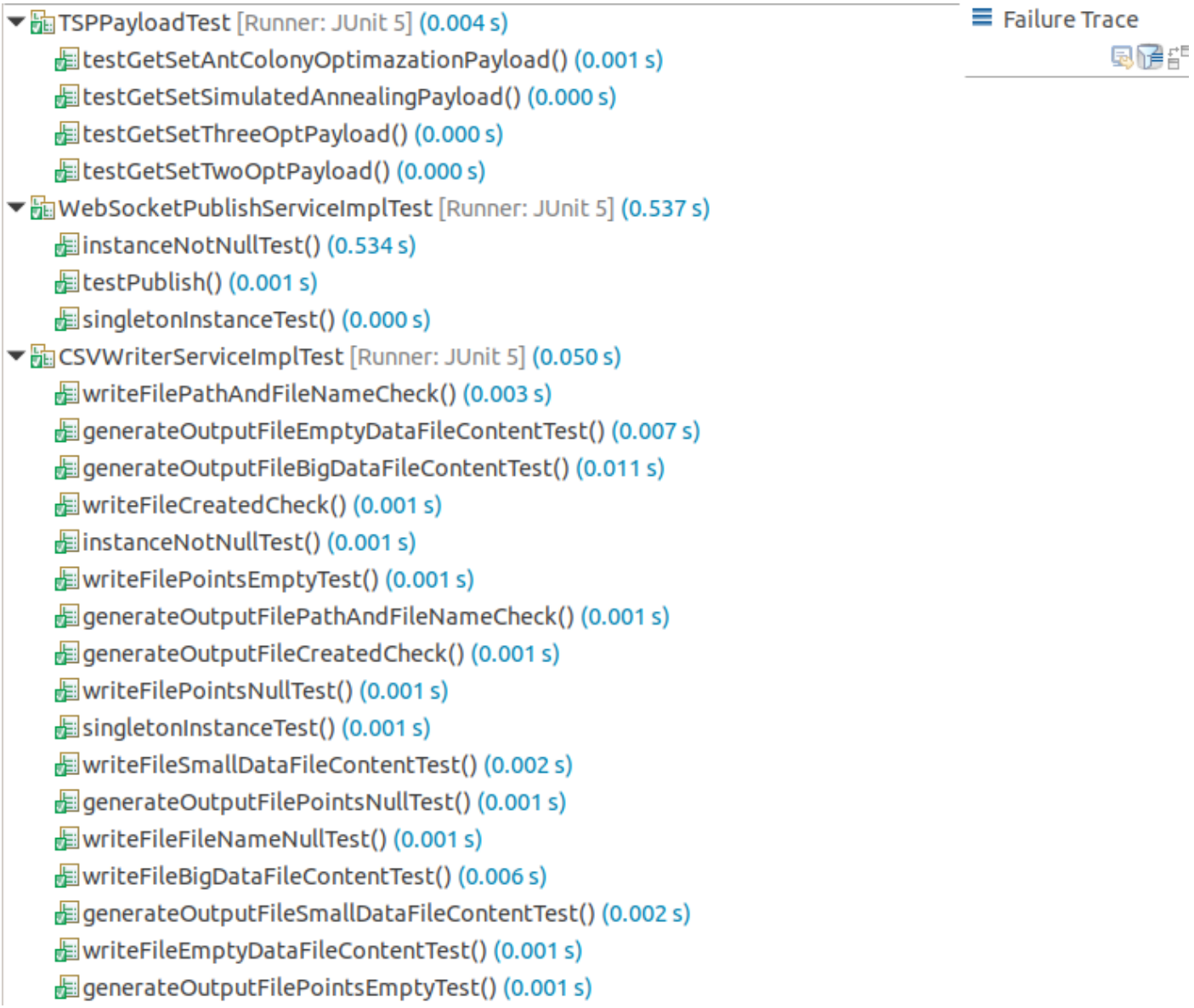
- Mathematical Analysis: O(budget + n^2), where budget is the number of iterations and n is the number of cities

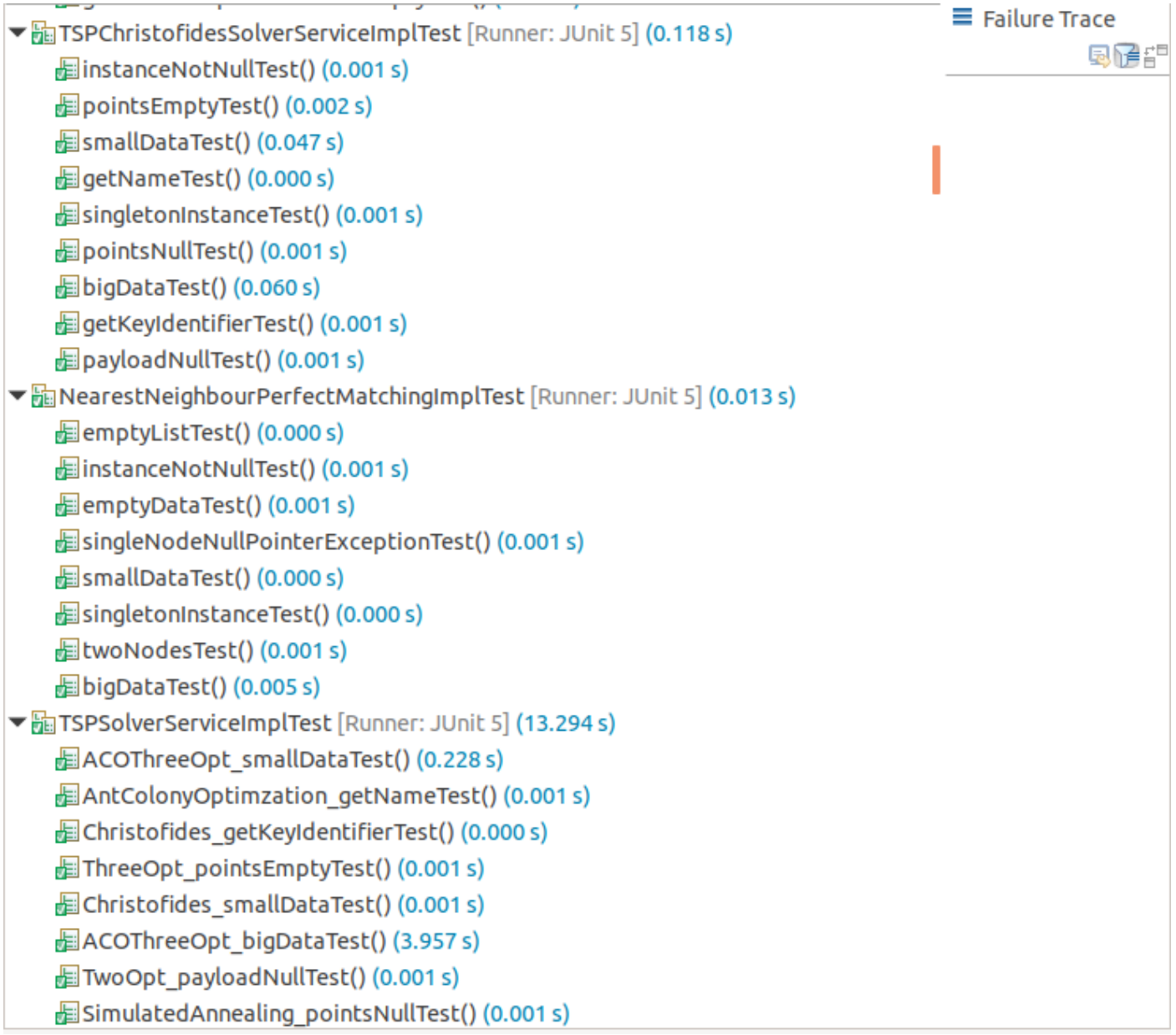
5. 3opt. (Selecting the strategy that gave the best answer)
    - Strategy: 3 (Randomly swap points and check for improvements for a particular time interval and then swap adjacent points until improvements are found)
    - Parameters: strategy: 3, budget: 1000000
    - Cost (approximate based because of the random factor): ~ ~634983.5210839442
    - Percentage (compared to MST) : ~23.69983267464554
    - Mathematical Analysis: O(budget + n^3), where budget is the number of iterations and n is the number of cities
6. 3opt. on top of ACO (currently gives the **best distance**)
    - Parameters: strategy: 3, budget: 5000000, number of Ants=10, pheromone exponent=1.0, heuristic exponent=2.0, pheromone evaporationRate=0.1, pheromone deposit factor=1.0, number of iterations=20, max improvement iterations=1000
    - Cost: **609944.6779660139**
    - Percentage (compared to TSP) : **18.822067187491704**

Unit Tests


Finished after 67.446 seconds
Runs: 537/537
Errors: 0
Failures: 0
TSPOutputTest [Runner: JUnit 5] (0.000 s)
setGetFileNameTest() (0.000 s)
setGetCompleteFilePathTest() (0.000 s)
EdgeTest [Runner: JUnit 5] (0.019 s)
compareEdgeToPointTest() (0.001 s)
bothNullPointCheck() (0.006 s)
distanceBetweenSameCoordinatesTest() (0.001 s)
compareEdgePointsTest() (0.000 s)
compareSamePointsDifferentEdgesTest() (0.003 s)
destinationNullPointCheck() (0.000 s)
sourceNullPointCheck() (0.001 s)
compareSameCoordinatePointsDifferentEdgesTest() (0.001 s)
compareEdgeFromPointTest() (0.001 s)
distanceBetweenCoordinatesTest() (0.001 s)
ActionTypeTest [Runner: JUnit 5] (0.007 s)
addPoylineAndFitBoundTest() (0.000 s)
addMSTPolylineAndFitBoundTest() (0.001 s)
pointRelaxedTest() (0.000 s)
changePointColorRedTest() (0.000 s)
addPointListAndFitBoundTest() (0.000 s)
addStartPointTest() (0.000 s)
changePointColorGreenTest() (0.000 s)
drawEdgeTest() (0.000 s)
clearMapTest() (0.001 s)
drawEdgeColorGreenTest() (0.001 s)
clearMSTPoylineTest() (0.000 s)
Failure Trace


MinIndexedDHeapTest [Runner: JUnit 5] (0.012 s)
testContains() (0.000 s)
testIncrease() (0.001 s)
testDelete() (0.003 s)
testDecrease() (0.001 s)
testInsert() (0.001 s)
testPeekMinValue() (0.001 s)
testUpdate() (0.001 s)
testPollMinValue() (0.001 s)
PointTest [Runner: JUnit 5] (0.025 s)
invalidLatitudeLessThanBoundaryTest() (0.005 s)
sameCoordinatesEqualTest() (0.000 s)
longitudeOnPositiveBoundaryTest() (0.000 s)
invalidLongitudeMoreThanBoundaryTest() (0.002 s)
longitudeOnNegativeBoundaryTest() (0.001 s)
sameCoordinatesDifferentIDNotEqualTest() (0.004 s)
invalidLargeLongitudeTest() (0.001 s)
invalidSmallLatitudeTest() (0.001 s)
invalidSmallLongitudeTest() (0.001 s)
invalidLongitudeLessThanBoundaryTest() (0.001 s)
latitudeOnPostiveBoundaryTest() (0.000 s)
invalidLatitudeMoreThanBoundaryTest() (0.001 s)
latitudeOnNegativeBoundaryTest() (0.000 s)
invalidLargeLatitudeTest() (0.001 s)
ActionTest [Runner: JUnit 5] (0.003 s)
setGetPayloadTest() (0.002 s)
setGetActionTypeTest() (0.000 s)
Failure Trace


TSPPayloadTest [Runner: JUnit 5] (0.004 s)
testGetSetAntColonyOptimazationPayload() (0.001 s)
testGetSetSimulatedAnnealingPayload() (0.000 s)
testGetSetThreeOptPayload() (0.000 s)
testGetSetTwoOptPayload() (0.000 s)
WebSocketPublishServiceImplTest [Runner: JUnit 5] (0.537 s)
instanceNotNullTest() (0.534 s)
testPublish() (0.001 s)
singletonInstanceTest() (0.000 s)
CSVWriterServiceImplTest [Runner: JUnit 5] (0.050 s)
writeFilePathAndFileNameCheck() (0.003 s)
generateOutputFileEmptyDataFileContentTest() (0.007 s)
generateOutputFileBigDataFileContentTest() (0.011 s)
writeFileCreatedCheck() (0.001 s)
instanceNotNullTest() (0.001 s)
writeFilePointsEmptyTest() (0.001 s)
generateOutputFilePathAndFileNameCheck() (0.001 s)
generateOutputFileCreatedCheck() (0.001 s)
writeFilePointsNullTest() (0.001 s)
singletonInstanceTest() (0.001 s)
writeFileSmallDataFileContentTest() (0.002 s)
generateOutputFilePointsNullTest() (0.001 s)
writeFileFileNameNullTest() (0.001 s)
writeFileBigDataFileContentTest() (0.006 s)
generateOutputFileSmallDataFileContentTest() (0.002 s)
writeFileEmptyDataFileContentTest() (0.001 s)
generateOutputFilePointsEmptyTest() (0.001 s)
Failure Trace


Failure Trace
TSPChristofidesSolverServiceImplTest [Runner: JUnit 5] (0.118 s)
instanceNotNullTest() (0.001 s)
pointsEmptyTest() (0.002 s)
smallDataTest() (0.047 s)
getNameTest() (0.000 s)
singletonInstanceTest() (0.001 s)
pointsNullTest() (0.001 s)
bigDataTest() (0.060 s)
getKeyIdentifierTest() (0.001 s)
payloadNullTest() (0.001 s)
NearestNeighbourPerfectMatchingImplTest [Runner: JUnit 5] (0.013 s)
emptyListTest() (0.000 s)
instanceNotNullTest() (0.001 s)
emptyDataTest() (0.001 s)
singleNodeNullPointerExceptionTest() (0.001 s)
smallDataTest() (0.000 s)
singletonInstanceTest() (0.000 s)
twoNodesTest() (0.001 s)
bigDataTest() (0.005 s)
TSPSolverServiceImplTest [Runner: JUnit 5] (13.294 s)
ACOThreeOpt_smallDataTest() (0.228 s)
AntColonyOptimzation_getNameTest() (0.001 s)
Christofides_getKeyIdentifierTest() (0.000 s)
ThreeOpt_pointsEmptyTest() (0.001 s)
Christofides_smallDataTest() (0.001 s)
ACOThreeOpt_bigDataTest() (3.957 s)
TwoOpt_payloadNullTest() (0.001 s)
SimulatedAnnealing_pointsNullTest() (0.001 s)

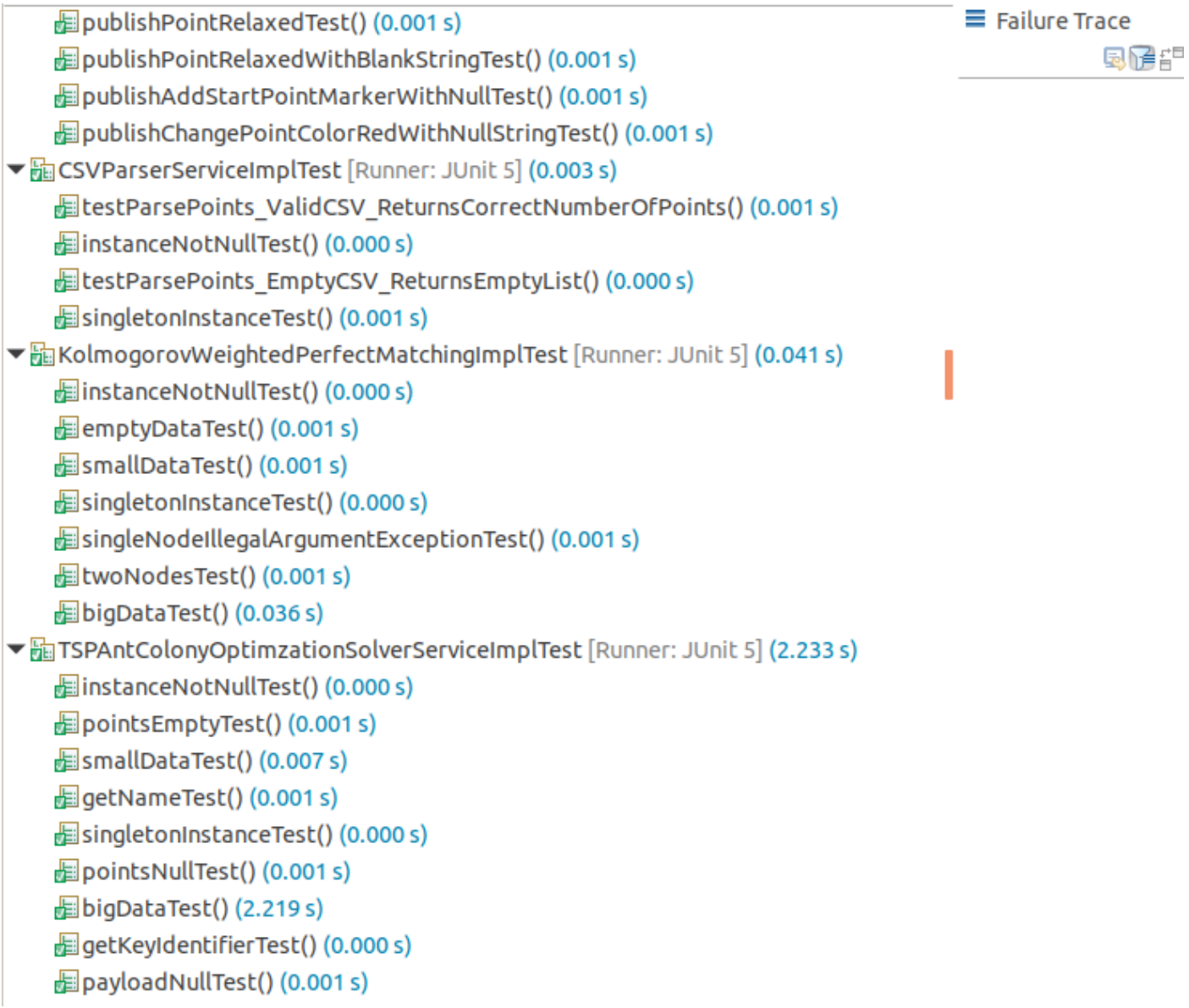
ThreeOpt_smallDataStrategy1Test() (0.128 s)
ThreeOpt_smallDataStrategy2Test() (0.001 s)
ThreeOpt_smallDataStrategy3Test() (0.115 s)
ThreeOpt_smallDataStrategy4Test() (0.001 s)
AntColonyOptimzation_payloadNullTest() (0.001 s)
ACOThreeOpt_pointsNullTest() (0.002 s)
ThreeOpt_bigDataTestStrategy1Test() (1.547 s)
ThreeOpt_bigDataTestStrategy2Test() (0.211 s)
ThreeOpt_bigDataTestStrategy3Test() (1.747 s)
ThreeOpt_bigDataTestStrategy4Test() (0.067 s)
Christofides_instanceNotNullTest() (0.000 s)
TwoOpt_pointsNullTest() (0.001 s)
SimulatedAnnealing_getNameTest() (0.000 s)
Christofides_bigDataTest() (0.017 s)
SimulatedAnnealing_smallDataTest() (0.427 s)
ThreeOpt_instanceNotNullTest() (0.000 s)
Christofides_singletonInstanceTest() (0.000 s)
Christofides_payloadNullTest() (0.001 s)
AntColonyOptimzation_getKeyIdentifierTest() (0.000 s)
ThreeOpt_pointsNullTest() (0.001 s)
TwoOpt_smallDataStrategy1Test() (0.054 s)
TwoOpt_smallDataStrategy2Test() (0.001 s)
TwoOpt_smallDataStrategy3Test() (0.054 s)
TwoOpt_pointsEmptyTest() (0.001 s)
AntColonyOptimzation_singletonInstanceTest() (0.000 s)
AntColonyOptimzation_bigDataTest() (2.177 s)
Christofides_pointsNullTest() (0.001 s)
AntColonyOptimzation_pointsEmptyTest() (0.001 s)

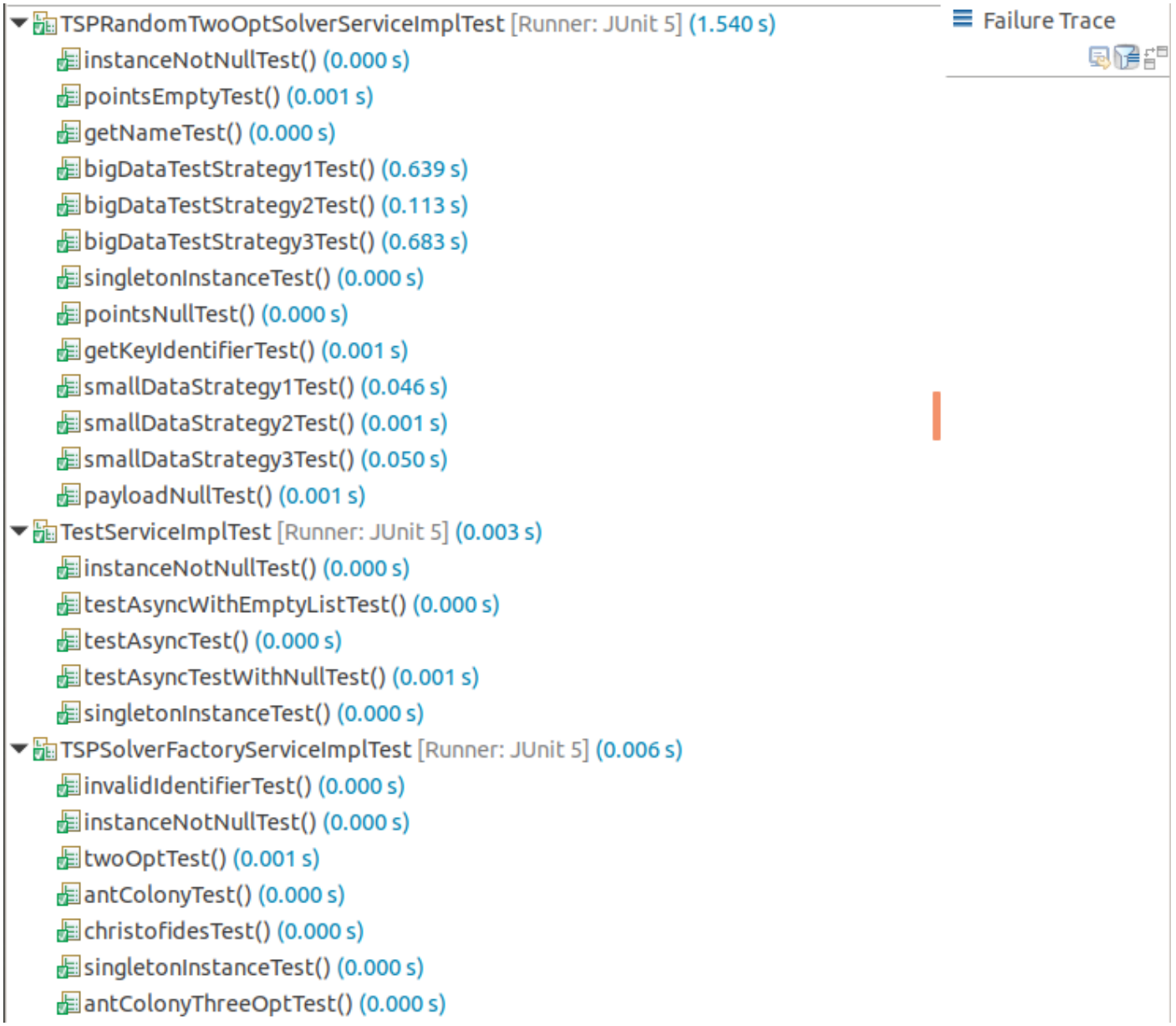
Failure Trace

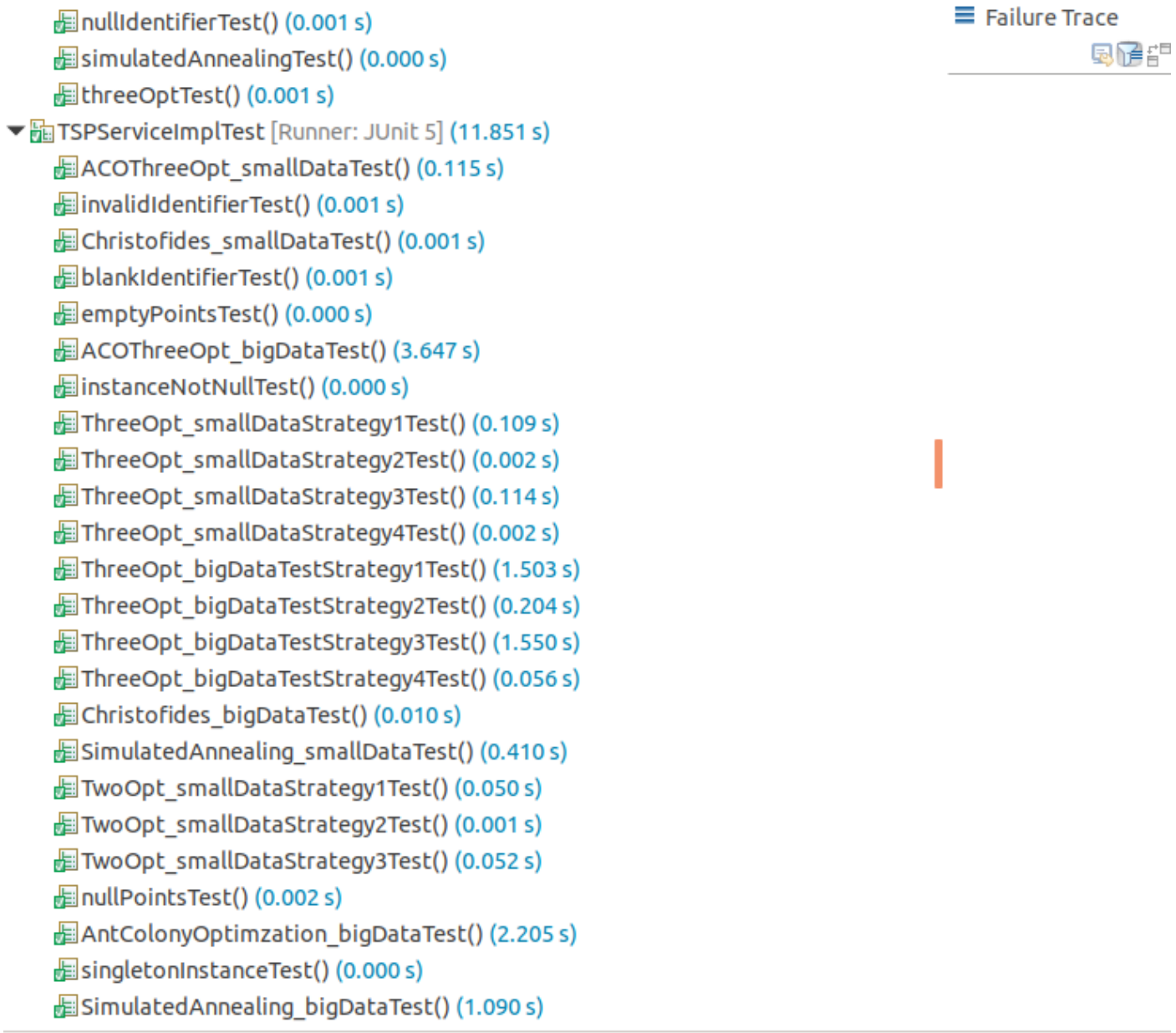
SimulatedAnnealing_singletonInstanceTest() (0.000 s)
SimulatedAnnealing_getKeyIdentifierTest() (0.000 s)
ACOThreeOpt_instanceNotNullTest() (0.001 s)
ThreeOpt_getKeyIdentifierTest() (0.001 s)
AntColonyOptimzation_instanceNotNullTest() (0.001 s)
Christofides_pointsEmptyTest() (0.001 s)
SimulatedAnnealing_payloadNullTest() (0.001 s)
ACOThreeOpt_payloadNullTest() (0.001 s)
SimulatedAnnealing_bigDataTest() (1.071 s)
TwoOpt_getNameTest() (0.001 s)
TwoOpt_singletonInstanceTest() (0.001 s)
AntColonyOptimzation_smallDataTest() (0.009 s)
TwoOpt_getKeyIdentifierTest() (0.000 s)
ACOThreeOpt_getNameTest() (0.001 s)
AntColonyOptimzation_pointsNullTest() (0.001 s)
ACOThreeOpt_getKeyIdentifierTest() (0.000 s)
ThreeOpt_getNameTest() (0.000 s)
ThreeOpt_payloadNullTest() (0.000 s)
TwoOpt_bigDataTestStrategy1Test() (0.660 s)
TwoOpt_bigDataTestStrategy2Test() (0.084 s)
TwoOpt_bigDataTestStrategy3Test() (0.684 s)
ThreeOpt_singletonInstanceTest() (0.000 s)
SimulatedAnnealing_pointsEmptyTest() (0.001 s)
ACOThreeOpt_pointsEmptyTest() (0.001 s)
ACOThreeOpt_singletonInstanceTest() (0.000 s)
TwoOpt_instanceNotNullTest() (0.000 s)
Christofides_getNameTest() (0.000 s)
SimulatedAnnealing_instanceNotNullTest() (0.000 s)

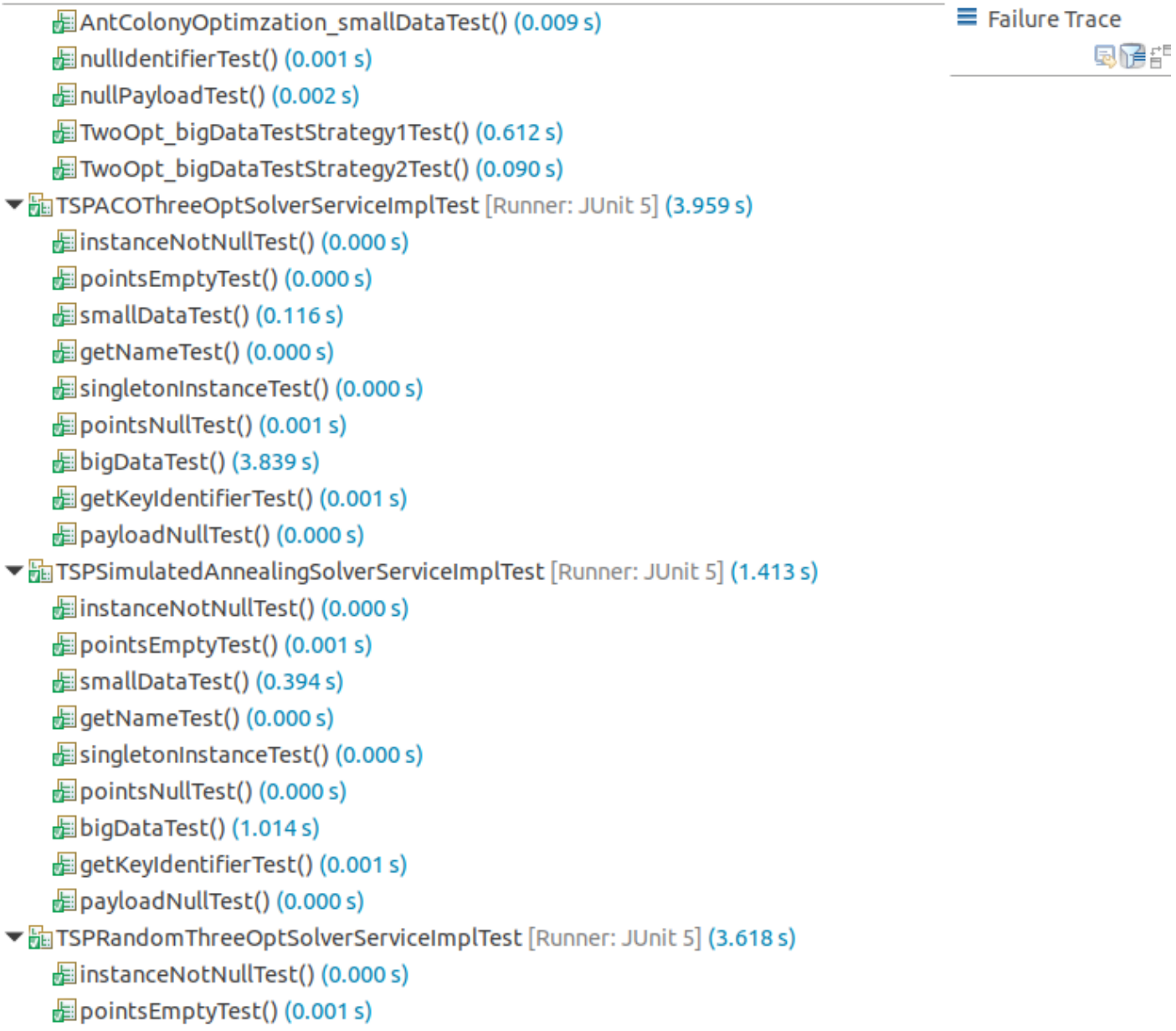
Failure Trace

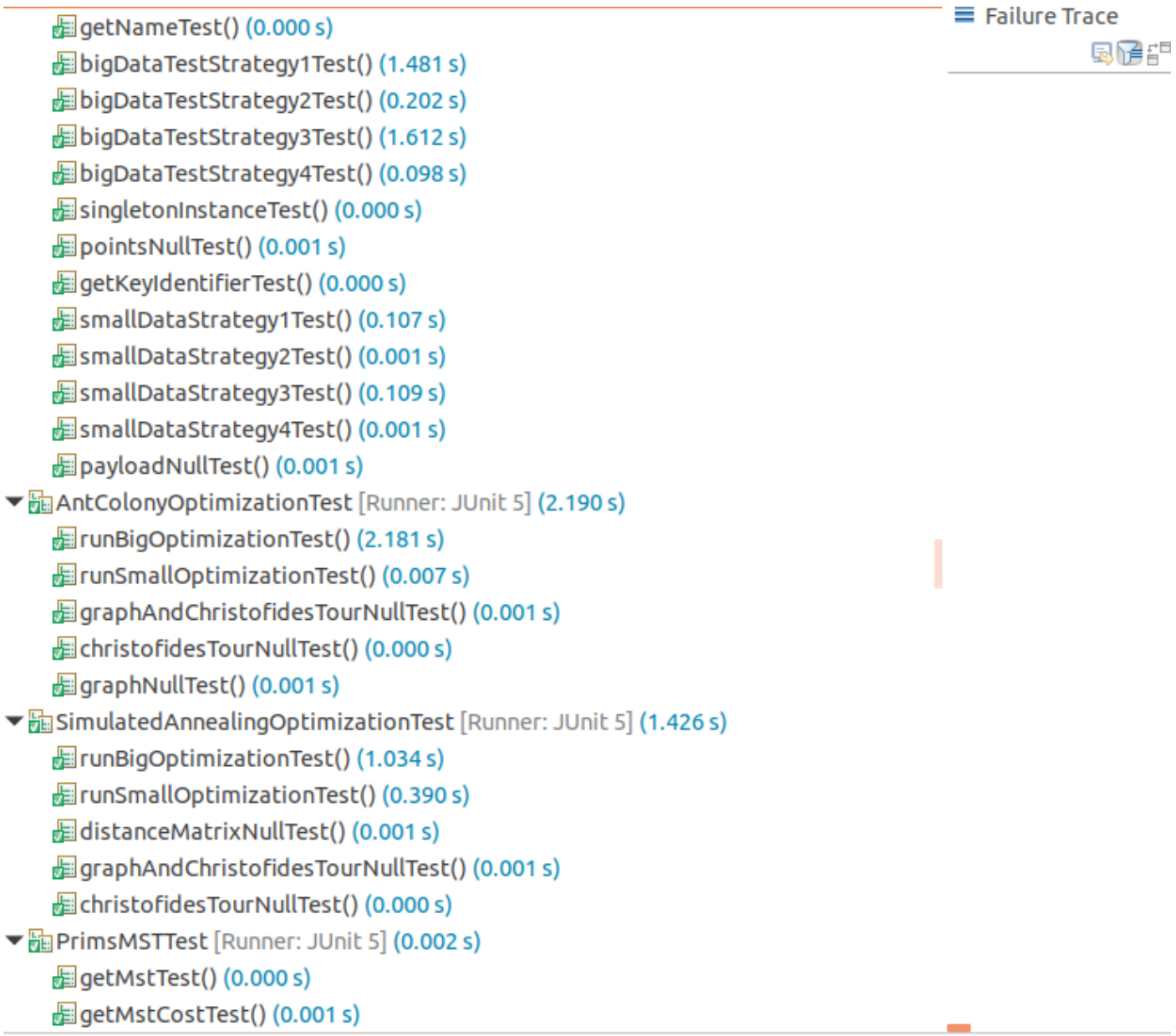

MapServiceImplTest [Runner: JUnit 5] (0.021 s)
publishChangePointColorRedTest() (0.001 s)
publishAddGreenLineWithNullTest() (0.000 s)
publishAddPointsAndFitBoundTest() (0.000 s)
publishAddStartPointMarkerTest() (0.001 s)
publishAddMSTPolylineAndFitBoundWithEmptyListTest() (0.000 s)
publishClearMSTPolylineTest() (0.001 s)
instanceNotNullTest() (0.000 s)
publishOutputWithNullStringTest() (0.001 s)
publishAddPointsAndFitBoundWithEmptyListTest() (0.000 s)
publishChangePointColorGreenTest() (0.001 s)
publishChangePointColorGreenWithNullStringTest() (0.000 s)
publishPointRelaxedWithNullStringTest() (0.001 s)
publishOutputBlankStringTest() (0.000 s)
publishAddMSTPolylineAndFitBoundWithNullTest() (0.000 s)
publishAddPointsAndFitBoundWithNullTest() (0.000 s)
publishAddPolylineAndFitBoundWithNullTest() (0.001 s)
publishChangePointColorGreenBlankStringTest() (0.000 s)
publishDrawEdgeWithNullTest() (0.001 s)
publishChangePointColorRedBlankStringTest() (0.000 s)
publishAddPolylineAndFitBoundTest() (0.000 s)
publishOutputTest() (0.000 s)
singletonInstanceTest() (0.000 s)
publishAddGreenLineTest() (0.000 s)
publishDrawEdgeTest() (0.001 s)
publishAddPolylineAndFitBoundWithEmptyListTest() (0.001 s)
publishClearMapTest() (0.001 s)
publishAddMSTPolylineAndFitBoundTest() (0.001 s)
Failure Trace


publishPointRelaxedTest() (0.001 s)
publishPointRelaxedWithBlankStringTest() (0.001 s)
publishAddStartPointMarkerWithNullTest() (0.001 s)
publishChangePointColorRedWithNullStringTest() (0.001 s)
CSVParserServiceImplTest [Runner: JUnit 5] (0.003 s)
testParsePoints_ValidCSV_ReturnsCorrectNumberOfPoints() (0.001 s)
instanceNotNullTest() (0.000 s)
testParsePoints_EmptyCSV_ReturnsEmptyList() (0.000 s)
singletonInstanceTest() (0.001 s)
KolmogorovWeightedPerfectMatchingImplTest [Runner: JUnit 5] (0.041 s)
instanceNotNullTest() (0.000 s)
emptyDataTest() (0.001 s)
smallDataTest() (0.001 s)
singletonInstanceTest() (0.000 s)
singleNodeIllegalArgumentExceptionTest() (0.001 s)
twoNodesTest() (0.001 s)
bigDataTest() (0.036 s)
TSPAntColonyOptimzationSolverServiceImplTest [Runner: JUnit 5] (2.233 s)
instanceNotNullTest() (0.000 s)
pointsEmptyTest() (0.001 s)
smallDataTest() (0.007 s)
getNameTest() (0.001 s)
singletonInstanceTest() (0.000 s)
pointsNullTest() (0.001 s)
bigDataTest() (2.219 s)
getKeyIdentifierTest() (0.000 s)
payloadNullTest() (0.001 s)
Failure Trace

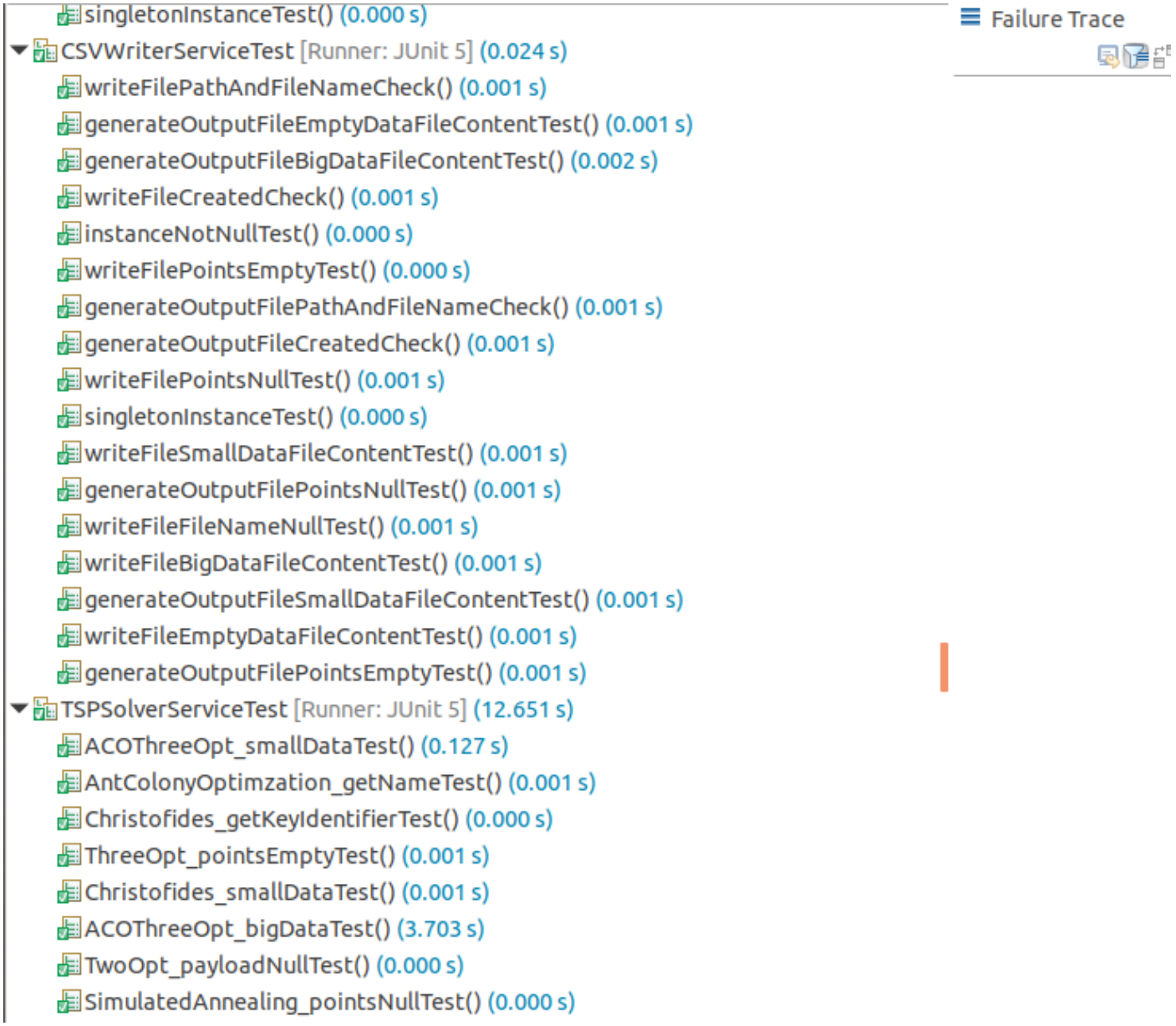

TSPRandomTwoOptSolverServiceImplTest [Runner: JUnit 5] (1.540 s)
instanceNotNullTest() (0.000 s)
pointsEmptyTest() (0.001 s)
getNameTest() (0.000 s)
bigDataTestStrategy1Test() (0.639 s)
bigDataTestStrategy2Test() (0.113 s)
bigDataTestStrategy3Test() (0.683 s)
singletonInstanceTest() (0.000 s)
pointsNullTest() (0.000 s)
getKeyIdentifierTest() (0.001 s)
smallDataStrategy1Test() (0.046 s)
smallDataStrategy2Test() (0.001 s)
smallDataStrategy3Test() (0.050 s)
payloadNullTest() (0.001 s)
TestServiceImplTest [Runner: JUnit 5] (0.003 s)
instanceNotNullTest() (0.000 s)
testAsyncWithEmptyListTest() (0.000 s)
testAsyncTest() (0.000 s)
testAsyncTestWithNullTest() (0.001 s)
singletonInstanceTest() (0.000 s)
TSPSolverFactoryServiceImplTest [Runner: JUnit 5] (0.006 s)
invalidIdentifierTest() (0.000 s)
instanceNotNullTest() (0.000 s)
twoOptTest() (0.001 s)
antColonyTest() (0.000 s)
christofidesTest() (0.000 s)
singletonInstanceTest() (0.000 s)
antColonyThreeOptTest() (0.000 s)
Failure Trace

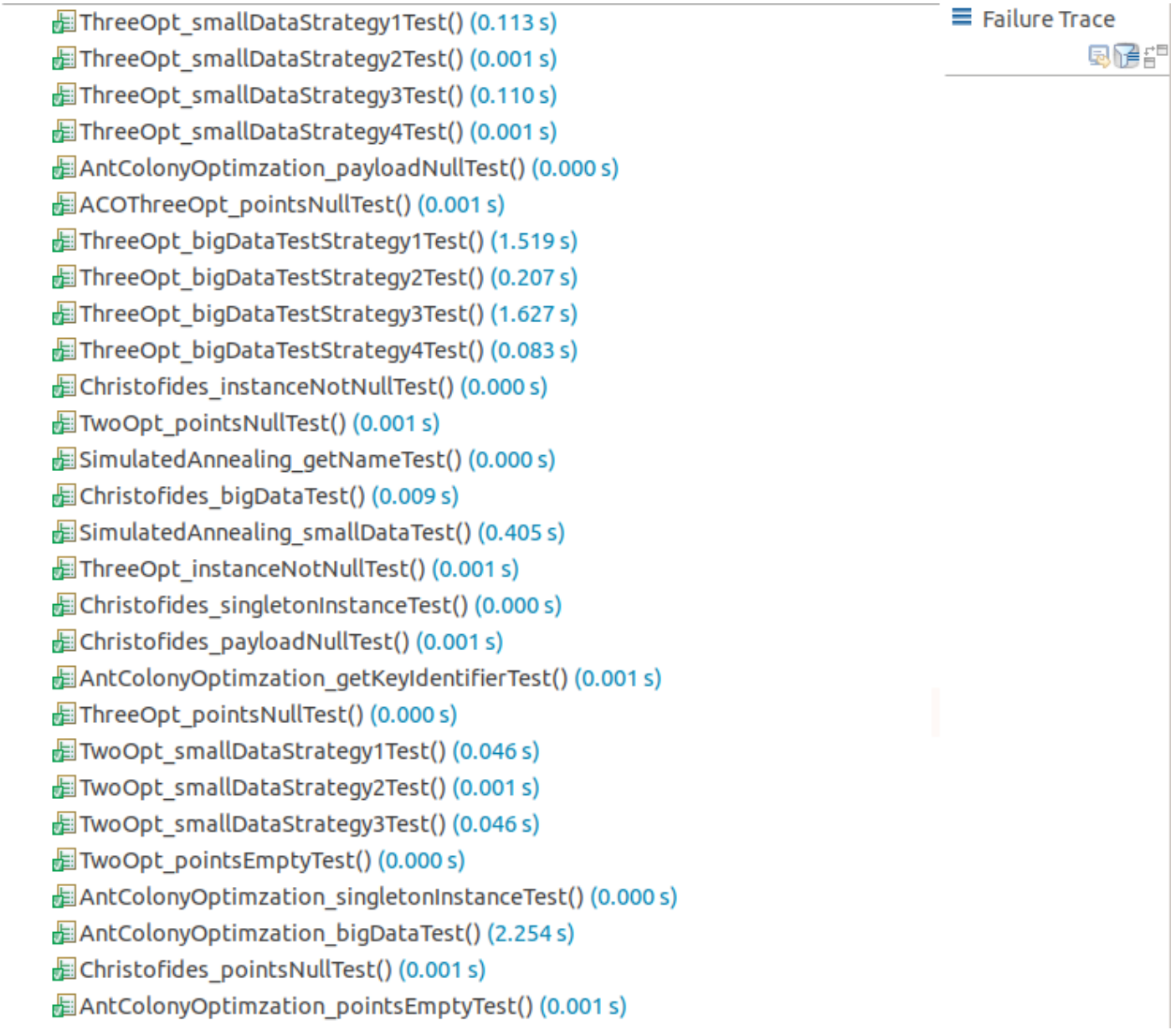

nullIdentifierTest() (0.001 s)
simulatedAnnealingTest() (0.000 s)
threeOptTest() (0.001 s)
TSPServiceImplTest [Runner: JUnit 5] (11.851 s)
ACOThreeOpt_smallDataTest() (0.115 s)
invalidIdentifierTest() (0.001 s)
Christofides_smallDataTest() (0.001 s)
blankIdentifierTest() (0.001 s)
emptyPointsTest() (0.000 s)
ACOThreeOpt_bigDataTest() (3.647 s)
instanceNotNullTest() (0.000 s)
ThreeOpt_smallDataStrategy1Test() (0.109 s)
ThreeOpt_smallDataStrategy2Test() (0.002 s)
ThreeOpt_smallDataStrategy3Test() (0.114 s)
ThreeOpt_smallDataStrategy4Test() (0.002 s)
ThreeOpt_bigDataTestStrategy1Test() (1.503 s)
ThreeOpt_bigDataTestStrategy2Test() (0.204 s)
ThreeOpt_bigDataTestStrategy3Test() (1.550 s)
ThreeOpt_bigDataTestStrategy4Test() (0.056 s)
Christofides_bigDataTest() (0.010 s)
SimulatedAnnealing_smallDataTest() (0.410 s)
TwoOpt_smallDataStrategy1Test() (0.050 s)
TwoOpt_smallDataStrategy2Test() (0.001 s)
TwoOpt_smallDataStrategy3Test() (0.052 s)
nullPointsTest() (0.002 s)
AntColonyOptimzation_bigDataTest() (2.205 s)
singletonInstanceTest() (0.000 s)
SimulatedAnnealing_bigDataTest() (1.090 s)
Failure Trace

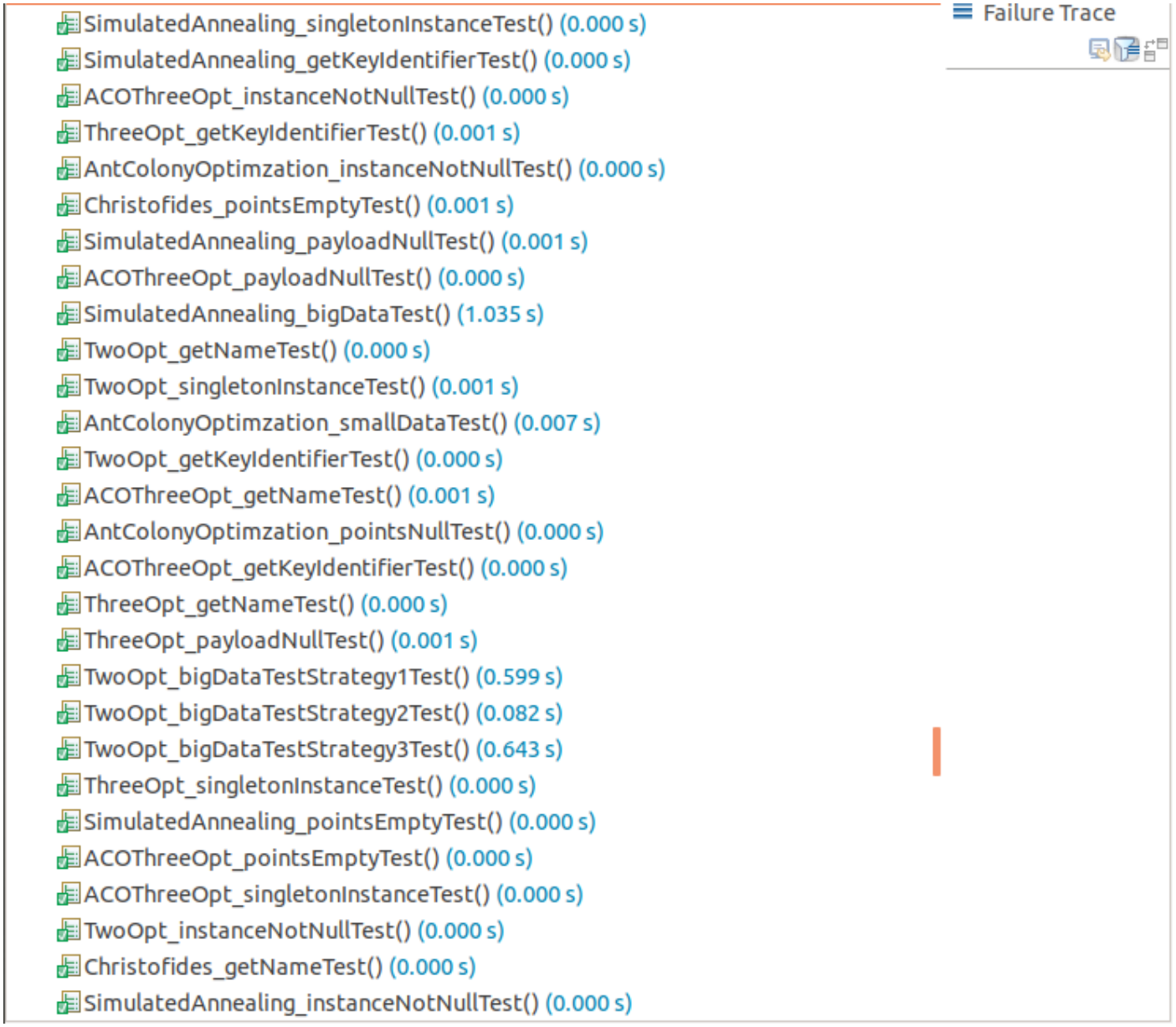

AntColonyOptimzation_smallDataTest() (0.009 s)
nullIdentifierTest() (0.001 s)
nullPayloadTest() (0.002 s)
TwoOpt_bigDataTestStrategy1Test() (0.612 s)
TwoOpt_bigDataTestStrategy2Test() (0.090 s)
TSPACOThreeOptSolverServiceImplTest [Runner: JUnit 5] (3.959 s)
instanceNotNullTest() (0.000 s)
pointsEmptyTest() (0.000 s)
smallDataTest() (0.116 s)
getNameTest() (0.000 s)
singletonInstanceTest() (0.000 s)
pointsNullTest() (0.001 s)
bigDataTest() (3.839 s)
getKeyIdentifierTest() (0.001 s)
payloadNullTest() (0.000 s)
TSPSimulatedAnnealingSolverServiceImplTest [Runner: JUnit 5] (1.413 s)
instanceNotNullTest() (0.000 s)
pointsEmptyTest() (0.001 s)
smallDataTest() (0.394 s)
getNameTest() (0.000 s)
singletonInstanceTest() (0.000 s)
pointsNullTest() (0.000 s)
bigDataTest() (1.014 s)
getKeyIdentifierTest() (0.001 s)
payloadNullTest() (0.000 s)
TSPRandomThreeOptSolverServiceImplTest [Runner: JUnit 5] (3.618 s)
instanceNotNullTest() (0.000 s)
pointsEmptyTest() (0.001 s)
Failure Trace

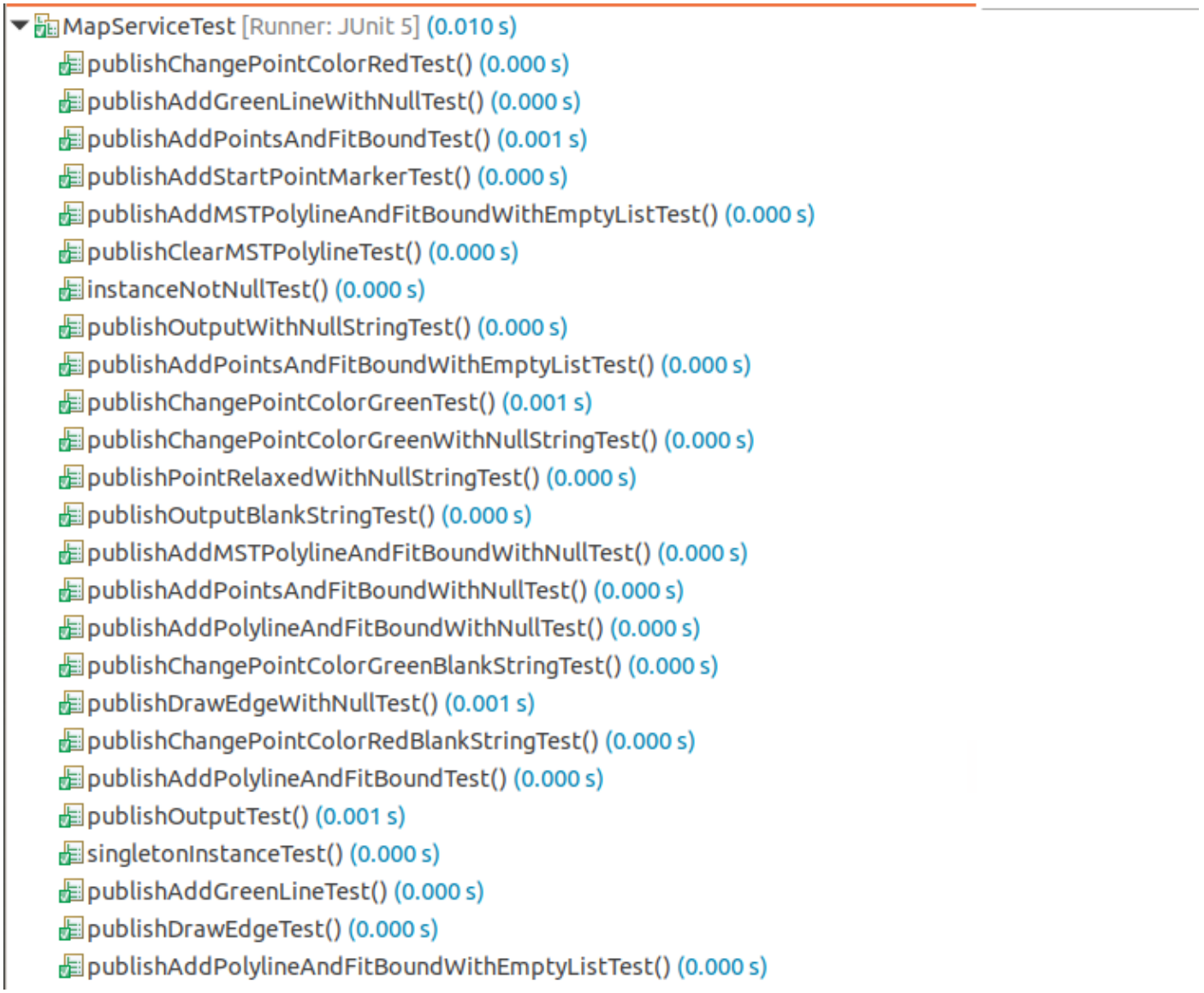

Failure Trace
getNameTest() (0.000 s)
bigDataTestStrategy1Test() (1.481 s)
bigDataTestStrategy2Test() (0.202 s)
bigDataTestStrategy3Test() (1.612 s)
bigDataTestStrategy4Test() (0.098 s)
singletonInstanceTest() (0.000 s)
pointsNullTest() (0.001 s)
getKeyIdentifierTest() (0.000 s)
smallDataStrategy1Test() (0.107 s)
smallDataStrategy2Test() (0.001 s)
smallDataStrategy3Test() (0.109 s)
smallDataStrategy4Test() (0.001 s)
payloadNullTest() (0.001 s)
AntColonyOptimizationTest [Runner: JUnit 5] (2.190 s)
runBigOptimizationTest() (2.181 s)
runSmallOptimizationTest() (0.007 s)
graphAndChristofidesTourNullTest() (0.001 s)
christofidesTourNullTest() (0.000 s)
graphNullTest() (0.001 s)
SimulatedAnnealingOptimizationTest [Runner: JUnit 5] (1.426 s)
runBigOptimizationTest() (1.034 s)
runSmallOptimizationTest() (0.390 s)
distanceMatrixNullTest() (0.001 s)
graphAndChristofidesTourNullTest() (0.001 s)
christofidesTourNullTest() (0.000 s)
PrimsMSTTest [Runner: JUnit 5] (0.002 s)
getMstTest() (0.000 s)
getMstCostTest() (0.001 s)

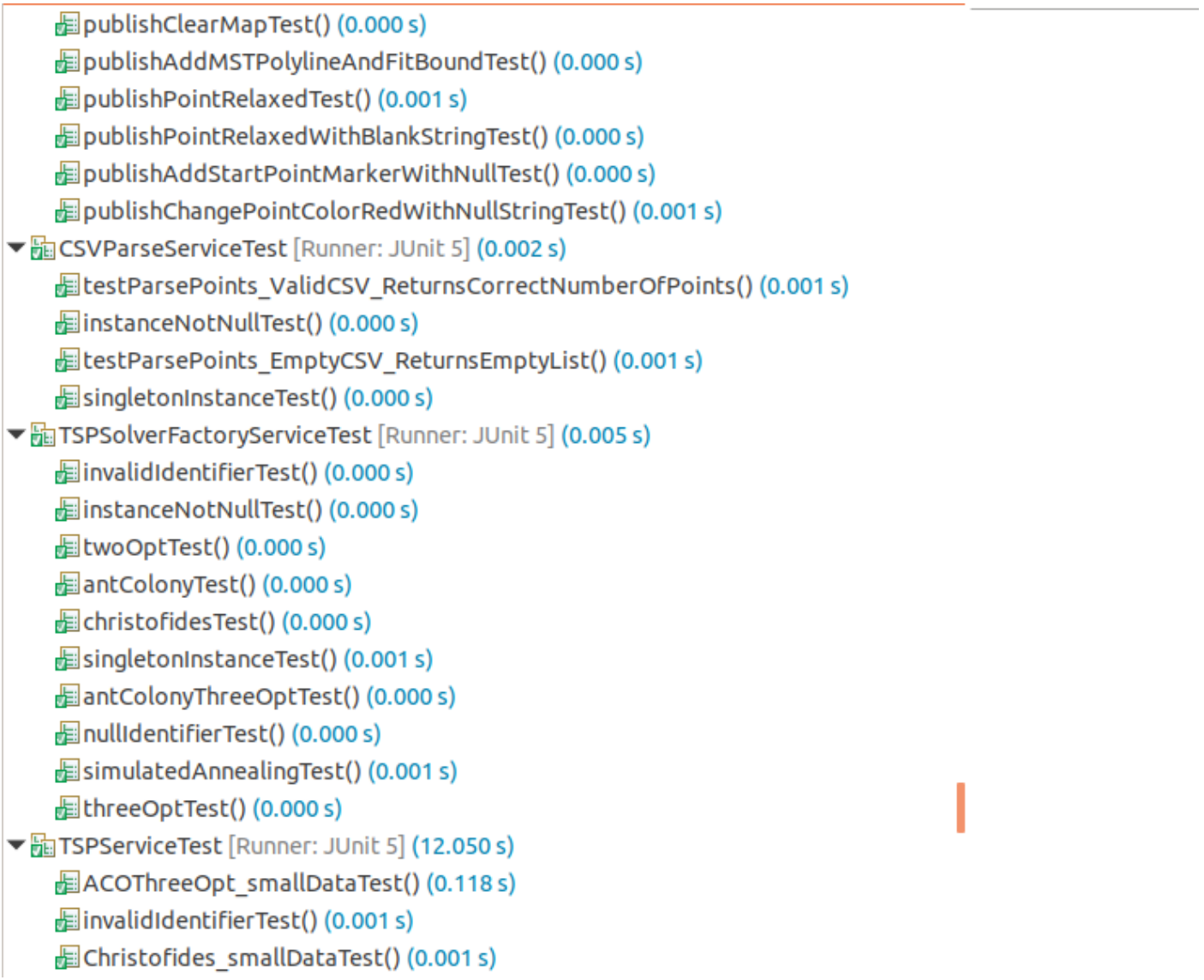

Failure Trace
getMstCostWithEmptyListTest() (0.001 s)
getMstCostBetweenOneNodeOnlyTest() (0.000 s)
getMstCostBetweenTwoNodesTest() (0.000 s)
PerfectMatchingSolverServiceTest [Runner: JUnit 5] (0.029 s)
kolmogorovTwoNodesTest() (0.000 s)
kolmogorovInstanceNotNullTest() (0.000 s)
nearestNeighbourTwoNodesTest() (0.000 s)
nearestNeighbourInstanceNotNullTest() (0.001 s)
nearestNeighbourSingleNodeNullPointerExceptionTest() (0.000 s)
kolmogorovSingletonInstanceTest() (0.001 s)
nearestNeighbourBigDataTest() (0.003 s)
kolmogorovEmptyDataTest() (0.000 s)
nearestNeighbourEmptyListTest() (0.000 s)
kolmogorovBigDataTest() (0.018 s)
nearestNeighbourEmptyDataTest() (0.001 s)
kolmogorovSmallDataTest() (0.001 s)
nearestNeighbourSingletonInstanceTest() (0.000 s)
kolmogorovSingleNodeIllegalArgumentExceptionTest() (0.000 s)
nearestNeighbourSmallDataTest() (0.000 s)
WebSocketPublishServiceTest [Runner: JUnit 5] (0.002 s)
instanceNotNullTest() (0.000 s)
testPublish() (0.000 s)
singletonInstanceTest() (0.001 s)
TestServiceTest [Runner: JUnit 5] (0.003 s)
instanceNotNullTest() (0.000 s)
testAsyncWithEmptyListTest() (0.001 s)
testAsyncTest() (0.000 s)
testAsyncTestWithNullTest() (0.000 s)


singletonInstanceTest() (0.000 s)
CSVWriterServiceTest [Runner: JUnit 5] (0.024 s)
writeFilePathAndFileNameCheck() (0.001 s)
generateOutputFileEmptyDataFileContentTest() (0.001 s)
generateOutputFileBigDataFileContentTest() (0.002 s)
writeFileCreatedCheck() (0.001 s)
instanceNotNullTest() (0.000 s)
writeFilePointsEmptyTest() (0.000 s)
generateOutputFilePathAndFileNameCheck() (0.001 s)
generateOutputFileCreatedCheck() (0.001 s)
writeFilePointsNullTest() (0.001 s)
singletonInstanceTest() (0.000 s)
writeFileSmallDataFileContentTest() (0.001 s)
generateOutputFilePointsNullTest() (0.001 s)
writeFileFileNameNullTest() (0.001 s)
writeFileBigDataFileContentTest() (0.001 s)
generateOutputFileSmallDataFileContentTest() (0.001 s)
writeFileEmptyDataFileContentTest() (0.001 s)
generateOutputFilePointsEmptyTest() (0.001 s)
TSPSolverServiceTest [Runner: JUnit 5] (12.651 s)
ACOThreeOpt_smallDataTest() (0.127 s)
AntColonyOptimzation_getNameTest() (0.001 s)
Christofides_getKeyIdentifierTest() (0.000 s)
ThreeOpt_pointsEmptyTest() (0.001 s)
Christofides_smallDataTest() (0.001 s)
ACOThreeOpt_bigDataTest() (3.703 s)
TwoOpt_payloadNullTest() (0.000 s)
SimulatedAnnealing_pointsNullTest() (0.000 s)
Failure Trace

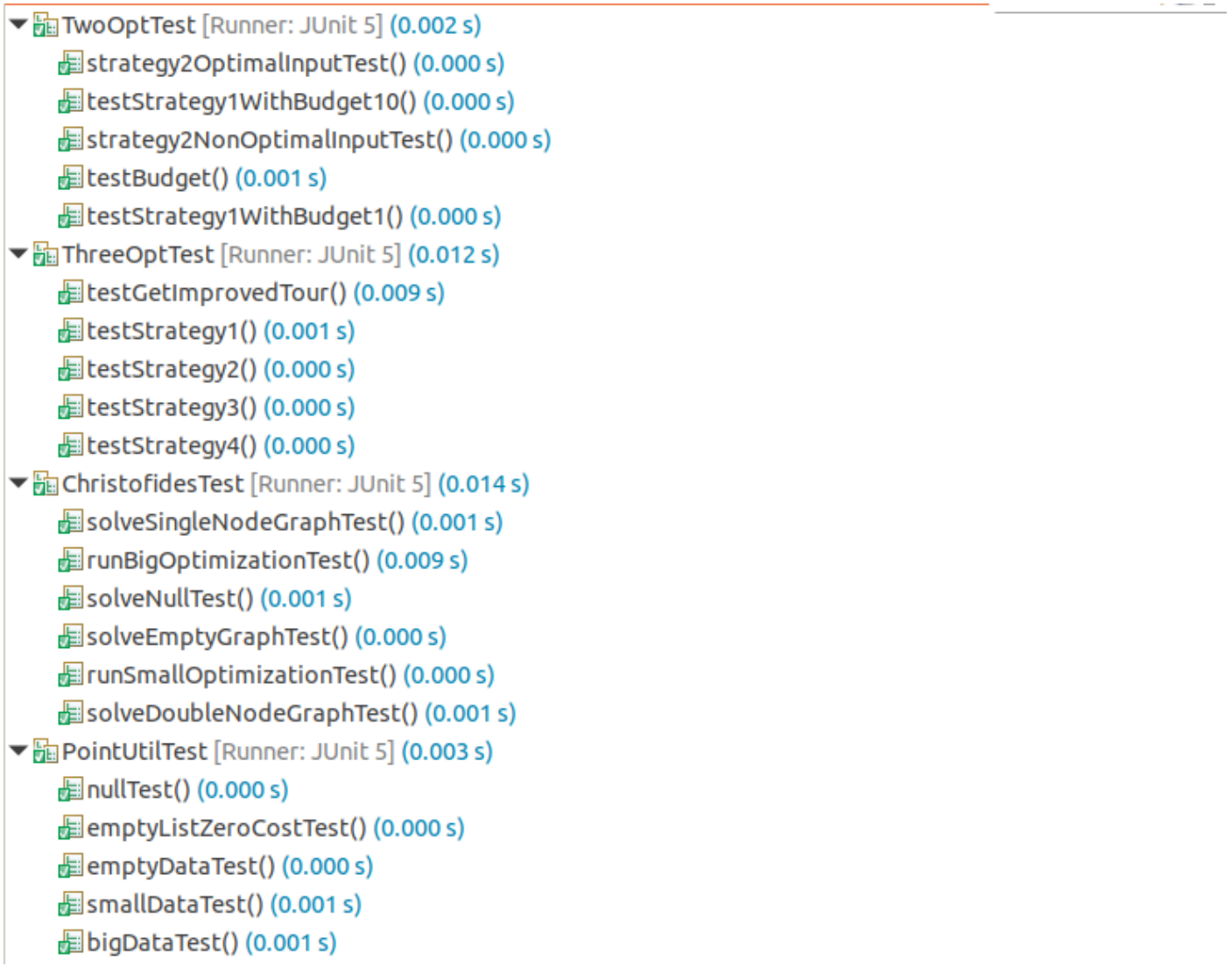

ThreeOpt_smallDataStrategy1Test() (0.113 s)
ThreeOpt_smallDataStrategy2Test() (0.001 s)
ThreeOpt_smallDataStrategy3Test() (0.110 s)
ThreeOpt_smallDataStrategy4Test() (0.001 s)
AntColonyOptimzation_payloadNullTest() (0.000 s)
ACOThreeOpt_pointsNullTest() (0.001 s)
ThreeOpt_bigDataTestStrategy1Test() (1.519 s)
ThreeOpt_bigDataTestStrategy2Test() (0.207 s)
ThreeOpt_bigDataTestStrategy3Test() (1.627 s)
ThreeOpt_bigDataTestStrategy4Test() (0.083 s)
Christofides_instanceNotNullTest() (0.000 s)
TwoOpt_pointsNullTest() (0.001 s)
SimulatedAnnealing_getNameTest() (0.000 s)
Christofides_bigDataTest() (0.009 s)
SimulatedAnnealing_smallDataTest() (0.405 s)
ThreeOpt_instanceNotNullTest() (0.001 s)
Christofides_singletonInstanceTest() (0.000 s)
Christofides_payloadNullTest() (0.001 s)
AntColonyOptimzation_getKeyIdentifierTest() (0.001 s)
ThreeOpt_pointsNullTest() (0.000 s)
TwoOpt_smallDataStrategy1Test() (0.046 s)
TwoOpt_smallDataStrategy2Test() (0.001 s)
TwoOpt_smallDataStrategy3Test() (0.046 s)
TwoOpt_pointsEmptyTest() (0.000 s)
AntColonyOptimzation_singletonInstanceTest() (0.000 s)
AntColonyOptimzation_bigDataTest() (2.254 s)
Christofides_pointsNullTest() (0.001 s)
AntColonyOptimzation_pointsEmptyTest() (0.001 s)
Failure Trace

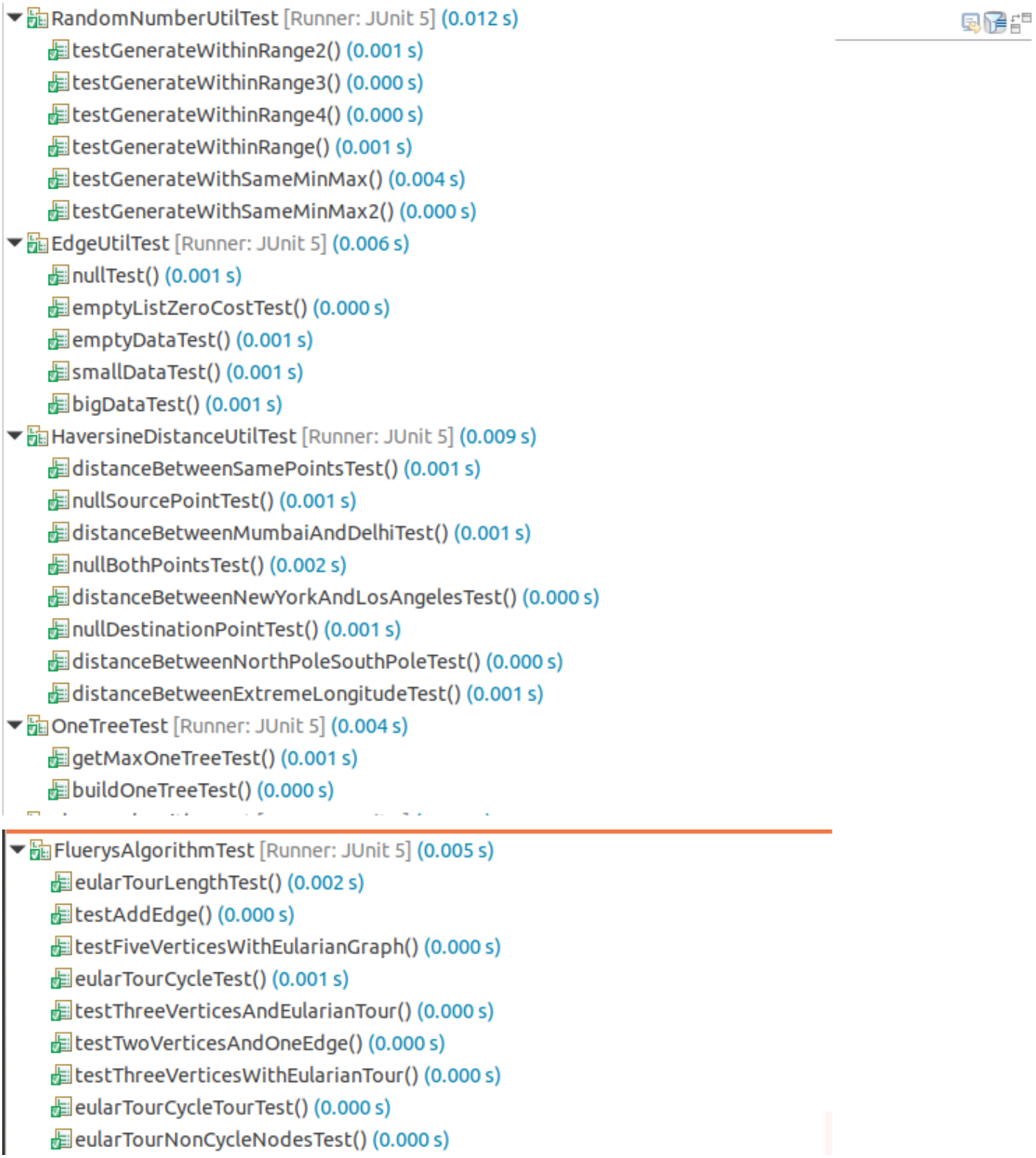

Failure Trace
SimulatedAnnealing_singletonInstanceTest() (0.000 s)
SimulatedAnnealing_getKeyIdentifierTest() (0.000 s)
ACOThreeOpt_instanceNotNullTest() (0.000 s)
ThreeOpt_getKeyIdentifierTest() (0.001 s)
AntColonyOptimzation_instanceNotNullTest() (0.000 s)
Christofides_pointsEmptyTest() (0.001 s)
SimulatedAnnealing_payloadNullTest() (0.001 s)
ACOThreeOpt_payloadNullTest() (0.000 s)
SimulatedAnnealing_bigDataTest() (1.035 s)
TwoOpt_getNameTest() (0.000 s)
TwoOpt_singletonInstanceTest() (0.001 s)
AntColonyOptimzation_smallDataTest() (0.007 s)
TwoOpt_getKeyIdentifierTest() (0.000 s)
ACOThreeOpt_getNameTest() (0.001 s)
AntColonyOptimzation_pointsNullTest() (0.000 s)
ACOThreeOpt_getKeyIdentifierTest() (0.000 s)
ThreeOpt_getNameTest() (0.000 s)
ThreeOpt_payloadNullTest() (0.001 s)
TwoOpt_bigDataTestStrategy1Test() (0.599 s)
TwoOpt_bigDataTestStrategy2Test() (0.082 s)
TwoOpt_bigDataTestStrategy3Test() (0.643 s)
ThreeOpt_singletonInstanceTest() (0.000 s)
SimulatedAnnealing_pointsEmptyTest() (0.000 s)
ACOThreeOpt_pointsEmptyTest() (0.000 s)
ACOThreeOpt_singletonInstanceTest() (0.000 s)
TwoOpt_instanceNotNullTest() (0.000 s)
Christofides_getNameTest() (0.000 s)
SimulatedAnnealing_instanceNotNullTest() (0.000 s)


MapServiceTest [Runner: JUnit 5] (0.010 s)
publishChangePointColorRedTest() (0.000 s)
publishAddGreenLineWithNullTest() (0.000 s)
publishAddPointsAndFitBoundTest() (0.001 s)
publishAddStartPointMarkerTest() (0.000 s)
publishAddMSTPolylineAndFitBoundWithEmptyListTest() (0.000 s)
publishClearMSTPolylineTest() (0.000 s)
instanceNotNullTest() (0.000 s)
publishOutputWithNullStringTest() (0.000 s)
publishAddPointsAndFitBoundWithEmptyListTest() (0.000 s)
publishChangePointColorGreenTest() (0.001 s)
publishChangePointColorGreenWithNullStringTest() (0.000 s)
publishPointRelaxedWithNullStringTest() (0.000 s)
publishOutputBlankStringTest() (0.000 s)
publishAddMSTPolylineAndFitBoundWithNullTest() (0.000 s)
publishAddPointsAndFitBoundWithNullTest() (0.000 s)
publishAddPolylineAndFitBoundWithNullTest() (0.000 s)
publishChangePointColorGreenBlankStringTest() (0.000 s)
publishDrawEdgeWithNullTest() (0.001 s)
publishChangePointColorRedBlankStringTest() (0.000 s)
publishAddPolylineAndFitBoundTest() (0.000 s)
publishOutputTest() (0.001 s)
singletonInstanceTest() (0.000 s)
publishAddGreenLineTest() (0.000 s)
publishDrawEdgeTest() (0.000 s)
publishAddPolylineAndFitBoundWithEmptyListTest() (0.000 s)


publishClearMapTest() (0.000 s)
publishAddMSTPolylineAndFitBoundTest() (0.000 s)
publishPointRelaxedTest() (0.001 s)
publishPointRelaxedWithBlankStringTest() (0.000 s)
publishAddStartPointMarkerWithNullTest() (0.000 s)
publishChangePointColorRedWithNullStringTest() (0.001 s)
CSVParseServiceTest [Runner: JUnit 5] (0.002 s)
testParsePoints_ValidCSV_ReturnsCorrectNumberOfPoints() (0.001 s)
instanceNotNullTest() (0.000 s)
testParsePoints_EmptyCSV_ReturnsEmptyList() (0.001 s)
singletonInstanceTest() (0.000 s)
TSPSolverFactoryServiceTest [Runner: JUnit 5] (0.005 s)
invalidIdentifierTest() (0.000 s)
instanceNotNullTest() (0.000 s)
twoOptTest() (0.000 s)
antColonyTest() (0.000 s)
christofidesTest() (0.000 s)
singletonInstanceTest() (0.001 s)
antColonyThreeOptTest() (0.000 s)
nullIdentifierTest() (0.000 s)
simulatedAnnealingTest() (0.001 s)
threeOptTest() (0.000 s)
TSPServiceTest [Runner: JUnit 5] (12.050 s)
ACOThreeOpt_smallDataTest() (0.118 s)
invalidIdentifierTest() (0.001 s)
Christofides_smallDataTest() (0.001 s)

blankIdentifierTest() (0.001 s)
emptyPointsTest() (0.001 s)
ACOThreeOpt_bigDataTest() (3.860 s)
instanceNotNullTest() (0.000 s)
ThreeOpt_smallDataStrategy1Test() (0.109 s)
ThreeOpt_smallDataStrategy2Test() (0.001 s)
ThreeOpt_smallDataStrategy3Test() (0.109 s)
ThreeOpt_smallDataStrategy4Test() (0.001 s)
ThreeOpt_bigDataTestStrategy1Test() (1.486 s)
ThreeOpt_bigDataTestStrategy2Test() (0.204 s)
ThreeOpt_bigDataTestStrategy3Test() (1.624 s)
ThreeOpt_bigDataTestStrategy4Test() (0.055 s)
Christofides_bigDataTest() (0.008 s)
SimulatedAnnealing_smallDataTest() (0.393 s)
TwoOpt_smallDataStrategy1Test() (0.046 s)
TwoOpt_smallDataStrategy2Test() (0.001 s)
TwoOpt_smallDataStrategy3Test() (0.046 s)
nullPointsTest() (0.001 s)
AntColonyOptimzation_bigDataTest() (2.219 s)
singletonInstanceTest() (0.001 s)
SimulatedAnnealing_bigDataTest() (1.059 s)
AntColonyOptimzation_smallDataTest() (0.007 s)
nullIdentifierTest() (0.001 s)
nullPayloadTest() (0.001 s)
TwoOpt_bigDataTestStrategy1Test() (0.611 s)
TwoOpt_bigDataTestStrategy2Test() (0.079 s)


TwoOptTest [Runner: JUnit 5] (0.002 s)
strategy2OptimalInputTest() (0.000 s)
testStrategy1WithBudget10() (0.000 s)
strategy2NonOptimalInputTest() (0.000 s)
testBudget() (0.001 s)
testStrategy1WithBudget1() (0.000 s)
ThreeOptTest [Runner: JUnit 5] (0.012 s)
testGetImprovedTour() (0.009 s)
testStrategy1() (0.001 s)
testStrategy2() (0.000 s)
testStrategy3() (0.000 s)
testStrategy4() (0.000 s)
ChristofidesTest [Runner: JUnit 5] (0.014 s)
solveSingleNodeGraphTest() (0.001 s)
runBigOptimizationTest() (0.009 s)
solveNullTest() (0.001 s)
solveEmptyGraphTest() (0.000 s)
runSmallOptimizationTest() (0.000 s)
solveDoubleNodeGraphTest() (0.001 s)
PointUtilTest [Runner: JUnit 5] (0.003 s)
nullTest() (0.000 s)
emptyListZeroCostTest() (0.000 s)
emptyDataTest() (0.000 s)
smallDataTest() (0.001 s)
bigDataTest() (0.001 s)



Conclusion

Applying Christofides algorithm to develop a candidate solution of the Travelling Salesman Problem gives a solution in very less amount of time which is also much better than simple algorithms like Nearest Neighbour search.[8]

Furthermore, improvements still exist and can be found using tactical strategies (2 opt. and 3 opt.) or strategic techniques (Simulated Annealing and Ant Colony Optimization).[8]

Using tactical strategies is pretty straightforward as it randomly swaps between edges of the tour and if a better solution is found the tour is updated. However, this greedy approach leads to being stuck in a local minima gravity well.[8]

Simulated Annealing allows the algorithm to accept worst moves in the initial phases of the iteration and as the temperature keeps decaying with time the probability of accepting worst solutions decreases. This approach allows exploration of a much wider solution space and prevents getting stuck in a local minima trap.[8][9]

Finally the application of Ant Colony optimization gives the best results in the least amount of time owing to its population-based nature that allows the existence of multiple solutions which allows ants to probabilistically select the next city to move based on pheromone trial and heuristic information. Also the local search procedures make minor adjustments to refine the solution.[10]

References

[1] "Christofides algorithm," Wikipedia. [Online]. Available: https://en.wikipedia.org/wiki/Christofides_algorithm.
[2] W. Fiset, "MinIndexedDHeap.java," GitHub. [Online]. Available: https://github.com/williamfiset/Algorithms/blob/master/src/main/java/com/williamfiset/algorithms/datastructures/priorityqueue/MinIndexedDHeap.java.
[3] W. Fiset, "EagerPrimsAdjacencyList.java," Github repository file, 2020. [Online]. Available: https://github.com/williamfiset/Algorithms/blob/master/src/main/java/com/williamfiset/algorithms/graphtheory/EagerPrimsAdjacencyList.java.
[4] KolmogorovWeightedMatching.java, GitHub repository, Jgrapht. [Online]. Available:https://github.com/jgrapht/jgrapht/blob/master/jgrapht-core/src/main/java/org/jgrapht/alg/matching/blossom/v5/KolmogorovWeightedMatching.java.

[5] V. Kolmogorov, "Blossom V: a new implementation of a minimum cost perfect matching algorithm," Mathematical Programming Computation, vol. 1, no. 1, pp. 43-67, 2009.
[6] GeeksforGeeks, "Fleury's Algorithm for printing Eulerian Path," [Online]. Available:https://www.geeksforgeeks.org/fleurys-algorithm-for-printing-eulerian-path/.
[7] "AIMA Pseudocode," GitHub repository, 2023. [Online]. Available: https://github.com/aimacode/aima-pseudocode/blob/master/md/Simulated-Annealing.md.
[8] freeCodeCamp.org. (2018). Simulated Annealing Explained [Video]. Retrieved from https://www.youtube.com/watch?v=GiDsjIBOVoA
[9] AI-TechSystems. (2019). Simulated Annealing. Medium. Retrieved from https://medium.com/ai-techsystems/simulated-annealing-580f73bd807a
[10] Jadav, C. (2020). Ant Colony Optimization (ACO). Medium. Retrieved from https://medium.com/@chiragjadav056/ant-colony-optimization-aco-b2716de9252c