Assignment 5: Steepest Delta Search

Mateusz Tabaszewski 151945 Bartłomiej Pukacki 151942

Source code:
https://github.com/MatTheTab/Evolutionary-Computation/blob/main/Assignment%205%20Deltas%20of%20Previous%20Moves/Assignment_5_Deltas.ipynb

1. Description of the Problem

Much like all the previous assignments, this one too is centered around a redefined TSP problem where the aim is to minimize an objective function defined as an Euclidean distance of a cycle connecting the available nodes plus the weight/cost of each node in the solution. Furthermore, only half of all nodes need to be included in the final cycle. Much like the previous report, this time the focus will be on improving the speed of the algorithm rather than the quality of the final solution. However, unlike the previous report algorithm considered, this time is exact in comparison to the steepest local search, meaning that it should produce the same results but obtain them much faster, unlike the previously considered candidate steepest search which resulted in a slight deterioration of the quality of the final obtained solution. In summary, in this report, we will compare the performance and computational speed of the new algorithm, in comparison to all the methods considered thus far with a special focus on retaining the solution quality when compared to the steepest search with edge exchange starting from a random solution.

Decision Variables:

$x_{ij} \in \{0, 1\}$ - included edges

$y_i \in \{0, 1\}$ - visited nodes

Objective Function:

$$min(\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij} + \sum_{i=1}^{n} w_i y_i)$$

sb. t.

$$\sum_{j \in V} x_{ij} = 2y_i \text{ , } \forall i \in V \text{ ; } \textit{where V is a set of all vertices}$$

$$\sum_{i=1}^{n} y_i \geq \frac{n}{2}$$

$$x_{ij} \in \{0, 1\}, \forall i, j \in V$$

$$y_i \in \{0, 1\}, \forall i \in V$$

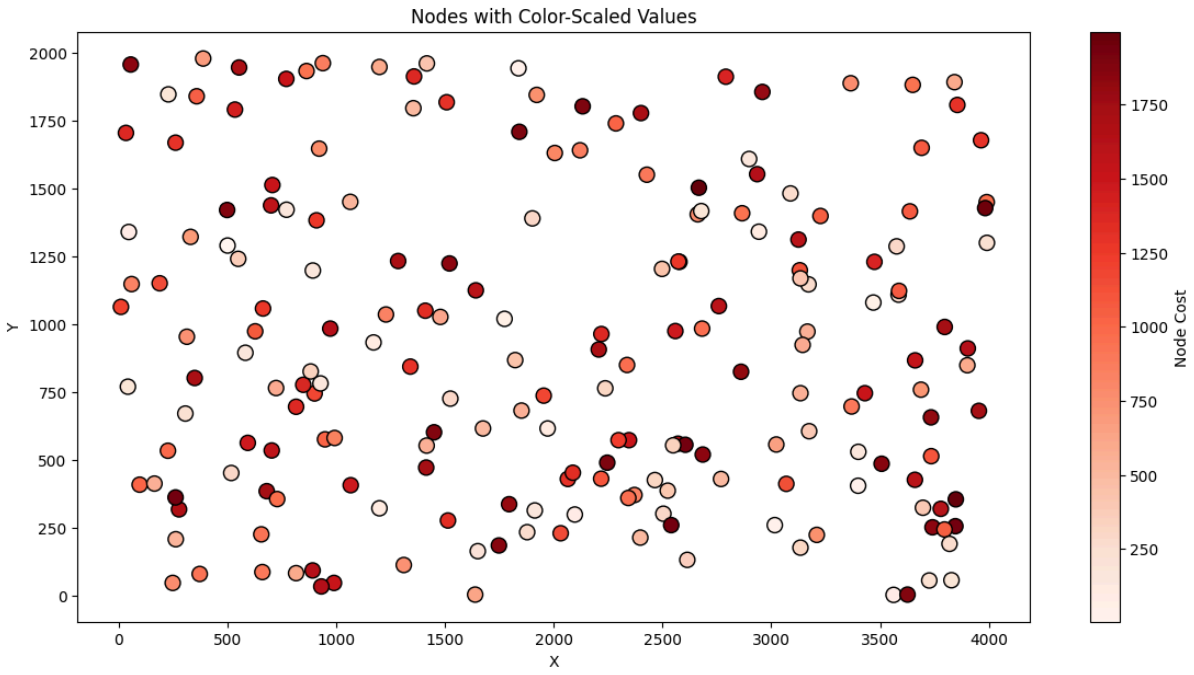

Fig 1. Visualization of the TSPA problem instance, each node's x and y locations on the plot correspond to their given x and y locations and the color intensity signifies the weight/cost of each node. The total length of the cycle and the sum of node weights should be minimized.

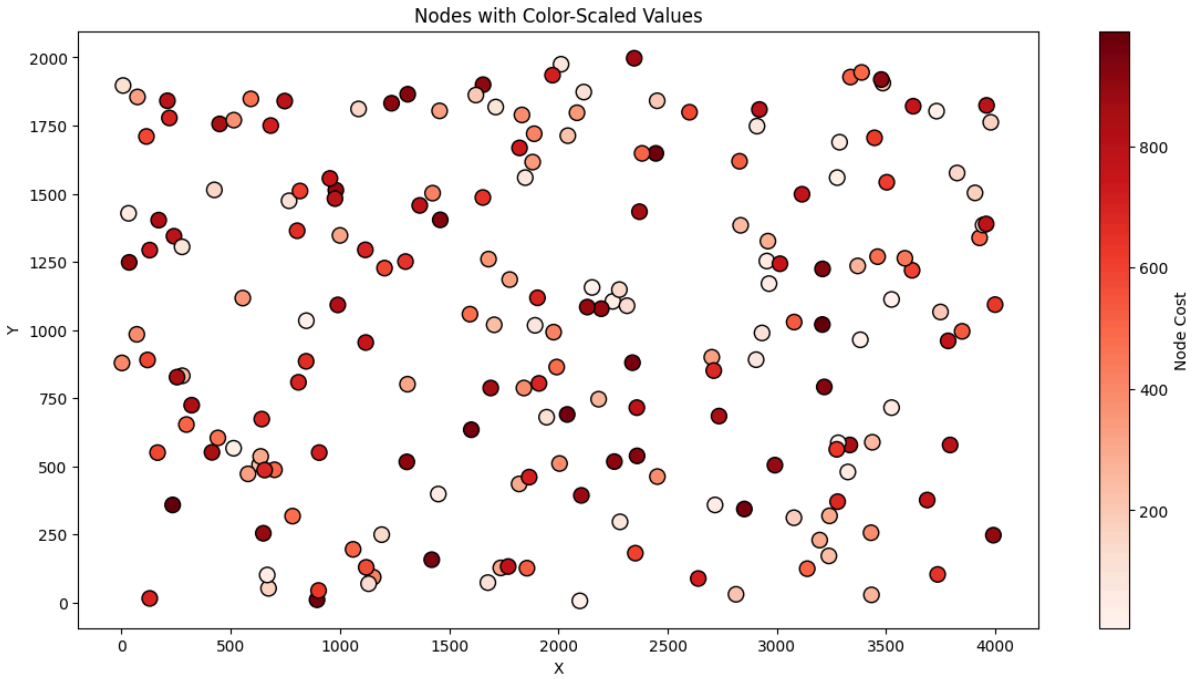

Fig 2. Visualization of the TSPB problem instance, each node's x and y locations on the plot correspond to their given x and y locations and the color intensity signifies the weight/cost of each node. The total length of the cycle and the sum of node weights should be minimized.

2. Algorithms

The algorithm implemented in this assignment and further described in this report is an extension of the original steepest local search algorithm that seeks to improve the given solution until no more local improvements can be made. The implemented extension is meant to speed up the computation time by utilizing deltas, earlier calculated values for certain moves to not recalculate them again in the future. This algorithm can be called **Steepest Local Search with Deltas**, however, for the sake of brevity, we will, from now on, refer to it as the **Steepest Delta Search** algorithm. However, first, let us recall the original implementation of the Steepest Local Search. This algorithm utilizes both inter and intra moves to find which nodes should be switched between those that are within the solution and those that are outside of the current solution, as well as combining the moves exchanging edges already within the solution, it is important to mention that the aforementioned edge exchange must include an inversion of the part of the solution (order of some nodes in the cycle) between the two exchanged edges - this is more relevant in the implementation of the Steepest Delta Search. Another thing of note is that the steepest algorithm always performs the best move and only terminates once no improving moves are left. Having mentioned all relevant aspects of the Steepest Local Search, we can define the differences with Steepest Delta Search. Since in both the redefined TSP and original definition the available to us moves' influences on the score only depend on the neighboring nodes, their impact on the score can be precalculated and then inserted into some data structure like a Sorted List which maintains an internal order of elements and allows for insertions $O(log\ n)$ time complexity. Then the new moves are recalculated as they become available and the moves that are no longer available are removed, this continues until no more available moves are found. Special care must be put into handling edge exchanges as an inversion of the order produces cases in which certain moves no longer become available due to inversion, but they should not be deleted from the list of moves as they can be used in the future once another inversion occurs. Another detail is that we only add moves as viable when the move is worth doing, all moves that worsen the solution are not added to the SortedList. Analytically considering the computational speed-up may be difficult due to the significant overhead introduced in the algorithm, however, let us ignore it for now, and assume the complexity of the standard steepest algorithm is $O(Mn^2)$ where M is the number of moves that need to be performed and n the number of nodes in the solution. Then let us define the complexity of the Steepest Delta Search as $O(Mn\ log\ n)$ with $O(log\ n)$ being the complexity of adding a move to a SortedList and $O(n)$ being the worst case in which all moves need to be recalculated. The theoretical improvement would be equal to $\frac{n}{log\ n} = \frac{100}{log(100)} \sim 15$ times. However, real improvement is likely to be much lower due to the **significant overhead** necessary for the algorithm to function properly.

```
FUNCTION deltas_steepest_local_search_edges(solution, score, distance_matrix,
weights):
    INPUT:
        solution - the starting point solution
        score - the initial score of the solution
        distance_matrix - matrix of distances between nodes
        weights - an array of weights associated with each node

    moves ← always sorted from best to worst moves
    moves += all improving deltas and moves FOR unique pairs of positions of nodes IN
solution AND outside solution
    moves += all improving deltas and moves for unique pairs of nodes IN solution
denoting edge beginnings with at least one node gap between

    i ← 0
    WHILE ANY valid moves remain:
        move_score, move = move[i]
        IF move is NOT between nodes IN solution:
            GET cycle_node, any_node, node_before_cycle_node, node_after_cycle_node FROM
move
            IF cycle_node changed OR any_node changed OR node_before_cycle_node changed
OR node_after_cycle_node changed → move is no longer valid:
                REMOVE move from moves
                CONTINUE back to loop start
            ELSE:
                UPDATE solution (node exchange)
                score += delta OF move
                UPDATE moves (ADD ONLY improving moves)
                i = 0
        ELSE:
            GET cycle_node, any_node, node_after_cycle_node, node_after_any_node FROM
move
            IF cycle_node changed OR any_node changed OR node_after_cycle_node changed
OR node_after_any_node changed:
                REMOVE move from moves
                CONTINUE back to loop start
            ELSE IF move was rotated:
                i += 1
                CONTINUE back to loop start
            ELSE:
                UPDATE solution (edge exchange)
                score += delta OF move
                UPDATE moves (ADD ONLY improving moves)
                i = 0
    RETURN solution, score
```

3. Experiments

To quantify the performance of the Steepest Delta Search algorithm was run 200 times on a random solution. In addition to objective function values, the run times of local searches were analyzed. Please note that due to the lack of seed and the initial random starting solution some very small differences may be present between seemingly exact same methods.

Method	TSPA av (min - max)	TSPB av (min - max)
Steepest Delta Search	**73910 (71118 - 78710)**	**48574 (46300 - 51342)**
Steepest Candidate Search	**77944 (73159 - 84951)**	**48497 (45342 - 52178)**
Steepest LS Edges Rand	**73954 (70948 - 77934)**	**48366 (45576 - 51616)**
Greedy LS Rand	85812 (78831 - 93289)	61000 (53759 - 69662)
Steepest LS Rand	87935 (75935 - 95175)	63036 (55323 - 70187)
Greedy LS Edges Rand	73781 (71507 - 76491)	48427 (45646 - 51763)
Greedy LS Best	71627 (70687 - 72882)	45460 (43826 - 51301)
Steepest LS Best	71619 (70626 - 72950)	45415 (43826 - 50876)
Greedy LS Edges Best	71515 (70571 - 72460)	45040 (43790 - 50495)
Steepest LS Edges Best	71468 (70510 - 72614)	44976 (43921 - 50495)
Random	264301 (223539 - 308435)	213397 (179796 - 253866)
Nearest Neighbor Closest	85109 (83182 - 89433)	54390 (52319 - 59030)
Nearest Neighbor All	73180 (71179 - 75450)	45870 (44417 - 53438)
Greedy Cycle	72606 (71488 - 74350)	51345 (48765 - 57262)
Greedy Regret Cycle	115630 (105852 - 123171)	72656 (67568 - 77329)
Weighted Greedy Regret Cycle	72133 (71108 - 73395)	50882 (47144 - 55700)

Table 1. Minimum, average, and maximum scores achieved by each method on both problem instances.

The **best scores achieved** are visualized below.

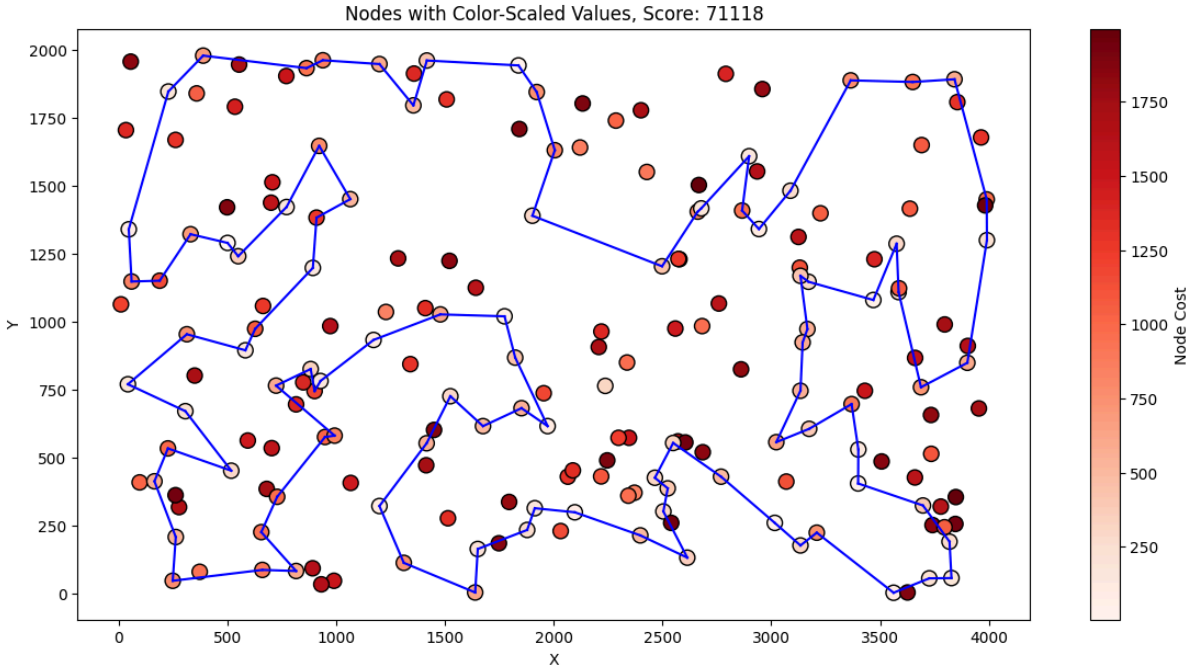

Fig 3. Visualization of the best solution found by the **Steepest Delta Search** on the TSPA problem instance starting from a random solution.

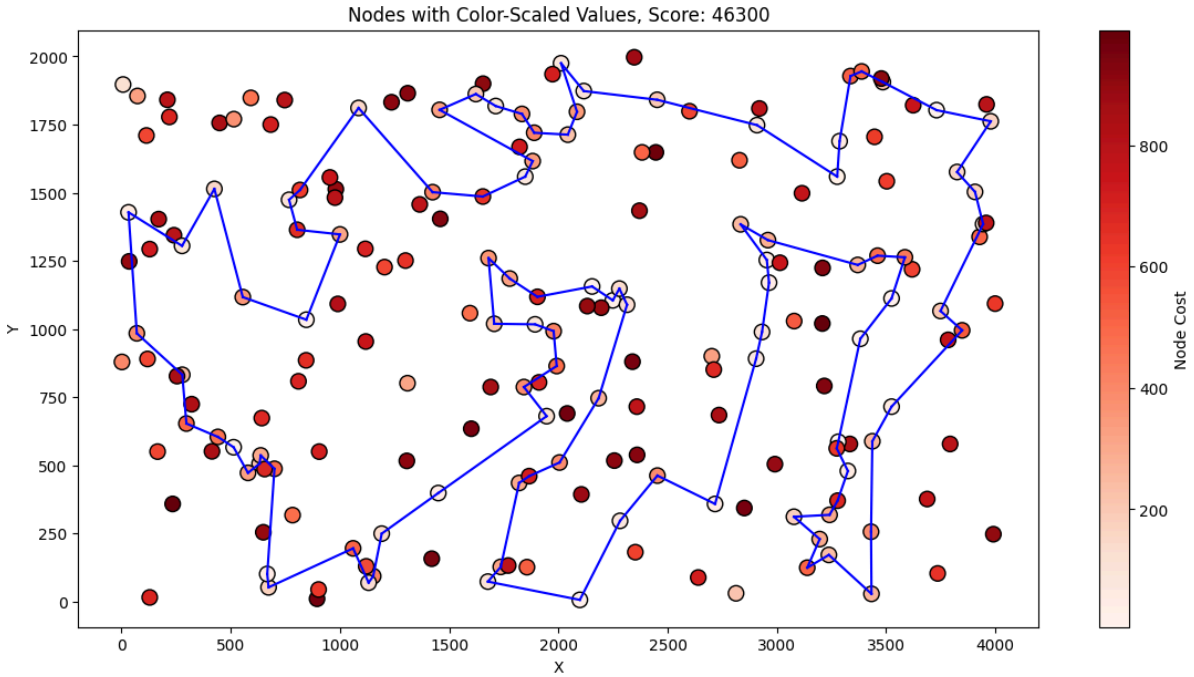

Fig 4. Visualization of the best solution found by the **Steepest Delta Search** on the TSPB problem instance starting from a random solution.

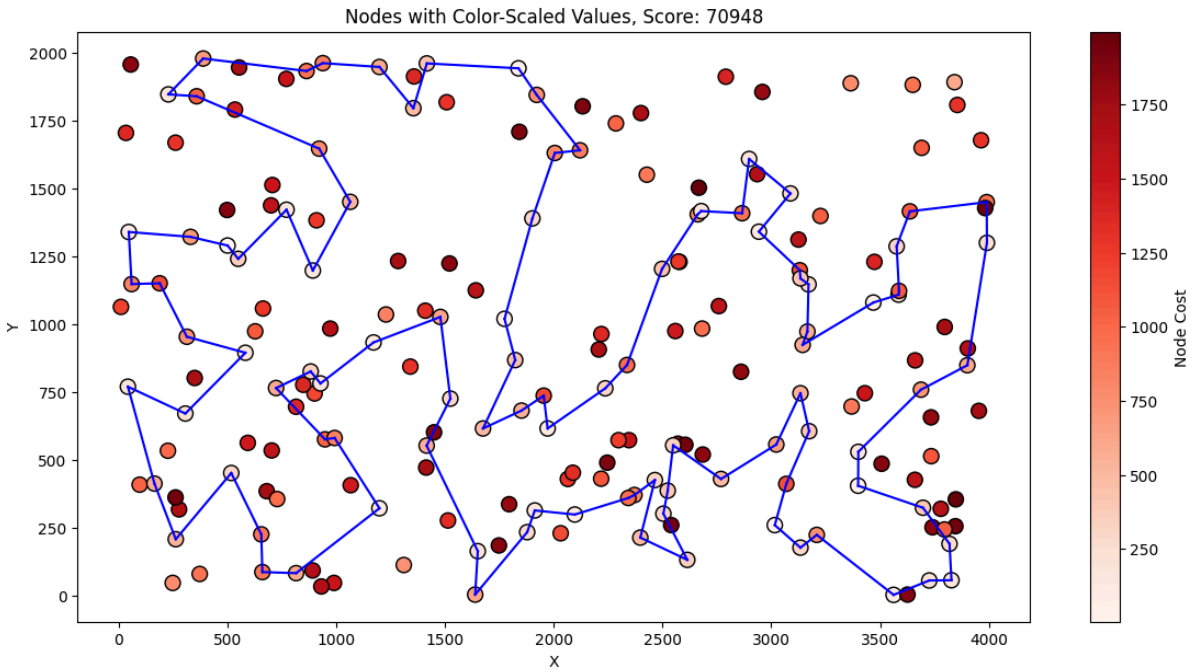

Fig 5. Visualization of the best solution found by the **Steepest LS Edges Rand** on the TSPA problem instance starting from a random solution.

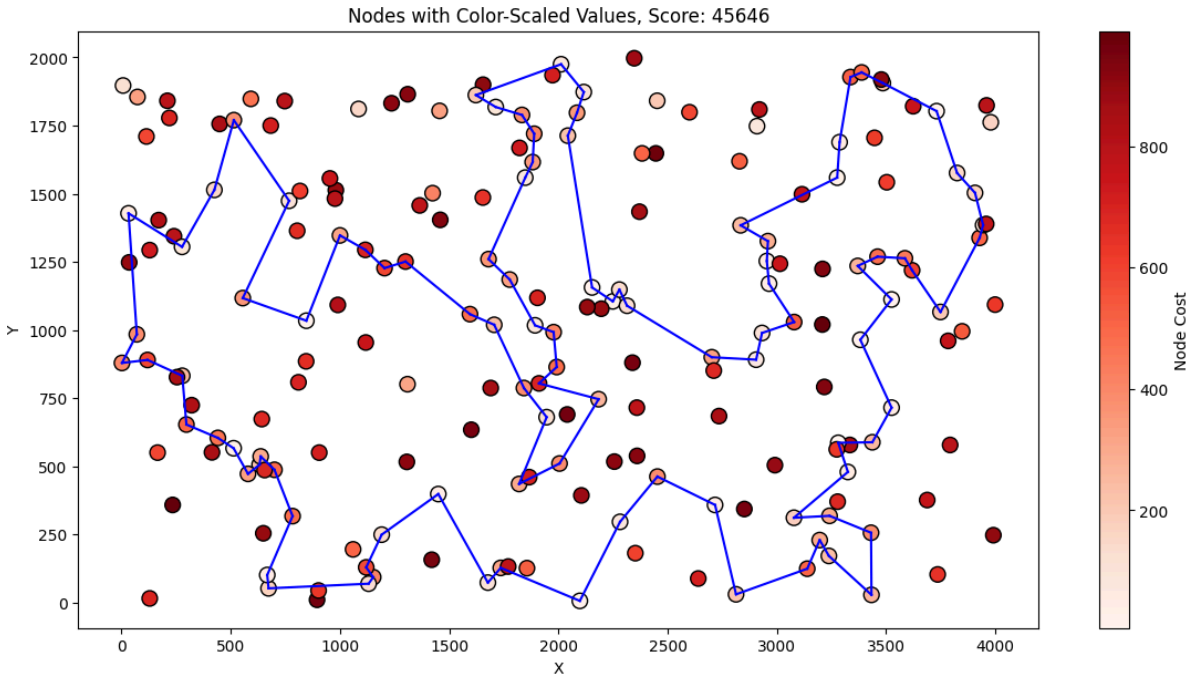

Fig 6. Visualization of the best solution found by the **Steepest LS Edges Rand** on the TSPB problem instance starting from a random solution.

All best solutions were checked using the solution checker spreadsheet available on eKursy. The lists of node indices in the best solutions and their scores are presented in the table below.

Problem instance	Algorithm	Score	Solution
TSPA	**Steepest Delta Search**	**71118**	**159, 193, 41, 139, 68, 46, 198, 115, 5, 42, 181, 34, 160, 184, 48, 54, 177, 10, 4, 112, 84, 35, 131, 149, 43, 116, 47, 65, 59, 51, 176, 80, 63, 79, 133, 151, 162, 123, 127, 70, 135, 154, 180, 53, 86, 75, 101, 1, 97, 152, 2, 120, 44, 25, 16, 171, 175, 113, 31, 78, 145, 179, 92, 129, 57, 55, 52, 178, 106, 185, 165, 40, 196, 81, 90, 27, 164, 7, 21, 14, 49, 102, 144, 62, 9, 148, 137, 23, 89, 183, 143, 0, 117, 93, 140, 108, 18, 22, 146, 195**
	Steepest Candidate Search	77944	181, 34, 160, 48, 54, 177, 10, 190, 4, 112, 184, 42, 43, 77, 105, 116, 65, 59, 197, 118, 51, 151, 133, 162, 123, 127, 135, 154, 180, 53, 26, 86, 75, 101, 1, 97, 152, 2, 120, 44, 25, 82, 129, 92, 145, 78, 31, 16, 171, 175, 113, 196, 81, 40, 185, 165, 90, 164, 7, 21, 14, 144, 62, 9, 102, 49, 178, 106, 52, 55, 57, 167, 148, 94, 189, 63, 122, 79, 80, 176, 137, 23, 89, 183, 143, 0, 117, 93, 68, 46, 115, 139, 41, 193, 159, 69, 108, 18, 22, 146
	Steepest LS Edges Rand	70948	4, 84, 184, 177, 54, 34, 160, 42, 181, 195, 146, 22, 159, 193, 41, 139, 115, 46, 68, 69, 18, 108, 140, 93, 117, 0, 143, 183, 89, 186, 23, 137, 176, 80, 133, 79, 122, 63, 94, 124, 148, 9, 62, 102, 144, 14, 49, 3, 178, 106, 52, 55, 185, 40, 119, 165, 39, 27, 90, 81, 196, 145, 78, 31, 113, 175, 171, 16, 25, 44, 120, 82, 92, 57, 129, 2, 152, 1, 101, 75, 86, 97, 26, 100, 53, 180, 154, 70, 135, 162, 151, 51, 59, 65, 116, 43, 131, 149, 123, 112
	Steepest LS Edges Best	70510	117, 0, 143, 183, 89, 186, 23, 137, 176, 80, 79, 63, 94, 124, 152, 97, 1, 101, 2, 82, 129, 92, 57, 55, 52, 49, 102, 148, 9, 62, 144, 14, 3, 178, 106, 185, 40, 165, 90, 81, 196, 179, 145, 78, 31, 56, 113, 175, 171, 16, 25, 44, 120, 75, 86, 26, 100, 121, 53, 180, 154, 135, 70, 127, 123, 162, 133, 151, 51, 118, 59, 65, 116, 43, 184, 112, 4, 190, 10, 177, 54, 48, 160, 34, 146, 22, 18, 108, 69, 159, 181, 42, 5, 41, 193, 139, 115, 46, 68, 93
	Random	223539	14, 111, 63, 123, 89, 157, 168, 81, 148, 62, 94, 42, 134, 192, 65, 162, 19, 75, 127, 103, 136, 70, 3, 194, 167, 146, 52, 55, 170, 39, 172, 51, 27, 7, 121, 166, 46, 18, 105, 28, 163, 0, 30, 53, 190, 54, 96, 43, 137, 66, 80, 86, 4, 16, 56, 184, 97, 181, 24, 159, 128, 31, 196, 133, 10, 73, 45, 41, 118, 59, 82, 2, 100, 176, 72, 78, 197, 107, 174, 169, 185, 76, 17, 37, 8, 11, 117, 77, 74, 40, 154, 140, 114, 132, 49, 32, 92, 182, 38, 151
TSPB	**Steepest Delta Search**	**46300**	**106, 153, 81, 77, 141, 36, 61, 21, 87, 82, 111, 35, 109, 0, 29, 49, 11, 139, 138, 33, 160, 144, 104, 8, 177, 5, 78, 175, 45, 80, 190, 73, 54, 31, 193, 117, 198, 156, 1, 38, 63, 135, 122, 131, 121, 51, 125, 90, 191, 147, 134, 43, 168, 195, 6, 188, 169, 132, 13, 145, 15, 70, 3, 155, 152, 183, 140, 4, 149, 28, 20, 60, 148, 47, 94, 179, 172, 166, 194, 114, 137, 127, 165, 89, 163, 103, 26, 113, 180, 176, 86, 185, 99, 130, 95, 55, 34, 18, 62, 124**
	Steepest Candidate Search	48497	6, 195, 168, 29, 0, 109, 35, 124, 106, 143, 41, 111, 82, 21, 8, 104, 144, 160, 33, 11, 139, 138, 182, 25, 121, 51, 90, 122, 107, 40, 63, 135, 38, 27, 1, 198, 117, 193, 31, 54, 73, 136, 190, 80, 162, 45, 175, 78, 5, 177, 36, 141, 77, 81, 153, 163, 89, 127, 114, 103, 113, 176, 194, 166, 86, 185, 95, 130, 99, 179, 172, 57, 66, 94, 47, 148, 60, 20, 28, 149, 4, 140, 183, 62, 18, 55, 34,

			170, 152, 184, 155, 3, 70, 15, 145, 13, 132, 169, 188, 188
	Steepest LS Edges Rand	45576	163, 89, 127, 114, 103, 113, 176, 194, 166, 172, 179, 185, 86, 95, 99, 22, 66, 94, 154, 47, 148, 60, 20, 28, 140, 183, 152, 170, 34, 55, 18, 62, 124, 106, 35, 109, 0, 29, 111, 82, 21, 8, 104, 33, 138, 182, 11, 139, 43, 168, 195, 13, 145, 15, 3, 70, 132, 169, 188, 6, 147, 178, 10, 133, 107, 40, 63, 135, 122, 90, 51, 121, 131, 1, 38, 27, 156, 198, 117, 193, 54, 31, 164, 73, 136, 190, 80, 45, 142, 175, 78, 5, 177, 36, 61, 79, 91, 141, 77, 153
	Steepest LS Edges Best	43921	131, 121, 51, 90, 191, 147, 6, 188, 169, 132, 13, 70, 3, 15, 145, 195, 168, 139, 11, 182, 138, 33, 160, 29, 0, 109, 35, 143, 106, 124, 62, 18, 55, 34, 170, 152, 183, 140, 4, 149, 28, 20, 60, 148, 47, 94, 66, 179, 22, 99, 130, 95, 185, 86, 166, 194, 176, 180, 113, 103, 114, 137, 127, 89, 163, 187, 153, 81, 77, 141, 91, 61, 36, 177, 5, 45, 142, 78, 175, 162, 80, 190, 136, 73, 54, 31, 193, 117, 198, 156, 1, 16, 27, 38, 63, 40, 107, 133, 122, 135
	Random	179796	78, 18, 141, 43, 65, 49, 184, 62, 35, 16, 121, 31, 167, 165, 45, 109, 174, 19, 132, 195, 67, 99, 194, 63, 144, 92, 54, 5, 59, 114, 15, 66, 111, 50, 108, 116, 82, 37, 40, 118, 185, 140, 143, 186, 139, 154, 22, 9, 170, 23, 129, 86, 130, 148, 76, 57, 120, 85, 179, 29, 126, 153, 56, 27, 94, 196, 70, 12, 169, 122, 51, 44, 6, 74, 3, 81, 192, 157, 182, 138, 71, 24, 102, 104, 105, 7, 98, 87, 34, 106, 172, 103, 124, 77, 176, 42, 68, 8, 113, 88

Table 2. Best solutions and their scores found by each algorithm in both instances.

Method	TSPA av (min - max) [s]	TSPB av (min - max) [s]
Greedy LS Rand	1.273 (1.047 - 1.975)	1.258 (0.991 - 1.646)
Steepest LS Rand	4.283 (3.261 - 6.218)	4.501 (3.292 - 5.609)
Greedy LS Edges Rand	1.171 (0.981 - 1.34)	1.113 (0.945 - 1.446)
Greedy LS Best	0.067 (0.025 - 0.145)	0.077 (0.033 - 0.187)
Steepest LS Best	0.170 (0.055 - 0.529)	0.196 (0.09 - 0.746)
Greedy LS Edges Best	0.062 (0.025 - 0.115)	0.078 (0.036 - 0.212)
Steepest LS Edges Best	0.194 (0.078 - 0.379)	0.229 (0.114 - 0.836)
Steepest LS Edges Rand	**3.571 (2.978 - 4.168)**	**3.654 (2.976 - 4.364)**
Steepest Candidate Search	**0.584 (0.479 - 0.705)**	**0.562 (0.481 - 0.693)**
Steepest Delta Search	**0.486 (0.400 - 0.625)**	**0.483 (0.383 - 0.610)**

Table 3. Minimum, average, and maximum run time achieved by local search methods on both problem instances.

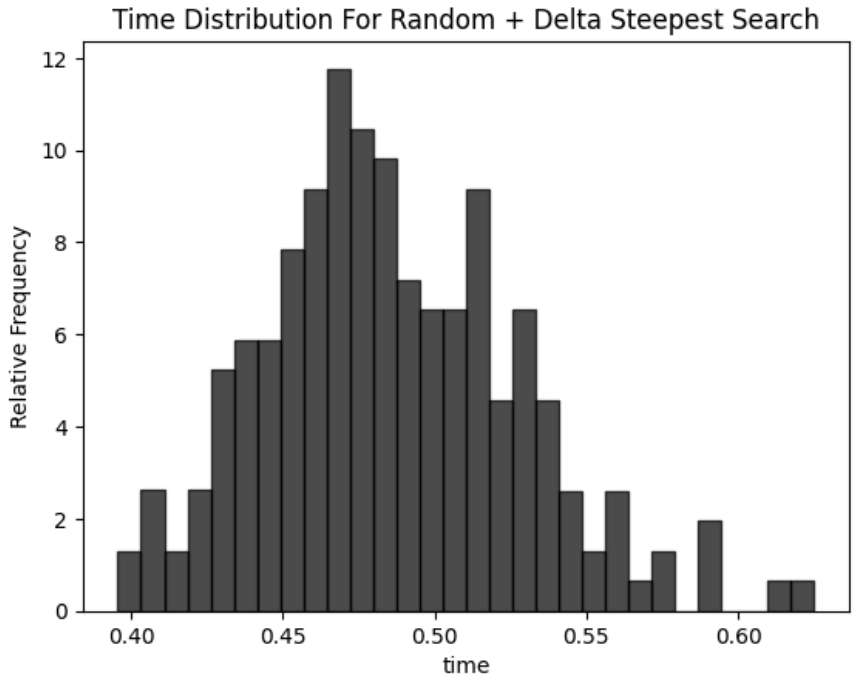

Fig 7. Visualization of time distributions for the use of the **Steepest Delta Search** on the TSPA problem instance starting from a random solution.

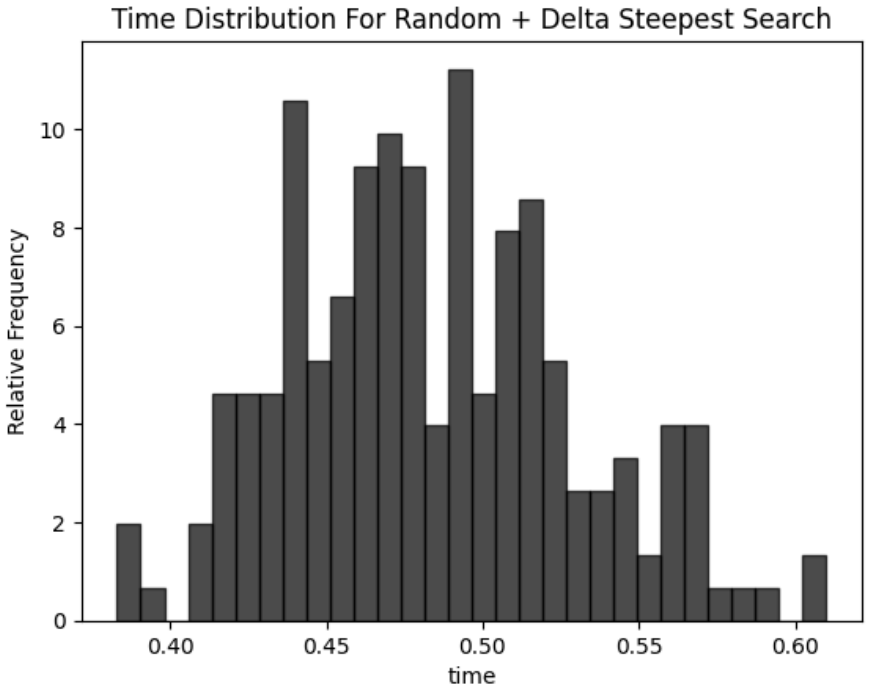

Fig 8. Visualization of time distributions for the use of the **Steepest Delta Search** on the TSPB problem instance starting from a random solution.

4. Conclusions

The Steepest Delta Search algorithm achieves the results much faster than the basic Steepest Local Search without the utilization of deltas. Furthermore, the algorithm achieves even better performance than the Candidate Local Search despite showing much better performance score-wise than this algorithm. The Steepest Delta algorithm offers a 7.34 times greater speed for the TSPA instance and a 7.57 times greater improvement on the computational speed for the TSPB instance. Furthermore, a 1.2 and a 1.16 times speedup is present between the Steepest Delta Search and the Candidate Local Search algorithm for the TSPA and TSPB instances respectively. In short, the implementation of the deltas in the local search algorithms allows for the computational speed up in any of the local algorithms without the loss of the quality of the final solution as long as the described method allows for a computation of the changes in the final score independent of some of the other nodes in the solution. In other words, if the change in the solution score is dependent on all of the nodes like in some problems, then using deltas would not be possible, however as long as the change in score is independent of at least some node states, then using deltas should provide a computational speed-up. However, it is important to note that depending on the used language and implementation details, the algorithm may require a significant computational overhead, meaning that the speed-up will be much smaller than a theoretical maximum which is why special care needs to be put in making the actual code as optimized as possible to fully utilize the speed-up offered by the use of the Steepest Delta Search algorithm.