

MD-mfcmTSP algorithm

Panagiotis Zachos

May 2024

1 Pseudocode for the MD-mfcmTSP

At the end of Algorithm 1, a local optimization function (3) is called which in addition to moving nodes in different places in the same route, also moves nodes between routes of different depots and different vehicle types.

Algorithm 1: MD-mfcmTSP heuristic

Input: G_T, G_M, \dots, G_D
Output: $Sol = \{Sol^i = \{R_T^i, R_M^i, \dots, R_D^i\}, Sol^{i+1}, \dots, Sol^m\}$
for each $i \in D$

- 1 Create clusters K^i of customer nodes for each depot $d^i \in D$
- 2 by assigning each customer to the closest possible depot
- 3 **for** each $d^i \in D$ **do**
- 4 Call $Initialization(d^i, K^i)$
- 5 **while** $(M_T^i > M_M^i \parallel M_T^i > M_D^i) \ \&\& \ stop \neq true$ **do**
- 6 $diff_M = M_T^i - M_M^i$
- 7 $diff_D = M_T^i - M_D^i$
- 8 **if** $diff_M \geq diff_D$ **then**
- 9 $vt = M$
- 10 $cap = \text{Motorbike's capacity}$
- 11 **else**
- 12 $vt = D$
- 13 $cap = 1$
- 14 **end if**
- 15 $M_{min} = M_T^i$
- 16 $r_{best} = \emptyset$
- 17 **for** $j = 1$ **to** $|R_T^i| - cap$ **do**
- 18 $successive_nodes = \emptyset$
- 19 $load = 0$
- 20 **while** $load + v_j^{demand} \leq cap \ \&\& \ v_j \in G_{vt}$ **do**
- 21 $successive_nodes += v_j$
- 22 **end while**
- 23 **if** $|successive_nodes| == cap$ **then**
- 24 $r_{new} = R_T^i[0] + \{successive_nodes\} + R_T^i[0]$
- 25 $R_{vt}^i = R_{vt}^i + r_{new}$
- 26 $M'_{vt} = R_{vt}^i$'s makespan
- 27 $R_T^i = R_T^i - \{successive_nodes\}$
- 28 $M'_T = R_T^i$'s makespan
- 29 $M_{max} = MAX(M'_T, M'_{vt})$
- 30 **if** $M_{max} < M_{min}$ **then**
- 31 $M_{min} = M_{max}$
- 32 $r_{best} = r_{new}$
- 33 **end if**
- 34 $r_{new} = \emptyset$
- 35 **end if**
- 36 $j += 1$
- 37 **end for**
- 38 **if** $M_{min} < R_T^i$ **then**
- 39 $R_T^i = R_T^i - \{r_{best}^{customers}\}$
- 40 $M_T = R_T^i$'s makespan
- 41 $R_{vt}^i += r_{best}$
- 42 $M_{vt} = R_{vt}^i$'s makespan
- 43 Call $local_optimization(R_T^i)$
- 44 Call $local_optimization(R_{vt}^i)$
- 45 **else**
- 46 $stop = true$
- 47 **end if**
- 48 **end while**
- 49 $Sol^i = \{R_T^i, R_M^i, \dots, R_D^i\}$
- 50 **end for**
- 51 $M_T = MAX(M_T^i, M_T^{i+1}, \dots, M_T^m)$
- 52 $M_M = MAX(M_M^i, M_M^{i+1}, \dots, M_M^m)$
- 53 $M_D = MAX(M_D^i, M_D^{i+1}, \dots, M_D^m)$
- 54 $M_{total} = MAX(M_T, M_M, \dots, M_D)$

Algorithm 2: Initialization(d^i, K^i)

- 1 **while** $\{K^i\} \cap \{G_T\} \neq \emptyset$ **do**
- 2 $R_T^i += \text{NearestNeighbour}(\{K^i\} \cap \{G_T\})$
- 3 **end while**
- 4 $M_T^i = R_T^i$'s makespan
- 5 $v_{free} = \{K^i\} - \{G_T\}$
- 6 **if** $v_{free} = \emptyset$ **then**
- 7 **return** R_T^i
- 8 **else**
- 9 **while** $v_{free} \neq \emptyset$ **do**
- 10 **if** $M_T - M_M \geq M_T - M_D \parallel G_D = \emptyset$ **then**
- 11 $R_M^i += \text{NearestNeighbour}(\{K^i\} \cap \{G_M\})$
- 12 $v_{free} = v_{free} - \{R_M^i\}$
- 13 $M_M^i = R_M^i$'s makespan
- 14 **else**
- 15 $R_D^i += \text{closest}(\{K^i\} \cap \{G_D\})$
- 16 $M_D^i = R_D^i$'s makespan
- 17 **end if**
- 18 **end while**
- 19 **end if**
- 20 **return** Sol^i

Algorithm 3: $local_opt_full(Sol, n_{max})$

- 1 Call $vt_optimization(Sol, n_{max} = 2)$
- 2 **for** each vt **do**
- 3 **for** each $i \in D$ **do**
- 4 Call $local_optimization(R_{vt}^i, n_{max} = 2)$
- 5 **end for**
- 6 $M_{vt} = MAX(R_{vt}^i, R_{vt}^{i+1}, \dots, R_{vt}^m)$
- 7 Call $mutual_optimization(R_{vt}, n_{max} = 2)$
- 8 **end for**
- 9 Call $vt_optimization(Sol, n_{max} = 2)$

Algorithm 4: $local_optimization(r, n_{max})$

- 1 **for** $n = 1$ **to** n_{max} **do**
- 2 **for** each combination of n successive nodes on the route
- 3 move the node(s) to a different place on the same route
- 4 evaluate the new route
- 5 **if** this route is better than the original and all constraints are satisfied **then**
- 6 replace the original route with the new one
- 7 continue in point 3 unless all possible places in the route have been evaluated
- 8 **end for**
- 9 **end for**
- 10 **return** r

Algorithm 5: *mutual_optimization*(R_{vt}, n_{max})

```
1 for  $n = 1$  to  $n_{max}$  do
2   for each possible pair of depots  $c1$  and  $c2$ 
3     for each combination of  $n$  successive nodes in the
       route of  $c1$ 
4       remove the nodes from the route of  $c1$  and insert
         them into  $c2$ 
5       evaluate the newly-created routes
6       if  $MAX(|R_{vt}^{c1}|, |R_{vt}^{c2}|) < MAX(|R_{vt}^{c1}|, |R_{vt}^{c2}|)$  and
         all constraints are satisfied then
7         replace the original routes with the new ones
8         continue in point 4 unless all possible places in  $c2$ 
           have been evaluated
9     end for
10  end for
11 end for
12 return  $R_{VT}$ 
```

Algorithm 6: *vt_optimization*(Sol, n_{max})

```
1 for  $n = 1$  to  $n_{max}$  do
2   for each depot  $i \in D$ 
3     for each possible pair of vehicle types  $t1, t2 \in VT$ 
4       for each combination of  $n$  successive nodes in
          $R_{t1}^i$ 
5         remove the nodes from  $R_{t1}^i$  and insert them in
            $R_{t2}^i$ 
6         if  $MAX(|R_{t1}^i|, |R_{t2}^i|) < MAX(|R_{t1}^i|, |R_{t2}^i|)$ 
           and all constraints are satisfied then
7           replace the original routes with the new
             ones
8           continue in point 5 unless all possible places
             in  $R_{t2}^i$  have been evaluated
9       end for
10  end for
11 end for
12 end for
13 return  $Sol$ 
```

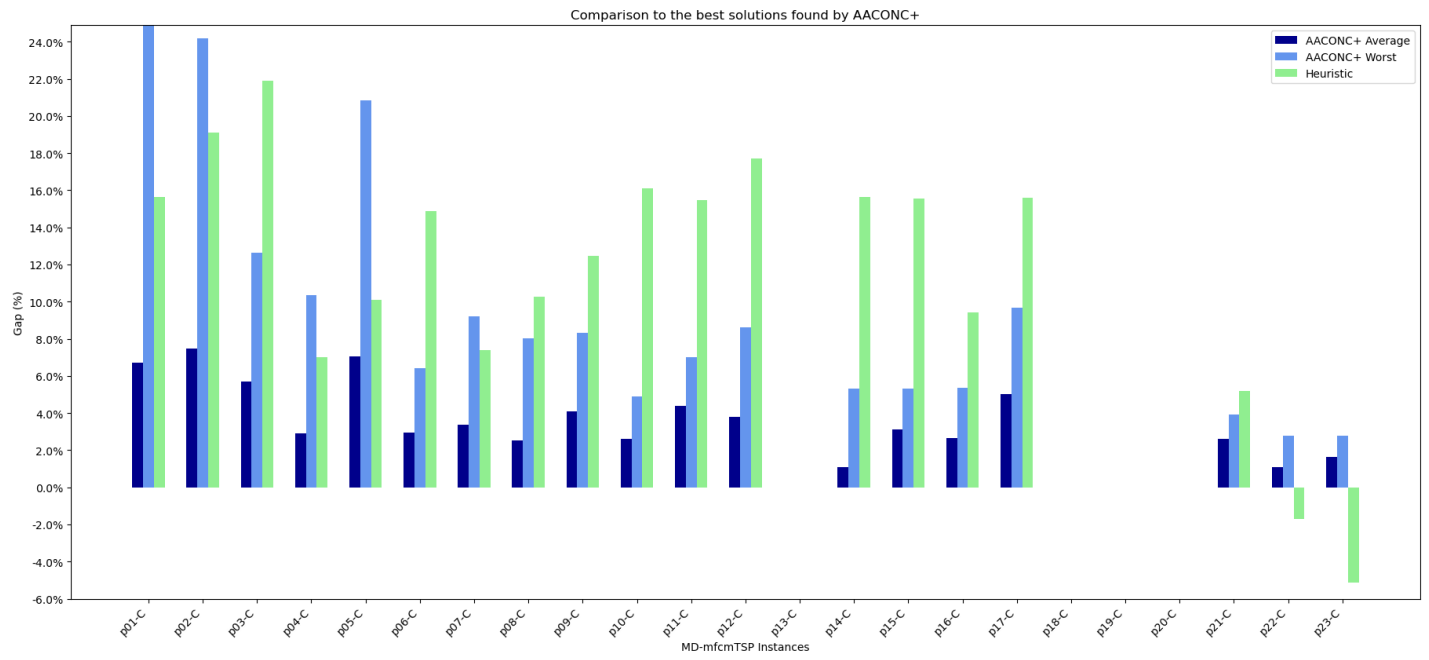


Table 1: Local optimization impact on heuristic

Instance	Best	Full Local Opt	gap(%)	No Local Opt	gap(%)	No Swap Local Opt	gap(%)	No Final Local Opt	gap(%)
p01-C	215.15	217.42	1.06	334.04	55.26	215.15	0.00	173.96	27.33
p02-C	218.4	219.20	0.37	308.83	41.41	218.40	0.00	289.58	32.59
p03-C	202.43	214.84	6.13	253.18	25.07	202.43	0.00	255.52	26.23
p04-C	668.81	668.81	0.00	779.34	16.53	711.80	6.43	691.62	3.41
p05-C	630.78	630.78	0.00	726.62	15.19	655.4	3.90	697.86	10.63
p06-C	431.32	435.42	0.95	650.32	50.77	431.32	0.00	564.79	30.94
p07-C	309.48	309.48	0.00	350.77	13.34	349.19	12.83	366.51	18.43
p08-C	3079.58	3333.93	8.26	3321.63	7.86	3079.58	0.00	3428.15	11.32
p09-C	1917.82	1917.82	0.00	2429.24	26.67	2102.98	9.65	2303.70	20.12
p10-C	1493.13	1493.13	0.00	1864.23	24.85	1525.37	2.16	1706.14	14.27
p11-C	1101.33	1145.75	4.03	1239.36	12.53	1101.33	0.00	1276.59	15.91
p12-C	1273.77	1273.77	0.00	1356.09	6.46	1307.11	2.62	1273.77	0.00
p13-C	1191.96	1226.63	2.91	1455.87	22.14	1191.96	0.00	1259.97	5.71
p14-C	1248.53	1284.73	2.90	1413.87	13.24	1248.53	0.00	1302.34	4.31
p15-C	1347.67	1347.67	0.00	1508.50	11.93	1355.08	0.55	1375.95	2.10
p16-C	1289.29	1289.29	0.00	1448.53	12.35	1328.00	3.00	1289.29	0.00
p17-C	1282.38	1320.91	3.00	1375.55	7.27	1282.38	0.00	1320.91	3.00
p18-C	1260.24	1260.24	0.00	1446.92	14.81	1317.54	4.55	1339.83	6.32
p19-C	1266.92	1266.92	0.00	1406.13	10.99	1289.95	1.82	1301.62	2.74
p20-C	1323.1	1323.10	0.00	1429.00	8.00	1338.42	1.16	1351.39	2.14
p21-C	1309.14	1363.18	4.13	1470.03	12.29	1309.14	0.00	1363.18	4.13
p22-C	1295.67	1295.67	0.00	1465.74	13.13	1351.96	4.34	1306.6	0.84
p23-C	1252.36	1252.36	0.00	1396.11	11.48	1262.65	0.82	1254.42	0.16
AVG	1113.44	1134.39	1.46	1279.56	18.85	1138.07	2.34	1199.72	10.54

Table 2: AACONC+ Results

Instance	Best	Average	gap(%)	Worst	gap(%)	Average time(s)
p01-C	188.00	200.66	6.73	234.78	24.88	73
p02-C	184.01	197.78	7.48	228.47	24.16	47
p03-C	176.26	186.33	5.71	198.56	12.65	202
p04-C	624.93	642.98	2.89	689.6	10.35	1173
p05-C	573.03	613.47	7.06	692.44	20.84	884
p06-C	379.10	390.34	2.96	403.39	6.41	1316
p07-C	288.24	297.91	3.35	314.78	9.21	798
p08-C	3023.77	3100.23	2.53	3265.97	8.01	1635
p09-C	1705.03	1774.86	4.10	1846.92	8.32	3391
p10-C	1286.28	1319.78	2.60	1349.47	4.91	3549
p11-C	992.16	1035.63	4.38	1061.7	7.01	3600
p12-C	1082.12	1123.30	3.81	1175.39	8.62	446
p13-C	1099.02	1135.30	3.30	1191.96	8.46	491
p14-C	1111.12	1123.33	1.10	1170.17	5.31	423
p15-C	1166.08	1202.57	3.13	1228.27	5.33	2471
p16-C	1178.51	1209.60	2.64	1241.67	5.36	2146
p17-C	1142.74	1199.88	5.00	1253.47	9.69	2628
p18-C	1231.96	1263.90	2.59	1296.42	5.23	3368
p19-C	1250.26	1266.54	1.30	1284.23	2.72	3600
p20-C	1246.36	1266.70	1.63	1280.77	2.76	3445
p21-C	1296.19	1330.11	2.62	1347.09	3.93	3600
p22-C	1317.94	1332.07	1.07	1354.61	2.78	3600
p23-C	1319.93	1341.50	1.63	1356.62	2.78	3600
Average	1037.52	1067.59	3.46	1107.25	8.68	2021.13

