# CCVRP optimization with Genetic and ACS

- ## Problem Description:

The Clustered capacitated vehicle routing problem (CCVRP) consist of n-1 costumers with certain need and one depot with some vehicles with specific amount of capacity.

Each customer vi ($i \in \{1,\ldots,n\}$) has a known nonnegative demand di to be delivered or collected and the depot has a fictitious demand $d0 = 0$. There exist m identical vehicles, each with a capacity Q and in order to ensure feasibility we assume that $di \leqslant Q$ for each $i \in \{1,\ldots,n\}$.
Problem assumption:

- each route starts and ends at the depot vertex;
- once a vehicle enters a cluster, it visits all the vertices within the cluster before leaving it;
- the sum of the demands of the visited vertices by a route does not exceed the capacity of the vehicle, Q.

- ## Instances Description:

Instances are created based on CVRP instances form TSPLIB library with difference that we created new problem that is a clustered version of CVRP.

Each CVRP instance file consists of two part as **specification part** that contains information about the instance data and **data part**.

- ## Algorithm Description:

The algorithm designed base on the related paper as *(A novel two-level optimization approach for clustered vehicle routing problem).*

Our approach is obtained by decomposing the problem into two logical and natural subproblems: an upper-level (global) subproblem and a lower-level (local) subproblem. The first subproblem aims at determining the routes visiting the clusters, called global routes, using a genetic algorithm applied to the corresponding global graph (see details in Section 3) while the aim of the second subproblem is to determine the visiting order within the clusters for the above-mentioned routes. The second subproblem is solved by transforming each global route into a TSP which then is computed optimally using the Concorde TSP solver.

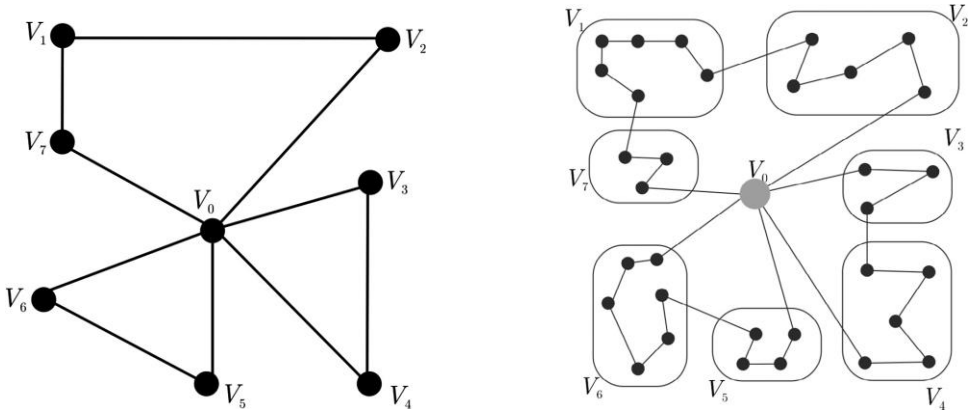
- ## Global subproblem with GA:

As mentioned, we solve this subproblem using GA, so we describe out GA as follow parts:

- Representation:

  Global subproblem search space is consist of number of clusters Vi ($i \in \{0\ldots,n\}$) witch depot node placed in V0 cluster .

So out representation would be a sequence of cluster numbers that shows out global routes toward clusters. Depot cluster would be seen repeatedly as finish each route from depot and back to it.




As images shows one feasible solution can be: (7 1 2 0 3 4 0 5 6)
For creating the chromosome, we list cluster needs in descending and trying to satisfy needs by minimum vehicle number.

- Fitness function:

  The fitness function of each individual chromosome in the population is given by the total length of the best corresponding clustered routes associated to the collection of global routes specified by the chromosome. This distance also takes into account the order in which the vertices within the clusters are visited. Our aim is to minimize this total distance.

- CrossOver:

  Our GA uses a custom version of the two-point crossover. The crossover function takes two parent candidate solutions as input and outputs two solutions.

  Cross over acts like two-point crossover but it allows repetition of depot cluster (because at the end of each route we came back to depot) as number of it was repeated in the parent chromosome.

  P1 = (6 8 0 | 1 3 7 | 0 5 4 2)
  P2 = (7 2 1 | 6 0 4 | 3 0 5 8)

  O1 = (2 6 0 1 3 7 4 0 5 8)

O2 = (8 0 1 3 6 0 4 7 5 2)

- Mutation:

  we use a swapping inter-cluster mutation operator which acts as follows: we randomly select genes (i.e. clusters) and if the genes are from different global routes, their position is exchanged.

  (5 8 1 0 3 7 0 6 4 2) -> (5 6 1 0 3 7 0 8 4 2)

- Parent Selection:

  Two parents with same chromosome size randomly selected from 30% of best population.

- Survivor Selection:
  Elitism manner used in the way that only best individual of every generation moving into the next generation.

- Local subproblem:

The basic idea used in our transformation is to add an artificial cost M to all the inter-cluster edges in this way forcing the vehicle to visit all the vertices within the cluster before leaving it.

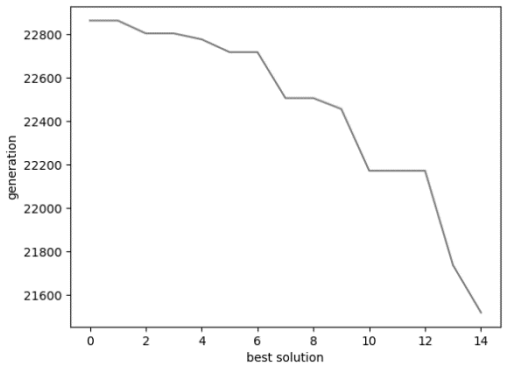
1. The set of nodes of Gp and Gp′ are the same.
2. The entries of the cost matrix Gp′ are defined as follows:
   (a) if vi,vj ∈ Vk then c′ (vi,vj) = c (vi,vj);
   (b) if vi ∈ Vk,vj ∈ Vl with k ≠ l then c′ (vi,vj) = c (vi,vj) + M;
   where M > Σ ∈ c (vi,vj).

After that we passing the edge matrix to TSP solver with bellow configuration.

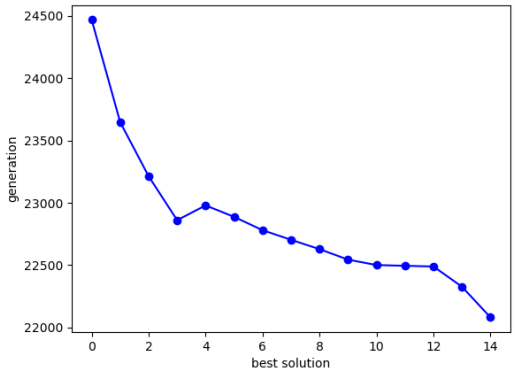
For solving local subproblem open source TSP solver used as bellow.

```
solver.read_mat(mat)
config = TwoOpt_solver(initial_tour='NN', iter_num=100)
        answer = solver.get_approx_solution(config )
```

- GA learning process:




Best solution per generation




average solution per generation

- GA Results:

Because of HW limitation of execution time (1 minute per instance) bellow configuration selected.

MUTATION_RATE = 0.2
POPULATION_SIZE = 40
MAX_GENERATION = 10
XOVER_METHOD = ORDER_2POINT
SELECTION = RANDOM
SURVIVOR_SEL_TYPE = ELITISM (best will be kept)

| ρ = 10% | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **index** | **clusters** | **vehicles** | **vertices** | **Q** | **BKS** | **Best** | **average** | **worst** | **variance** | **avg_time** | **avg vehicles** |
| 1 | 120 | 9 | 241 | 550 | 5759/25 | 16532.41 | 17153.33 | 17830.24 | 406.655 | 22.984 | 9 |
| 2 | 101 | 10 | 321 | 700 | 9247/92 | 22933.83 | 23503.14 | 24101.87 | 287.613 | 32.732 | 10 |
| 3 | 96 | 10 | 401 | 900 | 12904/6 | 29986.90 | 30851.00 | 31644.01 | 524.867 | 73.037 | 10 |
| 4 | 104 | 10 | 481 | 1000 | 17810/4 | 41385.85 | 42151.04 | 43103.16 | 503.638 | 115.14 | 10 |
| 5 | 49 | 5 | 201 | 900 | 8960/31 | 16526.34 | 17232.05 | 17702.95 | 334.257 | 56.114 | 5 |
| 6 | 67 | 7 | 281 | 900 | 10976/5 | 22204.49 | 22861.75 | 23504.54 | 372.046 | 64.244 | 7 |
| 7 | 88 | 9 | 361 | 900 | 12485/8 | 28235.49 | 28795.24 | 29269.60 | 378.717 | 71.651 | 9 |
| 8 | 108 | 11 | 441 | 900 | 13331/2 | 33223.84 | 34325.43 | 35153.69 | 519.279 | 84.987 | 11 |
| 9 | 51 | 15 | 256 | 1000 | 710/64 | 974.90 | 1022.87 | 1045.15 | 19.996 | 32.074 | 16.0 |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 10 | 56 | 18 | 324 | 1000 | 908/89 | 1243.49 | 1267.09 | 1300.11 | 19.571 | 45.193 | 18.0 |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| ρ = 25% | | | | | | | | | | | |
| **index** | **clusters** | **Vehicles** | **vertices** | **Q** | **BKS** | **best** | **average** | **Worst** | **variance** | **avg_time** | **avg vehicles** |
| 1 | 40 | 10 | 241 | 550 | 6051/04 | 9188.307 | 9592.559 | 9899.38 | 166.623 | 52.46 | 10 |
| 3 | 38 | 10 | 401 | 900 | 13692/6 | 17823.20 | 18334.96 | 18584.4 | 210.22 | 50.949 | 10 |
| 5 | 19 | 5 | 201 | 900 | 9340/7 | 11125.28 | 11383.95 | 11579.1 | 140.223 | 41.615 | 5 |
| **7** | 34 | 9 | 361 | 900 | 12348/1 | 16028.19 | 16513.08 | 17077.0 | 301.396 | 45.701 | 9 |
| 9 | 51 | 16 | 256 | 1000 | 717/63 | 1032.50 | 1080.82 | 1128.57 | 31.236 | 34.746 | 16 |
| 11 | 63 | 20 | 400 | 1000 | 1131/84 | 1621.83 | 1688.73 | 1757.85 | 34.881 | 55.476 | 19.4 |
| 13 | 98 | 27 | 253 | 1000 | 1034/3 | 1786.37 | 1818.89 | 1872.01 | 26.135 | 20.263 | 28.0 |
| 15 | 124 | 36 | 397 | 1000 | 1667/08 | 2892.85 | 2938.43 | 2987.39 | 32.63 | 41.011 | 36.4 |
| 17 | 98 | 23 | 241 | 200 | 795/33 | 1738.09 | 1780.32 | 1823.69 | 26.223 | 20.235 | 23.4 |
| 19 | 153 | 33 | 361 | 200 | 1538/2 | 3619.93 | 3694.05 | 3776.65 | 47.559 | 26.072 | 34.7 |
| | | | | | | | | | | | |
| ρ = 50% | | | | | | | | | | | |
| **index** | **clusters** | **vehicles** | **vertices** | **Q** | **BKS** | **best** | **average** | **worst** | **variance** | **avg_time** | **avg vehicles** |
| 11 | 37 | 18 | 400 | 1000 | 1101/51 | 1220.54 | 1285.73 | 1323.74 | 26.892 | 140.71 | 19.0 |
| 12 | 40 | 20 | 484 | 1000 | 1311/91 | 1468.56 | 1512.06 | 1538.39 | 20.35 | 160.09 | 20.0 |
| 13 | 58 | 28 | 253 | 1000 | 1053/47 | 1283.13 | 1313.73 | 1334.24 | 18.24 | 27.601 | 29.0 |
| 14 | 66 | 32 | 321 | 1000 | 1342/7 | 1690.19 | 1714.35 | 1738.92 | 17.303 | 34.624 | 33.0 |
| 15 | 73 | 36 | 397 | 1000 | 1657/22 | 2056.48 | 2088.56 | 2146.57 | 28.903 | 37.968 | 37.0 |
| 16 | 80 | 39 | 481 | 1000 | 2003/1 | 2483.53 | 2536.66 | 2574.27 | 31.63 | 41.825 | 40.0 |
| 17 | 47 | 24 | 241 | 200 | 881/66 | 1081.14 | 1101.15 | 1117.64 | 11.417 | 25.301 | 24.0 |
| 18 | 59 | 30 | 301 | 200 | 1199/12 | 1535.39 | 1562.26 | 1586.88 | 15.86 | 30.215 | 30.0 |
| 19 | 69 | 35 | 361 | 200 | 1612/33 | 2049.17 | 2102.60 | 2132.47 | 25.443 | 34.669 | 35.0 |
| 20 | 81 | 41 | 421 | 200 | 2278/64 | 2856.31 | 2941.40 | 2993.29 | 37.502 | 39.778 | 41.0 |
| | | | | | | | | | | | |
| ρ = 75% | | | | | | | | | | | |
| **index** | **clusters** | **vehicles** | **vertices** | **Q** | **BKS** | **best** | **average** | **worst** | **variance** | **avg_time** | **avg vehicles** |
| 2 | 13 | 13 | 321 | 700 | 10204/3 | 10520.98 | 10520.98 | 10520.98 | 0.0 | 31.146 | 13.0 |
| 4 | 13 | 13 | 481 | 1000 | 17077/5 | 17626.74 | 17626.74 | 17626.74 | 0.0 | 48.007 | 13.0 |
| 6 | 9 | 9 | 281 | 900 | 11452/0 | 11847.54 | 11847.54 | 11847.54 | 0.0 | 21.714 | 9.0 |
| 8 | 14 | 13 | 441 | 900 | 13882/23 | 14485.70 | 14516.30 | 14644.08 | 57.622 | 42.148 | 13.0 |
| 10 | 22 | 21 | 324 | 1000 | 1000/507 | 1028.73 | 1028.73 | 1028.73 | 0.0 | 31.6839 | 22.0 |
| 12 | 27 | 26 | 484 | 1000 | 1475/679 | 1502.48 | 1504.87 | 1518.18 | 4.54 | 20.698 | 26.0 |
| 14 | 42 | 41 | 321 | 1000 | 1520/546 | 1540.77 | 1552.04 | 1562.49 | 8.058 | 19.034 | 41.0 |
| 16 | 51 | 51 | 481 | 1000 | 2265/537 | 2308.49 | 2308.49 | 2308.49 | 0.0 | 22.7072 | 51.0 |
| 18 | 38 | 37 | 301 | 200 | 1392/153 | 1422.93 | 1422.93 | 1422.93 | 0.0 | 22.992 | 39.0 |
| 20 | 53 | 52 | 421 | 200 | 2502/34 | 2599.88 | 2599.88 | 2599.88 | 0.0 | 35.1147 | 53.0 |
| | | | | | | | | | | | |

| ρ = 100% | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **index** | **clusters** | **vehicles** | **vertices** | **Q** | **BKS** | **best** | **average** | **worst** | **variance** | **avg_time** | **avg vehicles** |
| 1 | 9 | 9 | 241 | 550 | 6293/036 | 6401.09 | 6401.09 | 6401.09 | 0.0 | 28.533 | 9.0 |
| 2 | 10 | 10 | 321 | 700 | 9879/586 | 10187.39 | 10187.39 | 10187.39 | 0.0 | 46.259 | 10.0 |
| 4 | 10 | 10 | 481 | 1000 | 16130/39 | 16664.14 | 16664.14 | 16664.14 | 0.0 | 75.881 | 10.0 |
| 5 | 5 | 5 | 201 | 900 | 8394/111 | 8679.34 | 8679.34 | 8679.34 | 0.0 | 49.568 | 5.0 |
| 7 | 8 | 8 | 361 | 900 | 11346/11 | 11705.95 | 11705.95 | 11705.95 | 0.0 | 49.408 | 8.0 |
| 8 | 10 | 10 | 441 | 900 | 13188/94 | 13572.42 | 13572.42 | 13572.42 | 0.0 | 48.076 | 10.0 |
| 10 | 16 | 16 | 324 | 1000 | 837/516 | 860.64 | 860.64 | 860.64 | 0.0 | 25.972 | 16.0 |
| 11 | 18 | 18 | 400 | 1000 | 1054/133 | 1091.88 | 1091.88 | 1091.88 | 0.0 | 42.810 | 18.0 |
| 19 | 34 | 34 | 361 | 200 | 1667/454 | 1696.12 | 1696.12 | 1696.12 | 0.0 | 25.765 | 34.0 |
| 20 | 39 | 39 | 421 | 200 | 2128/597 | 2158.55 | 2158.55 | 2158.55 | 0.0 | 28.537 | 39.0 |

| Instance name | BKS | | best | average | worst | variance | avg_time | avg vehicles |
|---|---|---|---|---|---|---|---|---|
| **e-n10-c2.map** | 2016/57 | | 2016.57 | 2016.57 | 2016.57 | 0.0 | 0.4629 | 2.0 |
| **a-n15-c4.map** | 1947/3 | | 1961.98 | 1961.98 | 1961.98 | 0.0 | 0.3818 | 3.0 |
| **b-n15-c4.map** | 2602/56 | | 2684.84 | 2933.94 | 3325.27 | 305.288 | 0.0462 | 2.4 |
| **a-n20-c5.map** | 2759/13 | | 2788.79 | 2788.79 | 2788.79 | 0.0 | 0.4808 | 3.0 |
| **c-n20-c5.map** | 3028/83 | | 3038.93 | 3118.86 | 3438.59 | 159.862 | 0.4508 | 4.0 |
| **d-n20-c5.map** | 2239/09 | | 2239.09 | 2244.23 | 2290.48 | 15.419 | 0.5597 | 3.0 |
| **e-n20-c5.map** | 3343/34 | | 3343.34 | 3343.34 | 3343.34 | 0.0 | 0.4799 | 4.0 |
| **b-n30-c6.map** | 3116/84 | | 3285.15 | 3361.59 | 3514.92 | 83.185 | 0.8846 | 3.0 |

- GA One-minute run Results:

| **ρ = 10%** | | | | |
|---|---|---|---|---|
| **file name** | **BKS** | **vehicles** | **best** | **vehicles** |
| **kelly01.ccvrp** | 5759/25 | 9 | 16751/19 | 9 |
| **kelly02.ccvrp** | 9247/92 | 10 | 22744/33 | 10 |
| **kelly03.ccvrp** | 12904/6 | 10 | 28874/64 | 10 |
| **kelly04.ccvrp** | 17810/4 | 10 | 41207/75 | 11 |
| **kelly05.ccvrp** | 8960/31 | 5 | 16768/86 | 5 |
| **kelly06.ccvrp** | 10976/5 | 7 | 22215/53 | 7 |
| **kelly07.ccvrp** | 12485/8 | 9 | 28314/09 | 9 |
| **kelly08.ccvrp** | 13331/2 | 11 | 33422/68 | 11 |
| **kelly09.ccvrp** | 710/64 | 15 | 1065/26 | 16 |

| | | | | |
|---|---|---|---|---|
| **kelly10.ccvrp** | 908/89 | 18 | 1287/338 | 19 |
| | | | | |
| | | | | |
| **ρ = 25%** | | | | |
| **file name** | **BKS** | **vehicles** | **best** | **vehicles** |
| **kelly01.ccvrp** | 6051/04 | 10 | 9409/454 | 10 |
| **kelly03.ccvrp** | 13692/6 | 10 | 18045/11 | 10 |
| **kelly05.ccvrp** | 9340/7 | 5 | 11209/1 | 5 |
| **kelly07.ccvrp** | 12348/1 | 9 | 16518/23 | 9 |
| **kelly09.ccvrp** | 717/63 | 16 | 1054/309 | 16 |
| **kelly11.ccvrp** | 1131/84 | 20 | 1664/447 | 20 |
| **kelly13.ccvrp** | 1034/3 | 27 | 1772/769 | 30 |
| **kelly15.ccvrp** | 1667/08 | 36 | 2851/527 | 38 |
| **kelly17.ccvrp** | 795/33 | 23 | 1635/427 | 23 |
| **kelly19.ccvrp** | 1538/2 | 33 | 3678/73 | 35 |
| | | | | |
| | | | | |
| **ρ = 50%** | | | | |
| **file name** | **BKS** | **vehicles** | **best** | **vehicles** |
| **kelly11.ccvrp** | 1101/51 | 18 | 1263/157 | 19 |
| **kelly12.ccvrp** | 1311/92 | 20 | 1498/865 | 20 |
| **kelly13.ccvrp** | 1053/47 | 28 | 1273/728 | 29 |
| **kelly14.ccvrp** | 1342/7 | 32 | 1673/094 | 33 |
| **kelly15.ccvrp** | 1657/22 | 36 | 2041/113 | 37 |
| **kelly16.ccvrp** | 2003/1 | 39 | 2516/534 | 40 |
| **kelly17.ccvrp** | 881/66 | 24 | 1088/141 | 24 |
| **kelly18.ccvrp** | 1199/12 | 30 | 1533/102 | 30 |
| **kelly19.ccvrp** | 1612/33 | 35 | 2030/046 | 35 |
| **kelly20.ccvrp** | 2278/64 | 41 | 2939/032 | 41 |
| | | | | |
| | | | | |
| **ρ = 75%** | | | | |
| **file name** | **BKS** | **vehicles** | **best** | **vehicles** |
| **kelly02.ccvrp** | 10204/3 | 13 | 10520/98 | 13 |
| **kelly04.ccvrp** | 17077/6 | 13 | 17626/74 | 13 |
| **kelly06.ccvrp** | 11452 | 9 | 11847/54 | 9 |
| **kelly08.ccvrp** | 13882/2 | 13 | 14492/69 | 13 |
| **kelly10.ccvrp** | 1000/51 | 21 | 1023/978 | 21 |
| **kelly12.ccvrp** | 1475/68 | 26 | 1502/488 | 26 |
| **kelly14.ccvrp** | 1520/55 | 41 | 1540/777 | 41 |
| **kelly16.ccvrp** | 2265/54 | 51 | 2308/495 | 51 |
| **kelly18.ccvrp** | 1392/15 | 37 | 1401/552 | 37 |
| **kelly20.ccvrp** | 2502/34 | 52 | 2572/521 | 52 |
| | | | | |
| | | | | |

| ρ = 100% | | | | |
|---|---|---|---|---|
| **file name** | **BKS** | **vehicles** | **best** | **vehicles** |
| **kelly01.ccvrp** | 6293/04 | 9 | 6401/095 | 9 |
| **kelly02.ccvrp** | 9879/59 | 10 | 10187/4 | 10 |
| **kelly04.ccvrp** | 16130/4 | 10 | 16664/14 | 10 |
| **kelly05.ccvrp** | 8394/11 | 5 | 8679/348 | 5 |
| **kelly07.ccvrp** | 11346/1 | 8 | 11705/95 | 8 |
| **kelly08.ccvrp** | 13188/9 | 10 | 13572/43 | 10 |
| **kelly10.ccvrp** | 837/516 | 16 | 860/6491 | 16 |
| **kelly11.ccvrp** | 1054/13 | 18 | 1091/883 | 18 |
| **kelly19.ccvrp** | 1667/45 | 34 | 1696/128 | 34 |
| **kelly20.ccvrp** | 2128/6 | 39 | 2158/556 | 39 |

| Instance name | BKS | | best | avg vehicles |
|---|---|---|---|---|
| **e-n10-c2.map** | 2016/57 | | 2016.57 | 2.0 |
| **a-n15-c4.map** | 1947/3 | | 1961.98 | 3.0 |
| **b-n15-c4.map** | 2602/56 | | 2684.84 | 2.4 |
| **a-n20-c5.map** | 2759/13 | | 2788.79 | 3.0 |
| **c-n20-c5.map** | 3028/83 | | 3038.93 | 4.0 |
| **d-n20-c5.map** | 2239/09 | | 2239.09 | 3.0 |
| **e-n20-c5.map** | 3343/34 | | 3343.34 | 4.0 |
| **b-n30-c6.map** | 3116/84 | | 3285.15 | 3.0 |

- GA Algorithm analysis:

Algorithm diversity seems to be low per some instances because of deleting non feasible solutions that could be create after Xover or mutation. Replacing parent instead of child when child isn't feasible will also decrease population diversity.
After analysis for example in execution of ***a-n15-c4.map*** instance 30% of population was not feasible after Xover or mutation. As dimension of problem decrease diversity of population will also decrease.

Algorithm representation is another important factor of diversity.
The representation that described in the related paper cause many-to-one mapping between phenotype and genotype in the way that for example Ch1 = (6 7 0 1 2 3 0 4 5) and Ch2 = (1 2 3 0 6 7 0 4 5) have same fitness value cause they both have same global routes but the order of them is different.

Even one chromosome itself could be mapped to many members of phenotype for because representation shows only sequence of clusters to be seen and doesn't specifies node orders to be visiting in a cluster.

In compare results were reasonably close to best known solution.
But algorithm configuration that used in this report has significantly low number of generation (10 generation) in compare with paper configuration (200 generation) and it causes less chance of getting out of local optima.

Mutation rate (20%) will shows its role of preserving diversity could not be achieved by small number of generations.

- Global and local subproblem with ACS:

  - Each ant solution form would be same as GA representation.
  - M ants would place randomly on graph clusters.
  - Local subproblem is the same as describe in GA and use in determining the global best solution in the colony.

```
Initialize
Loop /* at this level each loop is called an iteration */
   Each ant is positioned on a starting node
   Loop /* at this level each loop is called a step */
     Each ant applies a state transition rule to incrementally build a solution
     and a local pheromone updating rule
   Until all ants have built a complete solution
   A global pheromone updating rule is applied
Until End_condition
```

- **ACS state transition rule**

  an ant positioned on node *r* chooses the city *s* to move to by applying the rule given:

$$s = \begin{cases} \arg\max_{u \in J_k(r)} \left\{ [\tau(r,u)] \cdot [\eta(r,u)]^{\beta} \right\} & \text{if } q \le q_0 \quad \text{(exploitation)} \\ S & \text{otherwise} \quad \text{(biased exploration)} \end{cases}$$

  we S set as follow:

$$p_k(r,s) = \begin{cases} \dfrac{[\tau(r,s)] \cdot [\eta(r,s)]^{\beta}}{\sum_{u \in J_k(r)} [\tau(r,u)] \cdot [\eta(r,u)]^{\beta}} & \text{if } s \in J_k(r) \\ 0 & \text{otherwise} \end{cases}$$

- **ACS local updating rule**

$$\tau(r,s) \leftarrow (1-\rho)\cdot\tau(r,s) + \rho\cdot\Delta\tau(r,s)$$

where $0<\rho<1$ is a parameter.

(i) we set $\Delta\tau(r,s) = \tau_0$, where $\tau_0$ is the initial pheromone level
(ii) (ii) we set $\Delta\tau(r,s) = 0$.

- **ACS global updating rule**

$$\tau(r,s) \leftarrow (1-\alpha)\cdot\tau(r,s) + \alpha\cdot\Delta\tau(r,s)$$

$$\text{where } \Delta\tau(r,s) = \begin{cases} (L_{gb})^{-1} & \text{if } (r,s) \in \text{global - best - tour} \\ 0 & \text{otherwise} \end{cases}$$

$0<\alpha<1$ is the pheromone decay parameter, and $L_{gb}$ is the length of the globally best tour from the beginning of the trial

- **ACS parameter settings**

$\beta=2$, $q_0=0.9$, $\alpha=\rho=0.1$, $\tau_0=(n\cdot L_{nn})^{-1}$, where $L_{nn}$ is the tour length produced by the nearest neighbor heuristic

values of the parameters were largely independent of the problem, except for $\tau_0$ for which, as we said, $\tau_0=(n\cdot L_{nn})^{-1}$. The number of ants used is $m=10$ (this choice is explained in Section IV.B). Regarding their initial positioning, ants are placed randomly, with at most one ant in each city.

- ACS Implementation
  - State transition:

```
# exploitation
if q < Q0:
        s = candidates[args.index(max(args))]
# exploration
else:
        p = [arg/sum(args) for arg in args]

        # roulette wheel
        s = candidates[rouletteWheel(p)]
```

- Local update:

  Update the pheromone path in both ways.

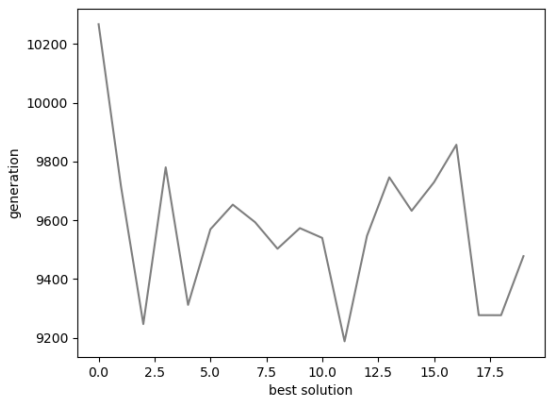
```
self.T[r][s] = (1-RHO) * self.T[r][s] + RHO * self.T0
self.T[s][r] = (1-RHO) * self.T[s][r] + RHO * self.T0
```

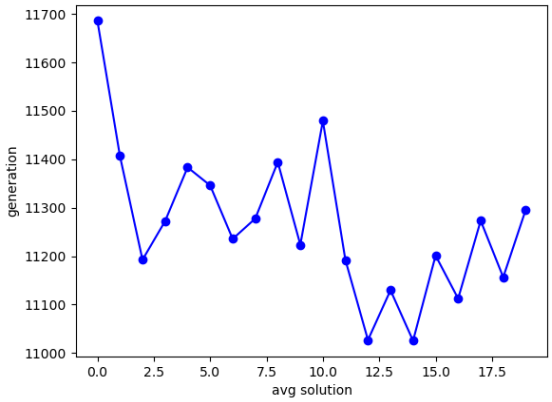
- Global update:
  If edge(i,j) is in the best solution.set the delta as 1/ $L_{gb}$.
  And after that update edge path in both ways.

```
if i in bestSol and j in bestSol and + \
            abs(bestSol.index(i) - bestSol.index(j)) == 1:

        deltaT = 1/self.best[1]

self.T[i][j] = (1-ALPHA) * self.T[i][j] + \
   (ALPHA * deltaT)

self.T[j][j] = (1-ALPHA) * self.T[j][i] + \
   (ALPHA * deltaT)
```

- ACS convergence process:

Best and average solution per iteration for instance 0.1%_ kelly01




It shows that overall, the pheromone trail leads the search to better solution during algorithm. Although exploration manner search could happen to.

- ACS Results:

iterations = 10
ALPHA = RHO = 0.2
Beta = 2
Q0 = 0.9
T0 = 0.00005 (value is based on some search results)
Ants_num = max(Clusters_num/10 , 2)

| ρ = 10% | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **index** | **clusters** | **vehicles** | **vertices** | **Q** | **BKS** | **Best** | **average** | **worst** | **variance** | **avg_time** |
| 1 | 120 | 9 | 241 | 550 | 5759/25 | 9311/123 | 9524/57 | 10076/36 | 230/952 | 18.097 |
| 2 | 101 | 10 | 321 | 700 | 9247/92 | 14608/26 | 15142/85 | 15540/95 | 279/1 | 26.299 |
| 3 | 96 | 10 | 401 | 900 | 12904/6 | 19725/90 | 20017/18 | 20326/97 | 190/524 | 41.711 |
| 4 | 104 | 10 | 481 | 1000 | 17810/4 | 25823/08 | 26967/54 | 27998/27 | 678/783 | 64.355 |
| 5 | 49 | 5 | 201 | 900 | 8960/31 | 13030/843 | 13436/76 | 13761/94 | 262/992 | 9.5907 |
| 6 | 67 | 7 | 281 | 900 | 10976/5 | 15562/00 | 16137/14 | 16708/44 | 361/586 | 18.995 |
| 7 | 88 | 9 | 361 | 900 | 12485/8 | 19062/93 | 19561/38 | 20021/09 | 332/632 | 35.513 |
| 8 | 108 | 11 | 441 | 900 | 13331/2 | 20047/18 | 21490/08 | 22168/08 | 620/596 | 50.363 |
| 9 | 51 | 15 | 256 | 1000 | 710/64 | 787/7316 | 799/3658 | 809/9543 | 6/832 | 4.0530 |
| 10 | 56 | 18 | 324 | 1000 | 908/89 | 989/7373 | 996/7994 | 1005/142 | 5/058 | 6.0530 |
| | | | | | | | | | | |
| | | | | | | | | | | |
| ρ = 25% | | | | | | | | | | |
| **index** | **clusters** | **Vehicles** | **vertices** | **Q** | **BKS** | **best** | **average** | **Worst** | **variance** | **avg_time** |
| 1 | 40 | 10 | 241 | 550 | 6051/04 | 7152/93 | 7324/41 | 7437/47 | 84/994 | 3.6489 |
| 3 | 38 | 10 | 401 | 900 | 13692/6 | 15311/51 | 15637/0 | 16007/93 | 199/748 | 9.7655 |
| 5 | 19 | 5 | 201 | 900 | 9340/7 | 10379/19 | 10486/2 | 10548/73 | 47/584 | 4.6584 |
| 7 | 34 | 9 | 361 | 900 | 12348/1 | 13918/10 | 14014/0 | 14080/33 | 58/941 | 9.4245 |
| 9 | 51 | 16 | 256 | 1000 | 717/63 | 767/8084 | 786/112 | 804/6502 | 10/616 | 3.5788 |
| 11 | 63 | 20 | 400 | 1000 | 1131/84 | 1221/950 | 1233/58 | 1246/456 | 9/698 | 8.7118 |
| 13 | 98 | 27 | 253 | 1000 | 1034/3 | 1165/701 | 1209/72 | 1233/166 | 20/108 | 5.5740 |
| 15 | 124 | 36 | 397 | 1000 | 1667/08 | 1863/173 | 1897/73 | 1923/843 | 15/59 | 13.096 |
| 17 | 98 | 23 | 241 | 200 | 795/33 | 1040/013 | 1070/78 | 1100/964 | 19/253 | 5.7340 |
| 19 | 153 | 33 | 361 | 200 | 1538/2 | 2026/525 | 2091/53 | 2131/029 | 25/555 | 19.384 |
| | | | | | | | | | | |
| ρ = 50% | | | | | | | | | | |
| **index** | **clusters** | **vehicles** | **vertices** | **Q** | **BKS** | **best** | **average** | **worst** | **variance** | **avg_time** |
| 11 | 37 | 18 | 400 | 1000 | 1101/51 | 1132/953 | 1152/95 | 1166/500 | 10/472 | 4.5660 |
| 12 | 40 | 20 | 484 | 1000 | 1311/91 | 1352/895 | 1355/98 | 1359/790 | 1/995 | 7.4104 |
| 13 | 58 | 28 | 253 | 1000 | 1053/47 | 1069/599 | 1080/11 | 1093/557 | 7/775 | 1.9484 |
| 14 | 66 | 32 | 321 | 1000 | 1342/7 | 1370/867 | 1390/00 | 1404/544 | 8/793 | 2.8605 |
| 15 | 73 | 36 | 397 | 1000 | 1657/22 | 1692/704 | 1702/33 | 1710/727 | 6/215 | 4.6001 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 16 | 80 | 39 | 481 | 1000 | 2003/1 | 2048/210 | 2067/94 | 2088/451 | 11/053 | 7.1417 |
| 17 | 47 | 24 | 241 | 200 | 881/66 | 899/0229 | 909/308 | 921/2504 | 5/69 | 1.2698 |
| 18 | 59 | 30 | 301 | 200 | 1199/12 | 1245/925 | 1255/40 | 1264/584 | 6/151 | 2.1820 |
| 19 | 69 | 35 | 361 | 200 | 1612/33 | 1666/086 | 1673/89 | 1679/125 | 4/428 | 3.0638 |
| 20 | 81 | 41 | 421 | 200 | 2278/64 | 2345/878 | 2362/97 | 2378/723 | 10/285 | 4.6380 |
| | | | | | | | | | | |
| ρ = 75% | | | | | | | | | | |
| **index** | **clusters** | **vehicles** | **vertices** | **Q** | **BKS** | **best** | **average** | **worst** | **variance** | **avg_time** |
| 2 | 13 | 13 | 321 | 700 | 10204/3 | 10204/3 | 10520/983 | 10520/983 | 10520/983 | 0 |
| 4 | 13 | 13 | 481 | 1000 | 17077/5 | 17077/59 | 17626/743 | 17626/743 | 17626/743 | 0 |
| 6 | 9 | 9 | 281 | 900 | 11452/0 | 11452/01 | 11847/545 | 11847/545 | 11847/545 | 0 |
| 8 | 14 | 13 | 441 | 900 | 13882/23 | 13882/23 | 14485/703 | 14491/99 | 14492/689 | 2/096 |
| 10 | 22 | 21 | 324 | 1000 | 1000/507 | 1000/507 | 1023/9781 | 1027/308 | 1028/7352 | 2/18 |
| 12 | 27 | 26 | 484 | 1000 | 1475/679 | 1475/679 | 1502/488 | 1517/308 | 1542/4236 | 16/796 |
| 14 | 42 | 41 | 321 | 1000 | 1520/546 | 1520/546 | 1540/7771 | 1542/4697 | 1546/4192 | 2/586 |
| 16 | 51 | 51 | 481 | 1000 | 2265/537 | 2265/537 | 2308/4947 | 2308/4947 | 2308/4947 | 0 |
| 18 | 38 | 37 | 301 | 200 | 1392/153 | 1392/153 | 1401/5521 | 1403/69 | 1422/931 | 6/414 |
| 20 | 53 | 52 | 421 | 200 | 2502/34 | 2502/34 | 2586/0475 | 2594/3502 | 2599/8854 | 6/779 |
| | | | | | | | | | | |
| | | | | | | | | | | |
| ρ = 100% | | | | | | | | | | |
| **index** | **clusters** | **vehicles** | **vertices** | **Q** | **BKS** | **best** | **average** | **worst** | **variance** | **avg_time** |
| 1 | 9 | 9 | 241 | 550 | 6293/036 | 6401/0949 | 6401/0949 | 6401/0949 | 0 | 1.9407 |
| 2 | 10 | 10 | 321 | 700 | 9879/586 | 10187/397 | 10187/397 | 10187/397 | 0 | 4.2388 |
| 4 | 10 | 10 | 481 | 1000 | 16130/39 | 16664/141 | 16664/141 | 16664/141 | 0 | 9.1544 |
| 5 | 5 | 5 | 201 | 900 | 8394/111 | 8679/3478 | 8679/3478 | 8679/3478 | 0 | 3.2554 |
| 7 | 8 | 8 | 361 | 900 | 11346/11 | 11705/953 | 11705/953 | 11705/953 | 0 | 5.3053 |
| 8 | 10 | 10 | 441 | 900 | 13188/94 | 13572/426 | 13572/426 | 13572/426 | 0 | 7.3177 |
| 10 | 16 | 16 | 324 | 1000 | 837/516 | 860/64914 | 860/64914 | 860/64914 | 0 | 2.9047 |
| 11 | 18 | 18 | 400 | 1000 | 1054/133 | 1091/8828 | 1091/8828 | 1091/8828 | 0 | 3.9081 |
| 19 | 34 | 34 | 361 | 200 | 1667/454 | 1696/1283 | 1696/1283 | 1696/1283 | 0 | 1.4732 |
| 20 | 39 | 39 | 421 | 200 | 2128/597 | 2158/5557 | 2158/5557 | 2158/5557 | 0 | 1.9021 |

| Instance name | BKS | | best | average | worst | variance | avg_time |
|---|---|---|---|---|---|---|---|
| **e-n10-c2.map** | 2016/57 | | 2016.571 | 2016.571 | 2016.571 | 0 | 0.0123 |
| **a-n15-c4.map** | 1947/3 | | 1961/9887 | 1961/9887 | 1961/9887 | 0 | 0.0153 |
| **b-n15-c4.map** | 2602/56 | | 2684.840 | 2684.840 | 2684.840 | 0 | 0.0206 |
| **a-n20-c5.map** | 2759/13 | | 2788.78 | 2788.78 | 2788.78 | 0 | 0.0283 |
| **c-n20-c5.map** | 3028/83 | | 3038.936 | 3038.936 | 3038.936 | 0 | 0.0224 |
| **d-n20-c5.map** | 2239/09 | | 2239.09 | 2239.09 | 2239.09 | 0 | 0.0288 |
| **e-n20-c5.map** | 3343/34 | | 3343.3399 | 3343.3399 | 3343.3399 | 0 | 0.0263 |
| **b-n30-c6.map** | 3116/84 | | 3125.264 | 3125.264 | 3125.264 | 0 | 0.0550 |

- ACS One-minute run Results:

| ρ = 10% | | | |
|---|---|---|---|
| **file name** | **BKS** | | **best** |
| **kelly01.ccvrp** | 5759/25 | | 9580/8395 |
| **kelly02.ccvrp** | 9247/92 | | 15034/384 |
| **kelly03.ccvrp** | 12904/6 | | 19805/949 |
| **kelly04.ccvrp** | 17810/4 | | 27206/636 |
| **kelly05.ccvrp** | 8960/31 | | 12710/506 |
| **kelly06.ccvrp** | 10976/5 | | 16211/253 |
| **kelly07.ccvrp** | 12485/8 | | 19635/97 |
| **kelly08.ccvrp** | 13331/2 | | 20358/169 |
| **kelly09.ccvrp** | 710/64 | | 796/07424 |
| **kelly10.ccvrp** | 908/89 | | 978/17974 |
| | | | |
| | | | |
| **ρ = 25%** | | | |
| **file name** | **BKS** | | **best** |
| **kelly01.ccvrp** | 6051/04 | | 7162/3492 |
| **kelly03.ccvrp** | 13692/6 | | 15173/437 |
| **kelly05.ccvrp** | 9340/7 | | 10379/19 |
| **kelly07.ccvrp** | 12348/1 | | 13591/408 |
| **kelly09.ccvrp** | 717/63 | | 776/39522 |
| **kelly11.ccvrp** | 1131/84 | | 1229/9866 |
| **kelly13.ccvrp** | 1034/3 | | 1207/9695 |
| **kelly15.ccvrp** | 1667/08 | | 1872/9098 |
| **kelly17.ccvrp** | 795/33 | | 1027/5664 |
| **kelly19.ccvrp** | 1538/2 | | 2054/3725 |
| | | | |
| | | | |
| **ρ = 50%** | | | |
| **file name** | **BKS** | | **best** |
| **kelly11.ccvrp** | 1101/51 | | 1136/72 |
| **kelly12.ccvrp** | 1311/92 | | 1354/43 |
| **kelly13.ccvrp** | 1053/47 | | 1067/8632 |
| **kelly14.ccvrp** | 1342/7 | | 1374/37 |
| **kelly15.ccvrp** | 1657/22 | | 1692/8696 |
| **kelly16.ccvrp** | 2003/1 | | 2044/8591 |
| **kelly17.ccvrp** | 881/66 | | 901/4698 |
| **kelly18.ccvrp** | 1199/12 | | 1233/3751 |
| **kelly19.ccvrp** | 1612/33 | | 1654/4038 |
| **kelly20.ccvrp** | 2278/64 | | 2353/2575 |

| | | | |
|---|---|---|---|
| | | | |
| **ρ = 75%** | | | |
| **file name** | **BKS** | | **best** |
| **kelly02.ccvrp** | 10204/3 | | 10520/983 |
| **kelly04.ccvrp** | 17077/6 | | 17626/743 |
| **kelly06.ccvrp** | 11452 | | 11847/545 |
| **kelly08.ccvrp** | 13882/2 | | 14492/689 |
| **kelly10.ccvrp** | 1000/51 | | 1023/9781 |
| **kelly12.ccvrp** | 1475/68 | | 1502/488 |
| **kelly14.ccvrp** | 1520/55 | | 1540/7771 |
| **kelly16.ccvrp** | 2265/54 | | 2308/4947 |
| **kelly18.ccvrp** | 1392/15 | | 1401/5521 |
| **kelly20.ccvrp** | 2502/34 | | 2572/5211 |
| | | | |
| | | | |
| **ρ = 100%** | | | |
| **file name** | **BKS** | | **best** |
| **kelly01.ccvrp** | 6293/04 | | 6401/0949 |
| **kelly02.ccvrp** | 9879/59 | | 10187/397 |
| **kelly04.ccvrp** | 16130/4 | | 16664/141 |
| **kelly05.ccvrp** | 8394/11 | | 8679/3478 |
| **kelly07.ccvrp** | 11346/1 | | 11705/953 |
| **kelly08.ccvrp** | 13188/9 | | 13572/426 |
| **kelly10.ccvrp** | 837/516 | | 860/64914 |
| **kelly11.ccvrp** | 1054/13 | | 1091/8828 |
| **kelly19.ccvrp** | 1667/45 | | 1696/1283 |
| **kelly20.ccvrp** | 2128/6 | | 2158/5557 |

| Instance name | BKS | | best |
|---|---|---|---|
| **e-n10-c2.map** | 2016/57 | | 2016.571 |
| **a-n15-c4.map** | 1947/3 | | 1961/9887 |
| **b-n15-c4.map** | 2602/56 | | 2684.840 |
| **a-n20-c5.map** | 2759/13 | | 2788.78 |
| **c-n20-c5.map** | 3028/83 | | 3038.936 |
| **d-n20-c5.map** | 2239/09 | | 2239.09 |
| **e-n20-c5.map** | 3343/34 | | 3343.3399 |
| **b-n30-c6.map** | 3116/84 | | 3125.264 |

- Comparing GA and ACS Results:

| ρ = 10% | | | | | | |
|---|---|---|---|---|---|---|
| file name | BKS | | GA best | GA avg time | ACS best | ACS avg time |
| kelly08.ccvrp | 13331/2 | | 33422/68 | 84.987 | 20047/18 | 50.363 |
| kelly10.ccvrp | 908/89 | | 1287/338 | 45.193 | 989/7373 | 6.0530 |
| | | | | | | |
| ρ = 25% | | | | | | |
| file name | BKS | | GA best | GA avg time | ACS best | ACS avg time |
| kelly11.ccvrp | 1131/84 | | 1664/447 | 55.476 | 1221/950 | 8.7118 |
| kelly19.ccvrp | 1538/2 | | 3678/73 | 26.072 | 2026/525 | 19.384 |
| | | | | | | |
| ρ = 50% | | | | | | |
| file name | BKS | | GA best | GA avg time | ACS best | ACS avg time |
| kelly19.ccvrp | 1612/33 | | 2030/046 | 34.669 | 1666/086 | 4/428 |
| kelly20.ccvrp | 2278/64 | | 2939/032 | 39.778 | 2345/878 | 10/285 |
| | | | | | | |
| ρ = 75% | | | | | | |
| file name | BKS | | GA best | GA avg time | ACS best | ACS avg time |
| kelly10.ccvrp | 1000/51 | | 1023/978 | 31.6839 | 1000/507 | 2/18 |
| kelly20.ccvrp | 2502/34 | | 2572/521 | 35.1147 | 2502/34 | 6/779 |
| | | | | | | |
| ρ = 100% | | | | | | |
| file name | BKS | | GA best | GA avg time | ACS best | ACS avg time |
| kelly08.ccvrp | 13188/9 | | 13572/43 | 48.076 | 13572/426 | 7.3177 |
| kelly19.ccvrp | 1667/45 | | 1696/128 | 25.765 | 1696/1283 | 1.4732 |

As result shows ACS perform better in solution quality and algorithm time but the initial heuristic version of computing $\tau_0 = (n \cdot L_{nn})^{-1}$ take much more time than GA (which I skipped that because of lacking of time), so time comparing of algorithms would not be much fair in this case.

In compare with GA we have both exploitation and exploration mechanism in ACS but instead of creating new solutions from parents in GA and select the better solutions based on pressure degree of the xOver and mutation result ,we choose constructive manner that creating the solution step by step based on some heuristic and collective information of colony(pheromone).

So, it seems that in this way search would be more targeted in compare with GA.