Models, relaxations and exact approaches for the capacitated vehicle routing problem

Final Report of Network Models

By Sara Nasiriyani

April 15, 2020

Contents

1 Introduction

In today's modern world filled with e-commerce and online shoppings, the urge for an excellent logistics and transportation system is sensed more than ever. Taking the issues of global warming and traffic of the roads into account, an optimized transportation system with less fuel consumption and shorter delivery paths will benefit humans effectively. That is why the concept of mathematical modelling in transportation is one of the essential applications of optimization.

Lots of approaches have been adopted to optimize the problem of transportation between companies and customers in many years, one of which is the Capacitated Vehicle Routing problems to find an optimal or near optimal delivery paths. The annual approximate cost of distribution for united states is 400 billion dollars [1] and in 2016, aggregate final expenditures on transportation (including insurance) amounted to 179.5 billion dollars in Canada [1] . Thus, any savings generated by improvements in route scheduling would be significant.

The vehicle routing problem (VRP) is a combinatorial optimization and integer programming problem. It is a more generalized version of the well-known traveling salesman problem. TSP is basically a VRP while the number of trucks is only one and we know for sure that the capacity of vehicle is greater or equal to the demand of the customers. Therefor, any solution method applied to the famous TSP problem could be adopted for this approach with slight changes.

However, we should bear in mind that modelling issues like CVRP have its own complications. Vehicle routing mathematical modeling could be one of the most complex problems to solve as they have lots of specifications and they stand under the category of NP-hard problems[2] which make the solution approaches way complicated than a simple programming model.

This project considers a Capacitated Vehicle Routing problem and its solution approaches. CVR problems are basically divided in to two main categories:

- Symmetric Capacitated Vehicle Routing (SCVR)
- Asymmetric Capacitated Vehicle Routing (ACVR)

Capacitated vehicle routing problems are not novel topics and lots of different modeling and approaches have been introduced over the past decades to tackle them. The overall objective of the CVRP modeling and studies is to determine the optimal routes used by a fleet of vehicles, based at one or more depots, to serve a set of customers. Many additional requirements and operational constraints are imposed on the route construction in practical applications of the VRP.

Lots of real world application could be considered as a CVRP. Any distribution center with more than one vehicle fleet can apply this optimization to its own situation, although the application of the case being studied in this study is pretty limited as only for few trucks and customers these problems are going to result in an exact optimum solution but there

[1]Comprehensive report of Transportation in Canada 2016

[2]NP-hardness (non-deterministic polynomial-time hardness) is, in computational complexity theory, the defining property of a class of problems that are informally "at least as hard as the hardest problems in NP"

are so many other ways to calculate the near optimum solution as presented in this paper and proposed by others like clarke-Wright or Gillette and Miller. Real world applications of a CVRP has been studied in many papers considering lot more assumptions like time window, customer availabilities and many more which is out of the scope of this projects. The final objective of vehicle routing is to provide a high level of customer service while keeping the operating and investment costs as low as possible [2]. This project will firstly model a CVRP and explain the exact approaches to solve a CVRP [3] which is branch and bound using different lower bound assignments approaches.

For generality, as Toth in his paper [3] considers, during the whole project the CVRP is going to be in a condition that all the customers correspond to deliveries, the demands are deterministic, known in advance and may not be split, the vehicles are identical and are based at a single central depot, only the capacity restrictions for the vehicles are imposed, and the objective is to minimize the total cost. It should be mentioned that the first part of the paper is only dedicated to ACVRPs and the second part is explaining the SCVRPs. This project is solely based on the ACVRP.

In the first section, the paper under study formulate an integer programming model based on the above mentioned constraints for an asymmetric problem. After formulation, it proposes several relaxation methods for the corresponded integer programming and compare their results based on their optimality precision. In second section it proposes another integer formulation for a symmetric case of vehicle routing problems. Moreover, it introduces three more relaxation approaches to make the algorithms for optimization solvable. Basically the asymmetric relaxations are adopted from symmetric approaches after being more generalized and fitted to the new case of asymmetricness. The first asymmetric relaxations proposed by toth and virgo are:

1. Assignment problem lower bounds (AP lower bound)
2. Disjunction lower bound
3. K-arborescences
4. Min cost flow relaxation

After introducing these four approaches they compare each approaches performance in the next section. Following examining the asymmetrical problems, the paper formulate a symmetric VRP and introduce three lower bound approaches on it:

1. The lower bound based on matching
2. The Lagrangian lower bound
3. Bounds based on set portioning

And in the last section, these three approaches performance were compared and evaluated by testing different routing cases as for the asymmetric cases were done.

2 Objective

The primary goal of this project is to take the Toth and Virgo model and implement it in a realistic case and get an exact optimum answer. As we studied the Toth work in the previous section, he is trying to introduce several approaches to solve the problems with different relaxations [3]. To have a relatively good scope of the project, one of the above mentioned methods are chosen to be implemented in a software so that the comparison between the bounds given by the relaxation and exact optimum answer that are going to be implemented using the original paper would be possible at the end. In this project, relaxations that is going to be studied is:

- Lower Bound calculation based on Min cost flow relaxation

In this study, the problem is going to be modeled as an Integer Programming with several parameters, Integer,and binary variables (In special cases of CVRP). The goal is to minimize the cost of the distribution operation while satisfying the constraints. During this study the optimization model was written in CPLEX Python as methods used to solve pure integer, and mixed-integer programming problems require dramatically more mathematical computation than those for similarly sized pure linear programs. To solve the Toth mentioned model and see the result from some generated data, I used a case proposed in the original paper of Fischetti [5] composed of 13 customer nodes with specific locations and demands (and a small case problems created randomly in python). This work would help a lot to fulfill Toth's purpose of presenting different solution approaches to CVRPs.

At the end its good to mention that this project would be beneficial regarding the network models concepts since it investigates a real-worl network modeling issue (Routing) and show us how these category of the problems could be tackled using relaxation methods.

3 Assymetric CVRP Exact Approach

Problem notation and definition:

All the notations and logic behind the definitions are adopted from Toth and Virgo paper [3].
CVRP formulation is based on a graph $G = (V, A)$ with nodes represented as V and edges represented as A. A capacitated vehicle routing problem is basically a sending out point like a depot, warehouse, or cross dock and several customers. V = {0,1,...,n} is the notation used to show n different customer nodes in the modeling including the depot. Each customer have a non negative demand d_j [3]. A nonnegative cost for each edge (i, j) is c_{ij}. It should be mentioned that a loop from a node to the same node is not possible so we show this as $c_{ii} = \infty$.
By the help of a given vertex set $S \subseteq V$ we define $\delta(s)$ as set of edges which have one or two endpoints in S. A set of K identical vehicles, each with capacity C, is available at the depot. Each vehicle may perform at most one route, and we assume that K is not smaller than $Kmin$, where $Kmin$ is the minimum number of vehicles needed to serve all the customers. Moreover, D is the maximum capacity of each truck that can carry. The CVRP consists of finding a

[3] depot demand is $d_0 = 0$

collection of K simple circuits (corresponding to vehicle routes) with minimum cost, defined as the sum of the costs of the arcs belonging to the circuits, and such that:

- each circuit visits vertex 0 (the depot vertex)
- each vertex $J \in V/\{0\}$ is visited by exactly one circuit
- the sum of the demand of the vertices visited by a circuit does not exceed the vehicle capacity.
- for simplicity we calculate the $Kmin$ as $K\min = \left\lceil \frac{\sum d_j}{D} \right\rceil$.

Problem formulation:

$$Min \sum_{i,j \in A} C_{ij} X_{ij} \tag{1}$$

$$\sum_{j \in V} X_{ij} = 1 \qquad \forall i \tag{2}$$

$$\sum_{i \in V} X_{ij} = 1 \qquad \forall j \tag{3}$$

$$\sum_{i \in V} X_{i0} = K \tag{4}$$

$$\sum_{j \in V} X_{0j} = K \tag{5}$$

$$\sum_{i \notin S} \sum_{j \in S} X_{ij} \geq \gamma(S) \quad \forall S \subseteq V \backslash \{0\}, \;\; S \neq 0 \tag{6}$$

$$X_{ij} \in \{0,1\} \tag{7}$$

Fischetti example:

Now using the formulation from **Problem formulation** part, we want to solve an asymmetric capacitated vehicle routing example proposed in [5] to obtain an exact optimum solution. The proposed example is as explained in the following:

The python code was written in a way that at the first all nodes and the depot including their demands are shown at the beginning of the optimization, so that viewer can simply picture what are the locations of the customers and what are we trying to optimize and at the end the each calculated optimum $X_{ij} = 1$ is represented as a green line showing the optimum calculated routes. The beforehand location plot of the proposed example is shown below in Figure 1.

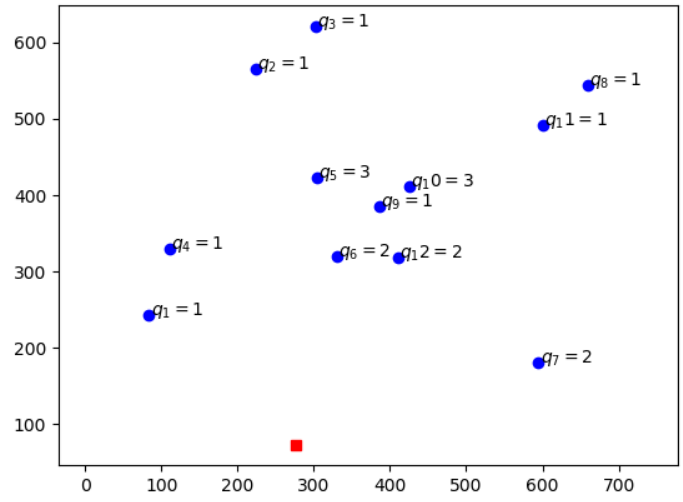


Figure 1: ACVRP locations and demands proposed by Fischetti.

As it is obvious in Figure 1 we have a depot node (red square) and 12 other customer points. Customer demands are $d_j = (0, 1, 1, 1, 1, 3, 2, 2, 1, 1, 3, 1, 2)$, D is 10 and k is equal to 3. The costs corresponding to each edge is calculated by the euclidean distance between each pair of nodes.[4]

example solution:

After implementing the formulation in Cplex-Python (the code is attached under the name of sara.ACVRP) we reach to an optimal solution which is shown in Figure 2.

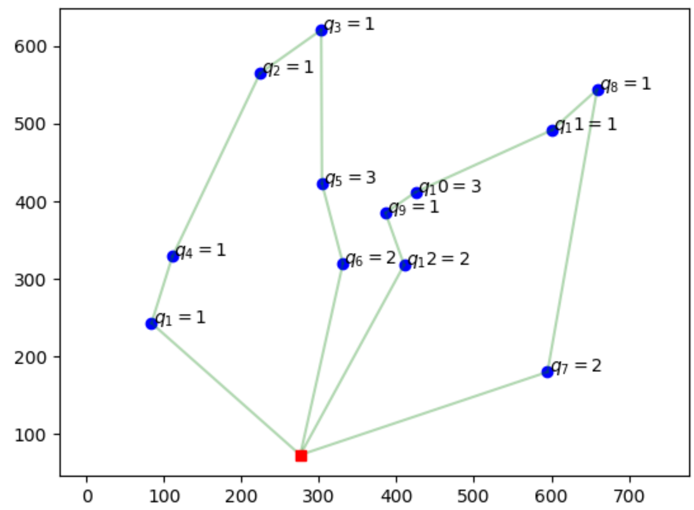


Figure 2: ACVRP optimal routes.

[4] the location of nodes were not given in the Fistechi mentioned paper [5] but using the picture given in the paper a good approximation of the locations have been introduced.

The optimal solution (Minimum Cost) of the ACVRP proposed by Fischetti is $2,637.74$.

Random example:
For justification of the model another randomize example has been conducted. In this example we have 10 customers and a depot node (red square), the truck capacities are 20 the demands and locations of each node has been randomly chosen using the "rnd" command of python (The Python code of this example is attached to the report).

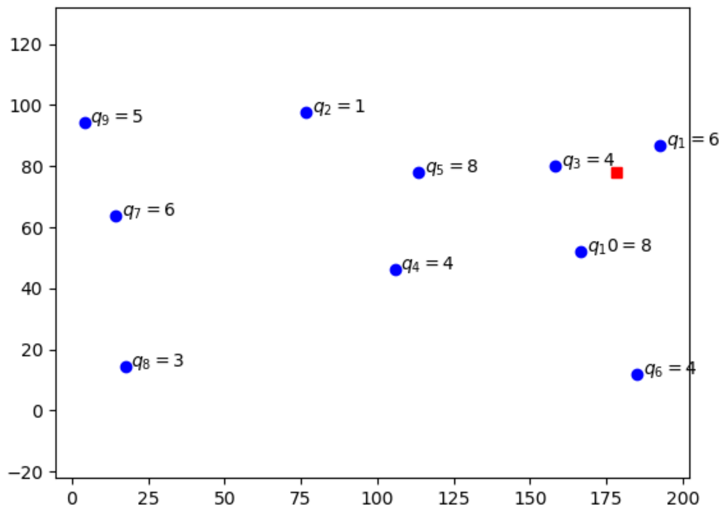


Figure 3: ACVRP locations and demands for random example.

we reach to an optimal solution which is shown in Figure 4.

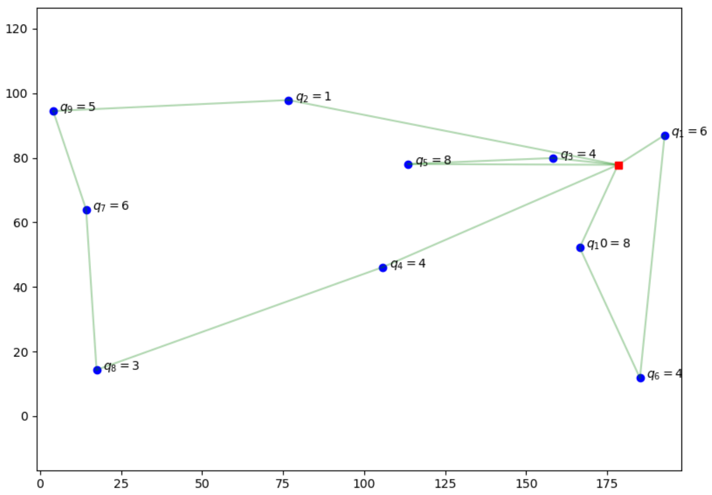


Figure 4: ACVRP optimal routes.

The optimal solution (Minimum Cost) of the ACVRP in random example is 726.249.

4 Assymetric CVRP lower bound relaxation Approach

The relaxation approach is basically possible by partitioning all the customers and depot vectors into several seperate partitions $\{S_1, ..., S_m\}$ such that:

- $0 \in S_1$
- $A_1 = \bigcup_{h=1}^{m} \{(i,j) \in A : i,j \in S_h\}$
- $A_2 = A \backslash A_1$

In simple wording it means we divide all the nodes in to the partitions and divide their corresponding arcs into internal arcs (arc in the subsets) and external arcs (arcs connecting subsets to one another). The LP lower bound here is based on the projection which for detailed explanation reader is referenced to [5]. The bound here is calculated by the sum of V_1 and V_2 which will be introduced in the following part. The contribution of V_1 which is the cost of internal arcs is neglected choosing the own words of Toth in [3] because after we change the exact formulation to a min-cost network flow the cost of arcs connecting internal nodes together is considered zero. That is why we can simply assign zero to the value of V_1.

Now it is obvious that for calculating the lower bound we need to only work with V_2. V_2 is a relaxed version of the exact formulation presented in the previous section. The relaxed formulation V_2 is shown below:

$$V_2 = Min \sum_{(i,j) \in A_2} C_{ij} X_{ij} \tag{8}$$

$$\sum_{i \in V : (i,j) \in A_2} X_{ij} \leq \begin{cases} 1 & \forall j \in V \backslash \{0\} \\ k & j = 0 \end{cases} \tag{9}$$

$$\sum_{j \in V : (i,j) \in A_2} X_{ij} \leq \begin{cases} 1 & \forall i \in V \backslash \{0\} \\ k & i = 0 \end{cases} \tag{10}$$

$$\sum_{i \notin S_h} \sum_{j \in S_h} X_{ij} = \sum_{i \in S_h} \sum_{j \notin S_h} X_{ij} \geq \begin{cases} \gamma(V \backslash S_h) & for\ h = 1; \\ \gamma(S_h) & for\ h = 2, ..., m; \end{cases} \tag{11}$$

$$X_{ij} \in \{0, 1\} \quad \forall (i,j) \in A_2 \tag{12}$$

According to Toth we know that this model can be solved efficiently since it can be rewritten as a minimum cost flow problem by some modifications. The modifications are as follows:

- two vertices, say i+ and i-, for all i $\in$ V
- two vertices, say a_h and b_h, for all h = 1...m
- a source vertex, s, and a sink vertex, t.

The arcs in the network, and the associated capacities and costs, are:

- for all $(i, j) \in A_2$: arc (i^+, j^-) with cost c_{ij} and capacity 1.
- for all $h = 1, ..., m$: arcs (a_h, i^+) and (i^-, b_h) for all $i \in S_h$, with cost 0 and capacity 1 (if $i \neq 0$) or K (if $i = 0$).
- for all $h = 1, ..., m$: arc $(a_h;\ b_h)$ with cost 0 and capacity $|S_h| - \gamma(S_h)$ (if $h \neq 1$) or $|S_1| + k - 1 - \gamma(V \backslash S_h)$ (if $h = 1$).
- for all $h = 1, ..., m$: arcs (s, a_h) and (b_h, t) both with cost 0 and capacity $|S_h|$ (if $h \neq 1$) or $|S_1| + k - 1$ (if $h = 1$).

The network described above is going to look like:[5]

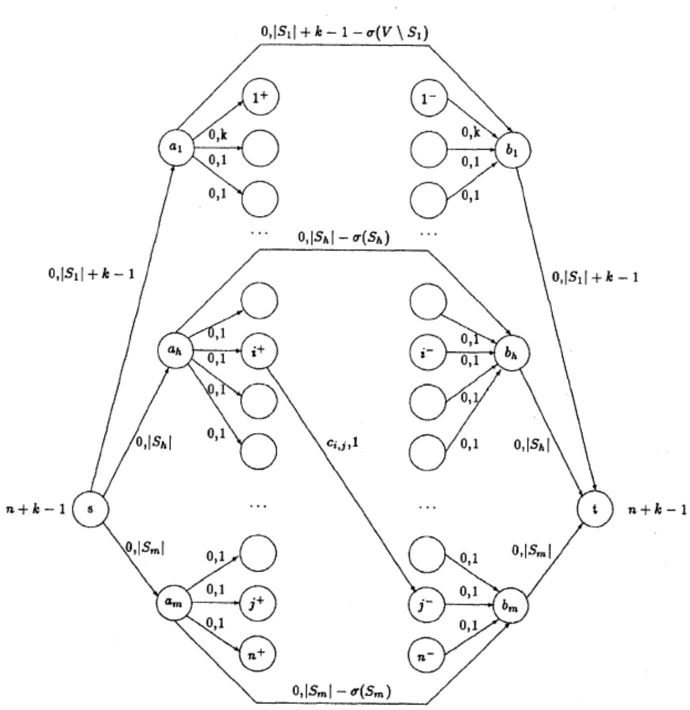


Figure 5: Auxiliary layered network of min-cost flow.

As Toth puts it, finding the min-cost s-t flow of value n+K on this network actually solves relaxation V1. As computation steps of this method is quite big we now use an algorithm called "ADDFLOW" proposed by Fischetti in [5].

[5] the picture is adopted from Fischetti [5]

begin 1. comment: initialize the current partition, as well as the current residual costs $\bar{c}_{ij}$ and the current lower bound value LB;
2. let m := n; S_h for $h = 1, .., m$; $\bar{c}_{ij}$:= c_{ij} for all (i, j) ∈ A; LB := 0;
repeat
3. solve relaxation FR with respect to $\{S_1, ..., S_m\}$ with input costs $\bar{c}_{ij}$, and let V_2, ($\bar{c}_{ij}$:= ${\varphi_{ij}}^2$ for $\forall(i, j) \in A_2$) be the computed lower bound value, residual cost vector, respectively.
4. set LB := LB $+V_2$ and update the current residual costs by set $\bar{c}_{ij} := {\varphi_{ij}}^2\, for\ \forall(i, j) \in A_2$;
5. find, if any, a violated subset collection, i.e., a collection $S_{h_1}, ..., S_{h_r}$, does not satisfy the capacity cut constraint (11) associated with $S^* := S_{h_1} \cup ... \cup S_{h_r}$, and replace $S_{h_1}, ..., S_{h_r}$ with S^* in the current partition;
6.until no suitable collection $S_{h_1}, ..., S_{h_r}$ has been found.
end

The algorithm is basically searching for all different subsets of nodes which union of them is violating the capacity cut constraint in the relaxed formulation. At first it starts with considering each node including the depot as a subset. Then the algorithm merge nodes in every possible way and assess their feasibility for constraint (11) which was mentioned above. As the algorithm is quite complex, I tried to show every single step of what algorithm do for the Fischetti example based on the implementation that I have done in Cplex Python.

As explaiend earlier we have 12 customer nodes and a depot. After considering all of them as seperate subsets and solve the relaxed problem we get:

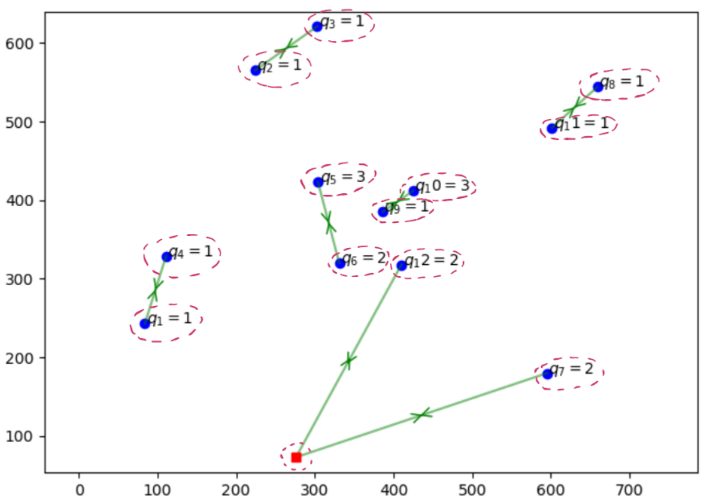
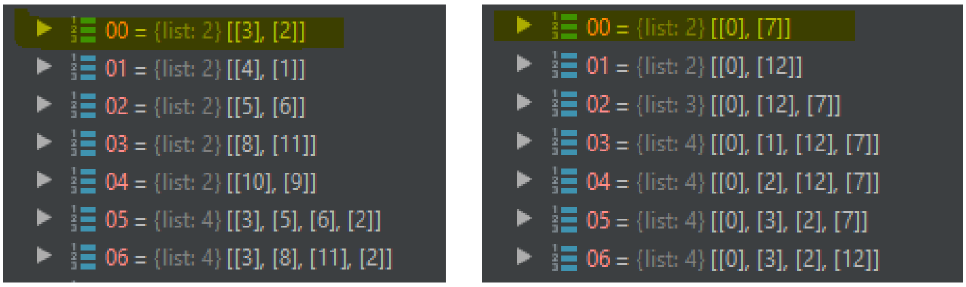


Figure 6: Iteration 1 Fischetti algorithm.

Now as the algorithm says we should search for sum collection of these above subsets to find the group that violate the capacity cut constraint. The coding do this step of the algorithm and the results have been divided to two separate subsets, the subset with node depot and the subsets including non depot nodes. This separation was done because the algorithm prefers to merge the non depot nodes as long as possible. Another thing that the algorithm must take into consideration is that among all the subsets produced the ones with the smallest length is preferred for example if merging node 1 and 2 gives us a violated subset and nodes 2,3 and 4 do the same thing according to Fischetti the algorithm should pick the first collection as it prevents the algorithm to merge into one big collection of all nodes very fast.

To satisfy the length preference after dividing the subsets into two category the algorithm flattens all the combinations that were found and calculate the length of the collections and then compare the lengths between the depot subsets and non depot subsets and choose the best subset out of them. An example of subsets generated by the algorithm is shown below to shed more light on this process:
The algorithm in the first iteration produce 85 non depot violated collections and 85 violated depot collections which can be seen in the code. Here we show some of the first ones as they have been sort size wise and we have to compare the shortest ones together.



(a) depot (b) non-depot

Figure 7: algorithm subset merge suggestion

The first candidates for each group are $S_2 \cup S_3$ and $S_0 \cup S_7$ which based on the algorithm the first merge is chosen. Now the next iteration runs from the scratch with a new 12 subsets instead of 13 which is shown below:

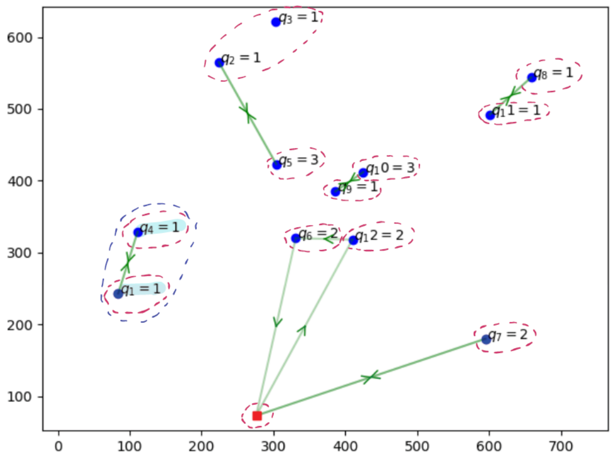
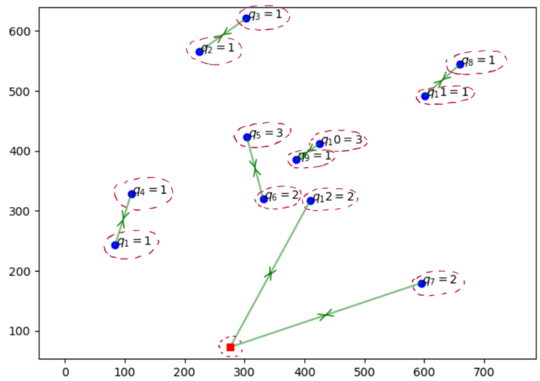
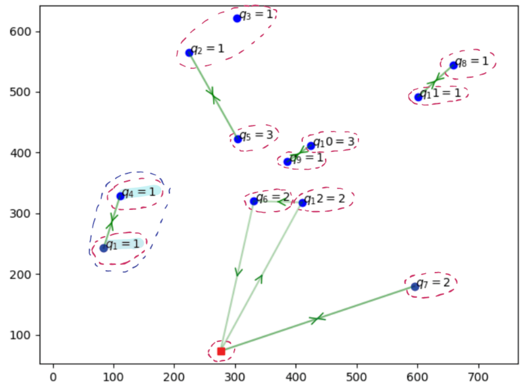


Figure 8: Iteration 2 Fischetti algorithm depot collections.

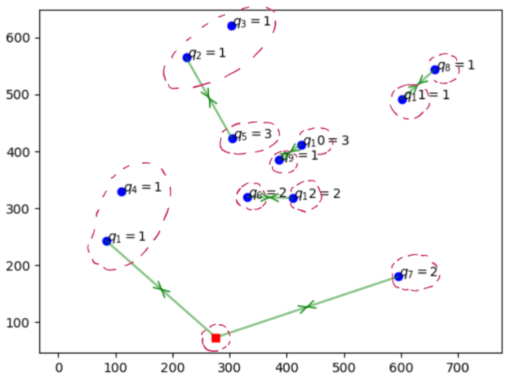
All the iterations from the begining to the end is shown below in Figure 9 and Figure 10.



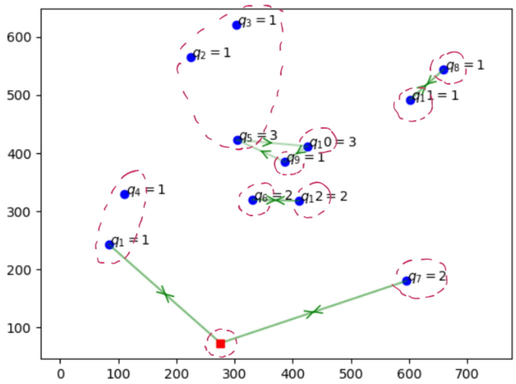
(a) iteration 1



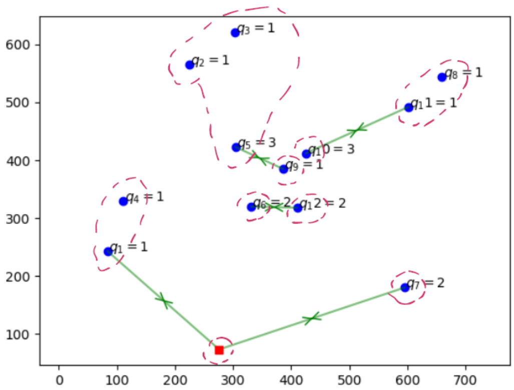
(b) iteration 2



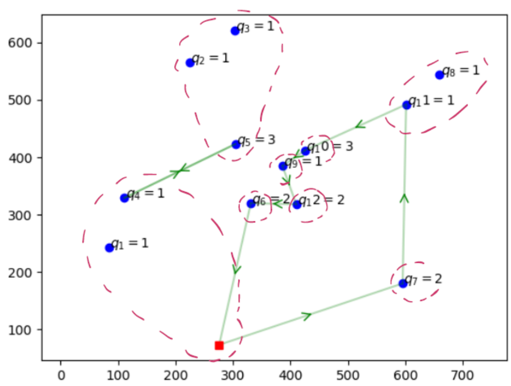
(c) iteration 3



(d) iteration 4

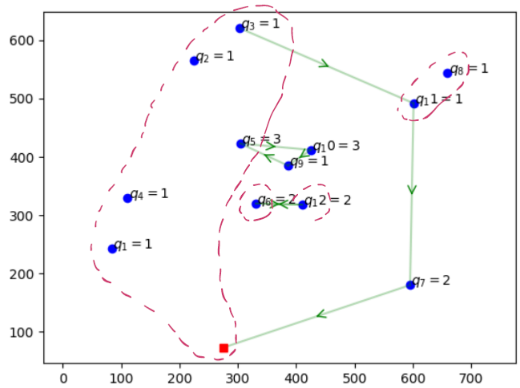


(e) iteration 5

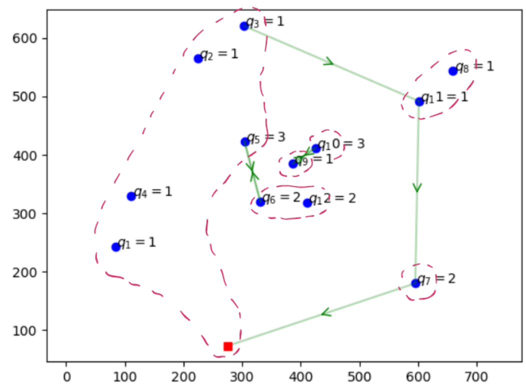


(f) iteration 6

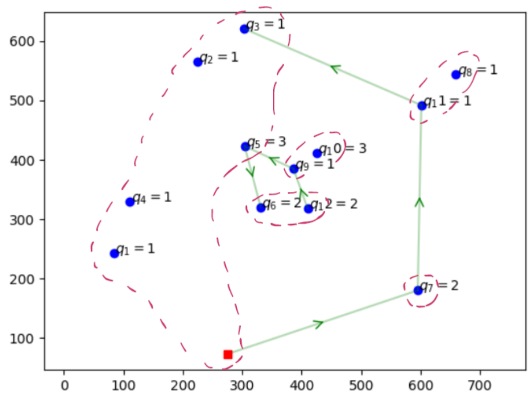
Figure 9: Iterations 1 to 6



(a) iteration 7



(b) iteration 8



(c) iteration 9

Figure 10: Iterations 7 to 9

As it can be seen in Figure 9 and Figure 12 the way algorithm is proceeding is really promising. The algorithm condition to stop is either ending up with the situation that there are no more violating subset in the collection or ending up with a subset collection of 13 nodes which means all the nodes become one single subset.

In this Fischetii example the algorithm converge and stop because of the fact that there are no more violating subset in iteration 9. This result can be seen from the V_1 and V_2 calculated at the final iteration which is presented below in Figure 11.

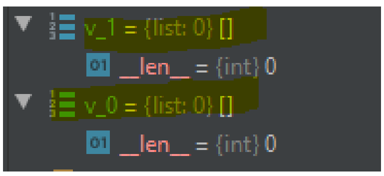
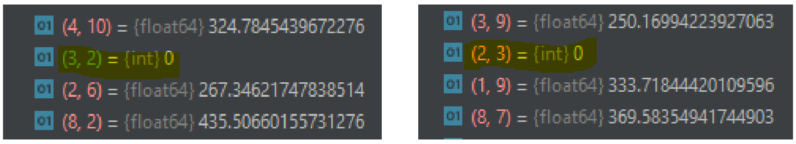


Figure 11: Iteration 9 Fischetti algorithm subset calculation.

The next important step of every iteration after creating subsets, is updating the residual cost matrix according to the ADDFLOW algorithm. The way I understood the process of the update from Fischetti in [5] and Toth in [3] is that at each iteration we use the previous cost matrix for calculating the objective function value of the relaxation formulation which is V_2.

For more clarity, for example in the first iteration cost matrix of each arc is the euclidean distance between each pair and there are no internal arcs as they are all separate from each other. This non-zero cost matrix of iteration 1 is going to be multiplied by the ${X^*}_{ij}$ calculated in the second iteration.

After the first iteration, the cost matrix will be updated according to the previous subset collection iteration. In this example, nodes 2 and 3 merged as a one single subset so the associated costs of internal arc $(2, 3)$ and $(3, 2)$ are going to be changed to zero which can be seen in the code results represented in Figure 12. This update process keeps happening until lots of elements in cost matrix of the model becomes zero but the issue here is that this cost matrix is diminishing to zero and that is good but it is not diminishing fast enough for the algorithm to calculate the lower bound in the right manner.



(a) iteration 7 (b) iteration 8

Figure 12: Cost update Iteration 1

That is why the computation of the lower bound is getting higher and higher in every iteration and getting far and far away from the optimum cost calculated in the exact approach in part **Asymmetric CVRP Exact Approach**. The fact that subset computations converge into a network which is really close to what has been presented in the exact approach (Figure 2 and Figure 12 part c) is showing that the algorithm is working perfectly. However, the residual cost matrix update is not converging to zero soon enough and that is why we would not end up with a good approximation of a lower bound for the formulation.

The lower bound calculation and the final objective function cost is represented below in Table 1.

Table 1: Lower bound and final values.

Iteration	Objective Function	Lower Bound
1	2,069.00	0
2	1,952.64	2,069.00
3	1,920.50	4,021.65
4	1,758.97	5,942.16
5	1,649.44	7,701.14
6	1,464.93	9,350.58
7	1,631.27	10,815.52
8	1,571.68	12,446.80
9	1,232.55	14,018.48

It is important to mention that as the code runs again the choice of subsets may differ and the results of the objective function value and the lower bounds may differ every time it is running but no matter what subset gets chosen at the start, the algorithm converge to the same subsets at the end.

5 Future research

In this project, I tried to understand and implement an exact network model in Cplex-Python and get results and analyze them. Moreover, a specific method of relaxation was applied on an exact problem in order to compare the results from a definite optimum answer and a relaxed one. I realized that obstacles of the relaxation way is really complicated and tackling them needs a lot of steps in terms of programming. On the other hand results from all three papers [5], [3], and [6] that I have read showed that getting close to optimum answer up to 80 percent is not an easy thing to do but if achieved it is really valuable. That is mainly because a relaxed version of a network model could help us solve much bigger and real world problems much faster.

This report is showing the importance of relaxation; however, as a future research it still needs some improvement regarding the update of the residual cost matrix which should help us receive to much more closer lower bound of the optimum answer. Additionally, reviewing all these papers showed lots of other creative ways of relaxing the asymmetric and symmetric capacitated vehicle routing problem which can be pursued later like ADDDISJ algorithm, K-arborescences.

References

[1] L. Bodin, "Routing and scheduling of vehicles and crews, the state of the art", *Comput. Oper. Res.*, vol. 10, no. 2, pp. 63–211, 1983.

[2] S. Anbuudayasankar and K Mohandas, "Mixed-integer linear programming for vehicle routing problem with simultaneous delivery and pick-up with maximum route-length", *International Journal of Applied Management and Technology*, vol. 6, no. 1, p. 2, 2008.

[3] P. Toth and D. Vigo, "Models, relaxations and exact approaches for the capacitated vehicle routing problem", *Discrete Applied Mathematics*, vol. 123, no. 1-3, pp. 487–512, 2002.

[4] S. Chopra and P. Meindl, "Supply chain management. strategy, planning & operation", in *Das summa summarum des management*, Springer, 2007, pp. 265–275.

[5] M. Fischetti, P. Toth, and D. Vigo, "A branch-and-bound algorithm for the capacitated vehicle routing problem on directed graphs", *Operations Research*, vol. 42, no. 5, pp. 846–859, 1994.

[6] P. Toth and D. Vigo, *The vehicle routing problem*. SIAM, 2002.