IOSR Journal of Mathematics (IOSR-JM)
e-ISSN: 2278-5728, p-ISSN: 2319-765X. Volume 15, Issue 3 Ser. III (May – June 2019), PP 05-15
www.iosrjournals.org

Vehicle Routing Problem with Exact Methods

Abdullahi A. IBRAHIM[1], Rabiat O. ABDULAZIZ[2] Jeremiah A. ISHAYA[3], Samuel O. SOWOLE[4]

[1]Department of Mathematical Sciences, Baze University, Nigeria.
[2]Department of Energy Engineering, PAUWES, University of Tlemcen, Algeria.
[3,4]Department of Mathematical Sciences, African Institute for Mathematical Science, Senegal.

Abstract: *TIn this research work, we applied different exact methods on Travelling salesman problem and Capaciated Vehicle routing problem which are some of the variants of Vehicle routing problem.*
For each of the problem considered, Branch and cut was applied on traveling salesman problem and colon generation technique was used on capacitated vehicle routing problem. We obtained optimal solution and hence; These methods can be used to solve similar problem to optimality.
Keywords: *Branch-and-Cut, Column Generation, Capaitated vehicle Routing Problem, Travelling Salesman problem, Vehicle Routing Problem.*

Date of Submission: 20-04-2019 Date of acceptance: 05-06-2019

I. Introduction

The cost of transportation of goods and services is one of the major challenges people face in their daily activities. This area has raised concern in todays society. A large sum of money is spent daily on fuel, goods and service delivery, equipment maintenance and so on [1]. This is where the knowledge and technique of Operations Research (OR) comes to play. If the available resources is known, one can employ the techniques of OR. According to [2], the use of computerized techniques in solving transportation problem most times often leads to about 5% 20% savings on transportation cost. Therefore, planning of distribution process, research and studying OR-techniques is worthwhile and will save some transportation cost.

The Vehicle Routing Problem (VRP) has been a problem for several decades and one of the most studied problem in logistics engineering, applied mathematics and computer science, and which is described as finding optimal routes for a fleet of vehicles to serve some scattered customers from depot [3]. VRP have several variants which includes; Travelling Salesman Problem (TSP) [4], Capacitated Vehicle Routing Problem (CVRP) [5], Periodic Vehicle Routing Problem (PVRP) [6], Vehicle Routing Problem with Time Windows (VRPTW) [7], Vehicle Routing Problem with Skills Sets (VRPSS) [8], Vehicle Routing Problem with Pickup and Devlivery (VRPPD) and so on. The VRP is a combinatorial optimization and integer programming problem which finds optimal path in order to deliver goods and services to a finite set of customers. The TSP is a classical and most widely studied problem in combinatorial optimization [9]. Which has been studied deeply in operations research and computer science since in the 1950s as a result of which a large number of techniques were developed in solving the problem. TSP describes a salesman who must travel through N cities. The order of visiting the cities is not important, as long as he is able to visit each city exactly once and comes back to the starting city [10]. Each city is connected to other cities through some link. CVRP is another variant of VRP involves vehicles with limited carrying capacity of goods and these goods must be delivered. In CVRP, the major factors we consider are the customers demands, number of vehicles availabe and the vehicle capacity.

The objective is to find optimal route such that each and every customer is served once and every vehicle is loaded according to its capacity and not more [1]. The first article was introduced by [11], which was on the concept of vehicle routing problem by trying to solve the “truck dispatching problem” with the objective of finding optimal supply-technique from the bulk terminal to the large number of service stations. [12] implemented the branch and cut algorithm to solve a large scale travelling salesman problem where the problem has to be solved many times in branch and cut algorithm before a solution to the TSP was obtained. Using large scale instances of the TSP, a substantial portion of the execution time of the entire branch and cut algorithm is spent in linear program optimiser. In their work, they constructed a full implementation of branch and cut algorithm, utilising the special structure and however, did not implement all of the refinements discussed in [13]. Column Generation (CG), an exact approach for solving VRP was used and reported successful in solving VRPTW [14, 15]. With its success, more researchers further their research with this technique. [16] proposed and used CG technique to solve the Heterogeneous Fleet Vehicle Routing Problem (HVRP). Applications of Vehicle Routing Problem cut across several areas which includes; Courier service [17], realtime delivery of customers demand [18], milk runs dispatching system in real time [19], milk collection problem [20]. The goal of this paper is to solve some variants of VRP using exacts methods. We shall use methods that will give us optimal routes for each of the variants we shall consider.

II. Materials and Methods

We briefly review Linear Programming (LP). Given a system of Linear Programming (LP) as

$$MaxZ = c_1x_1 + c_2x_2 + c_3x_3 + \ldots + c_mx_m$$

Subjected to:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \ldots + a_{1m}x_m &\leq b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \ldots + a_{2m}x_m &\leq b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + \ldots + a_{3m}x_m &\leq b_3 \\ \vdots \quad \vdots \qquad\quad \vdots \\ a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \ldots + a_{nm}x_m &\leq b_n \\ x_i \geq 0; i \in [1, \ldots, m] \end{aligned} \tag{1}$$

From the LP system above-mentioned, we can define the following

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \text{and } b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

From the definition of linear programming given above, we shall give some further definitions and theorems.

Definition 2.1. Given a system $Ax = b$ of n linear equations in m variables, for $m \geq n$. A basic solution to $Ax = b$ is obtained by setting variables $n - m = 0$and solving for the values of the remaining m variables. This assumes that setting the $n - m = 0$ produce unique values for the remaining m variables or, equivalently, the columns for the remaining m variables are linearly independent.

Definition 2.2. Any basic solution to (1) in which all variables are non-negative is a Basic Feasible Solution (bfs).

Theorem 1 (Extreme point): A point in the feasible region of an LP is an extreme point if and only if it is a basic feasible solution to the LP.

Theorem 2 (Direction of unboundedness): Consider an LP in standard form, having a bfs $b_1, b_2, \cdots, b_n$. Any point x in the LP's feasible region may be written in the form

$$x = d + \sum \sigma_i \, b_i$$

where d is 0or a direction of unboundedness and $\sum \sigma_i = 1$ and $\sigma_i \geq 0$.

Theorem 3 (Optimal bfs): Consider an LP with objective function $maxCx$and constraints $Ax = b$. Suppose this LP has an optimal solution. We now sketch a proof of the fact that the LP has an optimal bfs. If an LP has an optimal solution, then it has an optimal bfs.

Proof

Let x be an optimal solution to our LP in (1). Because $x$is feasible, theorem 2 tells us that we may write $x = d + \sum \sigma_i \, b_i$, where d is 0 or a direction of unboundedness and $\sum \sigma_i = 1$and $\sigma_i \geq 0$. If $cd > 0$, then for any $k \geq 0, kd + \sum \sigma_i \, b_i$ is feasible, and as k becomes larger and larger, the objective function value approaches ∞. This contradicts the fact that the LP has an optimal solution. If $cd < 0$, then the feasible point $\sum \sigma_i \, b_i$ has a larger objective function value than x. This contradicts the optimality of x. In short, we have shown that if x is optimal, then $cd = 0$. Now the objective function value for $x$is given by

$$cx = cd + \sum \sigma_i \, cb_i = \sum \sigma_i \, cb_i$$

suppose that $b_1$is the bfs with largest objective function value. Because $\sum \sigma_i = 1$and $\sigma_i \geq 0$.

$$cb_1 \geq cx$$

because $x$is optimal, this shows that $b_1$is also optimal, and the LP does indeed have an optimal bfs.

III. Mathematical Formulation of TSP

According to [21] there are many mathematical formulations for variants of the TSP, employing a variety of constraints that enforce the requirements of the problem in order to demonstrate how such a formulation is used in the comparative analysis as follows;

Given a complete graph $G = (V, E)$ with $|V| = n$ and $|E| = m = n\frac{(n-1)}{2}$ distances d_{ij} containing each vertex exactly once. Introducing binary variables x_{ij} for the possible inclusion of any edge $(i, j) \in E$ in the tour we get the following classical ILP formulation; Basic assumptions inlude; travel from any city to another, the graph is complete. That is to say, there is an edge between every pair of nodes. For each edge in the graph, we associate a binary variable as follows

$$x_{ij} = \begin{Bmatrix} 1, if\,(i,j) \in E \in tour \\ 0, otherwise \end{Bmatrix} \qquad (2)$$

Also since the edges are undirected, it suffices to include only edges with $i < j$ in the model. Furthermore, since we are minimizing the total distance travelled during the tour, so we calculate d_{ij} between each pair of nodes i and j. So the total distance travelled is then the sum of all the distances of the edges which are included in the tour as follows;

$$distance = \sum d_{ij} \, x_{ij} \qquad (3)$$

Since the tour can only pass through each city exactly once, then each node in the graph should have exactly one incoming and one outgoing edge i.e for every i node, exactly two of x_{ij} binary variables should be equal to (3). And we write it as follows,

$$\sum x_{ij} = 2, \forall i \in V \qquad (4)$$

Furthermore, eliminating sub tours that might arise from the above constraint, we add the following constraints;

$$\sum x_{ij} \leq |S| - 1, \forall S \subset V, S \neq 0 \qquad (5)$$

This constraints requre that for each proper(non-empty) subset of the set of cities V, the number of edges between the nodes of S must be at most $|S| - 1$.

Therefore, the final integer linear program of our TSP formulation is as follows;

$$Min \sum d_{ij} \, x_{ij} \qquad (6)$$

subjected to:

$$\sum x_{ij} = 2, \forall i \in V \qquad (7)$$

$$\sum x_{ij} \leq |S| - 1, \forall S \subset V, S \neq 0 \qquad (8)$$

$$x_{ij} \in \{0,1\} \qquad (9)$$

And if the set of cities V is of size n , then there are $2^n - 2$ subset of S of V , excluding $S = V$ and $S = 0$. Where equation (6) defines the objective function, (7) is the degree equation for each vertex, (8) are the subtour elimination constraints (SEC), which forbid solution consisting of several disconnected tours, and (9) defines all of these with the integrality constraints. Also note that some of the SEC are redundant: for the vertex sets $S \subset V, S \neq 0$, and$S^! = V - S$ we get pairs of SEC both enforcing the connection os S and $S^!$.

Algorithm 1: *Branch-and-Cut*

```
Choose an initial linear system Ax ≤ bsatisfied by allx ∈ S
set L = (Ax ≤ b), u = +∞;
while L ≠ ∅ do
    remove a system (Cx ≤ d) from L;
    if x : Cx ≤ d = ∅ then
        set t = +∞
    elsefind x* that minimizes c^T x
        subject to Cx ≤ d
        and set t = c^T x*;
    end if
    while t < u and x* ∉ S and FINDCUTS (S, x*) ≠ ∅ do
        add cuts returned by FINDCUTS (S, x*)) to Cx ≤ d;
        if {x : Cx ≤ d} = ∅ then
            set t = +∞
        elsefind x* that minimizes c^T x
            subject to Cx ≤ d
            and set t = c^T x*;
        end if
    end while
    if t ¡ u then
        if x* ∈ S then
            set u = t, x̄ = x*
        else choose a vector α and numbers β' < β'' so that
each x ∈ S satisfies either α^T x ≤ β' or α^T x ≥ β''
add (Cx ≤ d, α^T x ≤ β' and (Cx ≤ d, −α^T x ≤ −β'')
to L;
        end if
    end if
    if u = +∞ then
        return "S = ∅";
    else return x̄
    end if
```

The branch and cut algorithm is given above.
We shall solve this mathematical formulation using the above algorithm. Similarly, let formulate the CVRP as well.

IV. Mathematical Formulation of CVRP

All vehicles will originate and end at the depot, while each of the customer is visited exactly once. Let us define the following:

- $C = \{v_1, v_2, \ldots, v_m\}$: represent the set of m-customers to be considered.
- L: denote the fleet of available vehicles in a single depot. All vehicles considered are homogeneous, and we have n-vehicles.
- Q: is the maximum capacity of a vehicle, which limits the number of customers to be visited before returning to the depot.

The vehicle routing problem is a directed graph *G(V, E)* with a cost-matrix, *C* where

- $V = \{v_0, v_{1,\ldots,} v_m, v_{m+1}\}$is the set of vertices associated with C. The vertices $\{v_0, v_{m+1}\}$ represent the depot, i.e $v_0 = v_{m+1}$ and $\{V_{1,\ldots,} v_m\}$ represent m-customers.
- $E = \{(v_i, v_j) \vee 0 \leq i, j \leq m, i = j\}$is a set of $|V| \times (|V| - 1)$ directed routes/edges between the vertices. If in both directions the distance between two vertices are identical, we then add the $(i < j)$ restriction, and this is the symmetric variant.
- $C = C_{ij}$ is a cost-matrix and $C_{ij} \geq 0$ is the corresponding distance of edges (v_i, v_j), the diagonal of the matrix i.e $c_{ii} = 0$ always. Depending on whether the VRP variant in consideration is symmetric or not, $c_{ij} = c_{ji}$. The triangle inequality is assumed to hold generally, i.e $c_{ij} \leq c_{ik} + c_{kj}$ and $(0 \leq i, j, k \leq m$.

Furthermore, we need to define some important terms in this VRP problem;

• $R_i = \{v_0^i, v_1^i, \ldots, v_{k_i}^i, v_{k_i+1}^i\}$is a vector of the route of vehicle i which start and end at the depot, with $v_0^i = v_{k_i+1}^i = v_0$, $v_j^i \neq v_l^i$, $(0 \leq j < l \leq k_i)$, and k_i is the length of route R_i .

• $S = \{R_1, R_2, \ldots, R_n\}$is the set of route which represent the VRP solution instance.

• $C(R_i) = \sum_{j=0}^{k_i} C\left(v_j^i, v_{j+1}^i\right)$is the cost of route R_i .

• $C(S) = \sum_{i=1}^{n} C\,(R_i)$is the total cost of solution S which satisfies;

$R_i \cap R_j = \{v_0\}\ \forall R_i, R_j, (1 \leq i, j \leq n, i \neq j)$ and

$$.\cup_{i=1}^{n} R_i = V$$

in order for each customer to be served once. The route vectors is treated here as a set. The goal of the VRP is to minimize the $C(S)$ on the graph $G(V, E)$.

G is the graph which contains $|E| + 2$vertices, and the customers ranges from (1, 2, . . . , m). The starting and returning depots are denoted by 0 and *m + 1* respectively. Earlier in this section, we introduced the vehicle routing problem which we have now defined. However, the problem is not all about visiting the customers, there is more to their demands. In the following definitions, we shall specify these additional demands of the customers:

- **demand**; d = $(d_0, \ldots, d_m, d_{m+1})$ with $d_i > 0$ and *m* is the total number of customers which is a vector of the demands of customer, the demand of the depot is denoted by d_0 ; $d_0 = d_{m+1} = 0$ always.
- **service time**; δ is a function of service time: time to unload all the goods at customer v_i ,

i = 1, 2, . . . , m. Often times, δ is dependent on the size of the customer's demand. Henceforth, we shall use these notations as the same henceforth, $\delta_i = \delta(v_i)$.

- Let us define our decision variable as $y_{ij} = 1$ if *(i, j)* is a route and 0 otherwise.

The problem definition will be based on the following assumptions;

- The capacity constraints of all the vehicles are observed.
- Each customer can be served by only one vehicle.
- Each and every route starts at vertex 0 and ends at vertex (*m + 1*).

The mathematical formulation of Capacitated Vehicle Routing Problem is stated below.

We start with the objective function;

$$min \sum_{i=0}^{m+1} \sum_{j=0}^{m+1} C_{ij}\ y_{ij}\,, \tag{10}$$

Subjected to:

$$\sum_{\substack{j=1 \\ j\neq 1}}^{m+1} y_{ij} = 1, \forall i = 1,2, \ldots, m \tag{11}$$

$$\sum_{\substack{j=1 \\ j\neq 1}}^{m+1} y_{ij} = 1, \tag{12}$$

$$\sum_{\substack{i=0 \\ i\neq h}} y_{ih} - \sum_{\substack{j=1 \\ j\neq h}} y_{hj} = 0, \tag{13}$$

$$\sum_i y_{i,m+1} = 1, \tag{14}$$

$$x_j \geq x_i + d_j y_{ij} - Q\left(1 - y_{ij}\right), \forall i, j = \{0,1, \ldots, m+1\} \tag{15}$$

$$d_i \leq x_i \leq Q, \forall i = \{0,1,, \ldots, m+1\}, \tag{16}$$

$$y_{ij} \in \{0,1\}, \forall i, j = \{0,1,, \ldots, m+1\}, \tag{17}$$

where (10) is the objective function which minimize the total travel cost by vehicle, This function is subjected to several constraints. Constraint (11) restrict each customer to be visited and served by only one vehicle. Constraints (12), (13) and (14) ensure that each and every vehicle must originate from starting depot; 0, pass through various destinations of demands and return to end depot; m + 1. Constraints (15) and (16) ensures the capacity constraint is observed. And constraint (17) indicate integrality constraints.
Note that subtours are avoided in the solution with constraint (14) that is, cycling paths which do not pass through the depot. Constraints (15) and (16) advantage in this problem is that in terms of our customers, the formulation has a polynomial number of constraints.

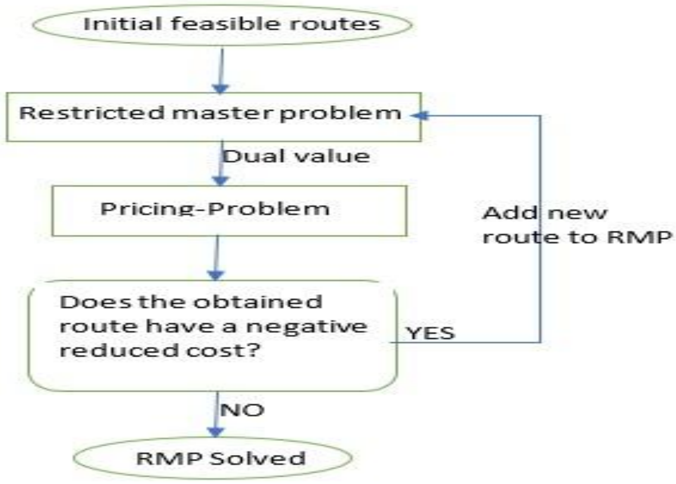

Figure 1: *Column generation procedures*

We shall solve this formulation using the column generation technique. A brief explanation of this technique is as follow;
(1) **Restricted Master Problem**: The Restricted Master Problem (RMP) is a set partitioning problem. Some routes have the potential to improve the objective function, the RMP considers these paths which were added to the set of routes due to their potential to improve the objective function. The advantage of this technique is, only routes with high potential are added to the set of routes. Set partitioning problem has been considered because it is flexible with the objective function and its constraints. This approach allows the adding of constraints and this has an impact on the existing routes. This constraint-adding characteristics, makes set-partitioning problem more flexible than several other approaches [22]. When solving RMP, a dual value is assigned to each customer, this assigned value correspond to how much we can improve the total solution by adding better routes including this customer.
(2) **Pricing Problem**: The second part of CG is the pricing-problem which is a sub-problem following the RMP to identify and generate new routes and column that will enter the set of routes (variables) in the RMP. The RMP assigns dual values to customers, which assists to identify new routes with potential to improve the objective function. The new routes generated from pricing-problem are added to the master problem. RMP is then re-solved to generate new dual-values to each customers.
(3) Lastly, we use post-optimisation technique to further improve the solution we obtained. Further explanation can be obtain in [1]. Now that we have defined the two algorithms, we proceed to the experiments as given in the next section.

V. Results

Our experimental results for the problems are given here. We use the branch and bound algorithm to solve TSP and column generation technique to solve CVRP.

Experiment 1: Exact Method with TSP

This section presents the performance tests of branch and cut algorithm on Euclidean instances of TSPLIB library. The tests were performed on a computer processor intel(R) core (TM) i5-2450m cpu 2.60GHZ @ 2.60GHT and 8GB of Ram. The adaptation of the proposed algorithm is coded into a python programming language version 3.6 . Result obtained by applying the exact method in solving the symmetric TSP problem is summarised in the tables below.

Table 1 containing the following features; Instances (denotes the data file), No of instance (number of cities in the graph), Optimality (best known solutions), Exact Method (our experimental result), Time (python time in seconds), Error (the optimality gap between the Optimality and exact method).

Table 1: Exact method with optimal solution

Instances	No of instance	Optimality	Exact method	Time(s)	Error (%)
Wi29	29	27603	**27603**	0.10	0.00
DJ38	38	6656	**6656**	*0.12*	0.00
Berlin52	52	7542	**7542**	0.15	0.00
Pr76	76	21282	**21282**	0.66	0.00
KroA100	100	108159	**108159**	0.62	0.00
Pr136	136	96772	**96772**	0.44	0.00
Pr144	144	58537	**58537**	1.63	0.00
Ch150	150	6528	**6528**	7.44	0.00
qa192	192	9352	**9352**	1.82	0.00
KroA200	200	29437	**29437**	1.61	0.00

$$Optimalgap = \frac{|ExactMet\ hod - Optimality\ |}{Optimality} \times 100\% \quad (18)$$

Table (1) shows a summary of the results obtained when we applied our exact method algorithm on the TSPLIB instances. And (18) is the optimal gap formula.

We shall further visualize results of table 1.

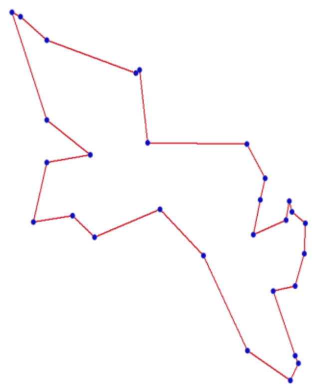

Figure 2: *A plot of wi29 Instance*

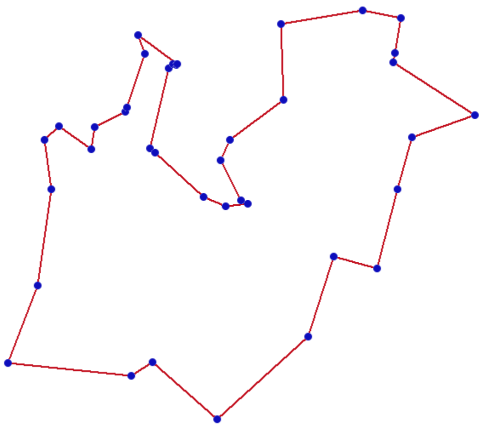

Figure 3: A plot of Dj38 Instance

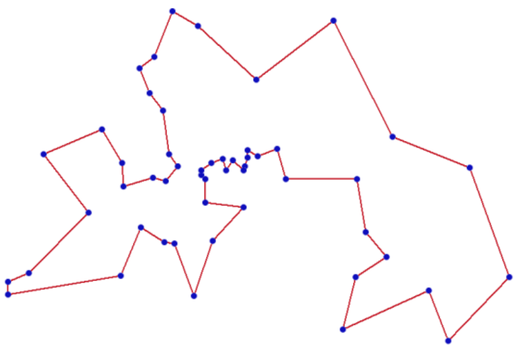

Figure 4: *A plot of Berlin52 Instance*

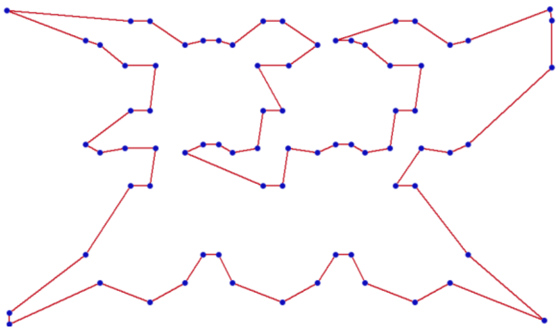

Figure 5*: A plot of pr76 Instance*

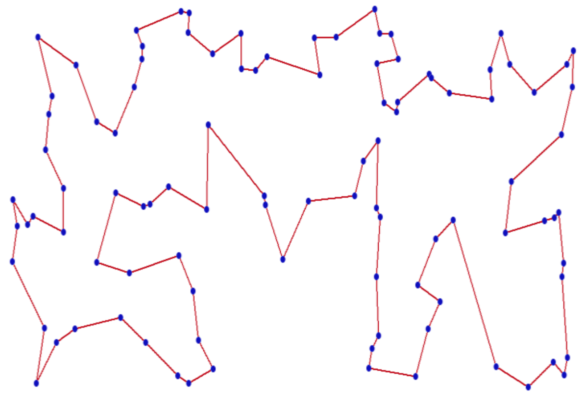

Figure 6: *A plot of KroA100 Instance*

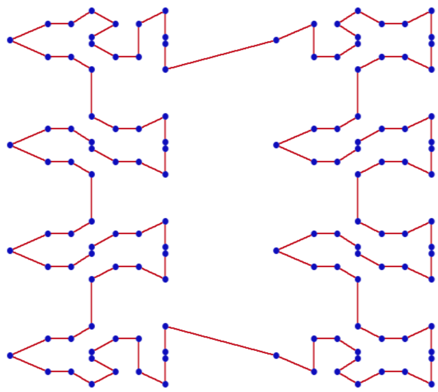

Figure 7: *A plot of pr136 Instance*

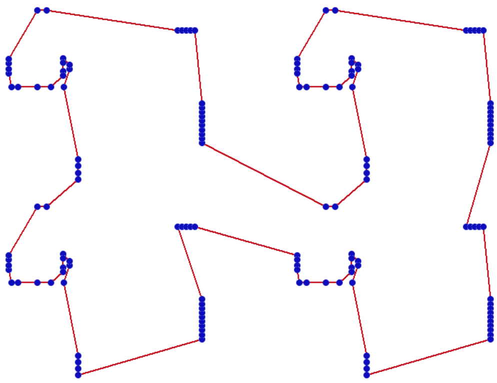

Figure 8: *A plot of pr144 Instance*

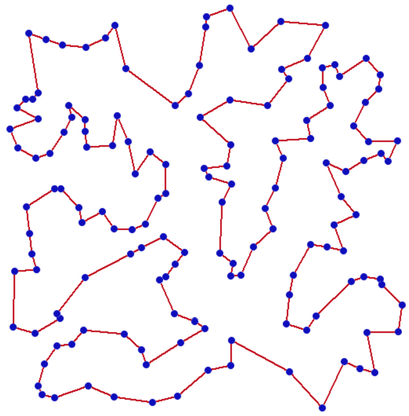

Figure 9: *A plot of ch150 Instance*

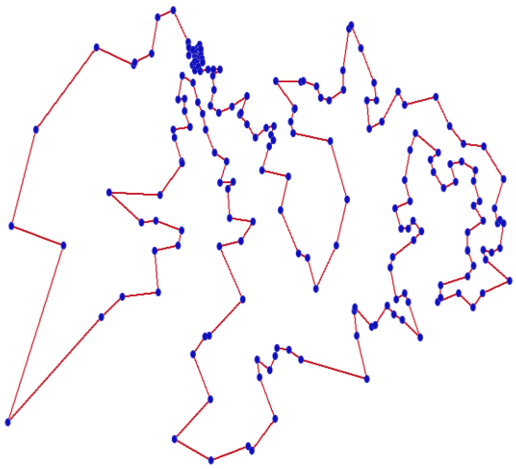

Figure 10: *A plot of qa194 Instance*

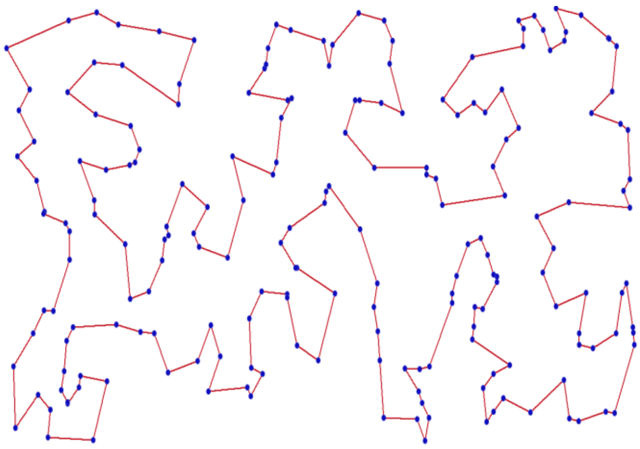

Figure 11: *A plot of KroB200 Instance*

Figures (2) - (11) above shows the routes for each of the instance considered in this experiment. The cost of each tour have been shown in (1).

Experiment 2: Exact Method with CVRP

The Column Generation (CG) method is an exact method for solving the CVRP and the VRP in general, this technique has been explicitly explained in Section (2.4), and gives an optimal solution to a small-size problem but become inefficient on big-size problem. Table (3) gives the summary of the comparison of this techniques primal and (cost value) dual problem, since CG work on dual solution of the relaxed master problem, their optimal gap in percentage. Table (3) consists of five columns; Instances, Relaxed Master Problem (RMP), (cost value) Column Generation (based on dual values), column generation computational time and optimality gap. Table (4) shows the optimal tours for each vehicles if the cost value is to be respected. These results were obtained using gurobi solver in python [23].

$$Optimalgap = \frac{Upperbound \ \ -Lowerbound}{Lowerbound} \times 100\% \qquad (19)$$

Table 2: *Column Generation results*

Instances	RMP(Primal)	Cost value	CG time(s)	Gap (%)
P-n16-k8	450	450.00	14.72	0.00
P-n20-k2	220	220.00	38.64	0.00
P-n22-k2	216	216.00	44.31	0.00
P-n22-k8	603	603.00	19.66	0.00

Table 3*: Optimal Routes*

Instance	Optimal tour	Cost
p-n16-k8	vehicle 1: $1 \to 3 \to 1$ vehicle 2: $1 \to 7 \to 1$ vehicle 3: $1 \to 1 \to 1$ vehicle 4: $1 \to 5 \to 12 \to 1$ vehicle 5: $1 \to 11 \to 13 \to 16 \to 1$ vehicle 6: $1 \to 14 \to 9 \to 8 \to 15 \to 1$ vehicle 7: $1 \to 6 \to 10 \to 4 \to 1$	450
p-n20-k2	Vehicle 1: $1 \to 7 \to 6 \to 15 \to 17 \to 10 \to 14 \to 3 \to 11 \to 2 \to 1$ Vehicle 2: $1 \to 5 \to 12 \to 16 \to 13 \to 4 \to 19 \to 18 \to 9 \to 14 \to 8 \to 20 \to 1$	220
p-n22-k2	Vehicle 1: $1 \to 2 \to 3 \to 4 \to 5 \to 6 \to 7 \to 8 \to 9 \to 10 \to 1$ Vehicle 2: $1 \to 11 \to 12 \to 13 \to 14 \to 15 \to 16 \to 17 \to 18 \to 19 \to 20 \to 21 \to 22 \to 1$	216
p-n22-k8	Vehicle 1: $1 \to 20 \to 1$ Vehicle 2: $1 \to 8 \to 6 \to 1$ Vehicle 3: $1 \to 16 \to 17 \to 1$ Vehicle 4: $1 \to 14 \to 12 \to 1$ Vehicle 5: $1 \to 11 \to 9 \to 4 \to 5 \to 1$ Vehicle 6: $1 \to 15 \to 21 \to 19 \to 1$ Vehicle 7: $1 \to 13 \to 18 \to 22 \to 1$ Vehicle 8: $1 \to 7 \to 2 \to 3 \to 10 \to 1$	603

Table 2 above shows the experimental results for the CVRP, and table 3 shows the optimal tour for these instances. Table 3 contains three features; Instance, Optimal tour and cost which implies; the instance considered, the routes that will give the optimal solution and the cost of each routes respectively.

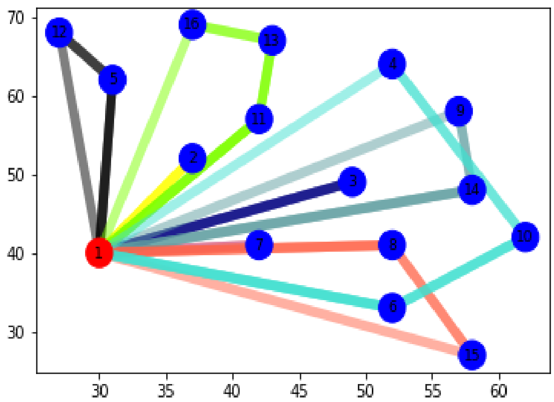

Fig ure 8: *Tour with 16 locations and 8 route*

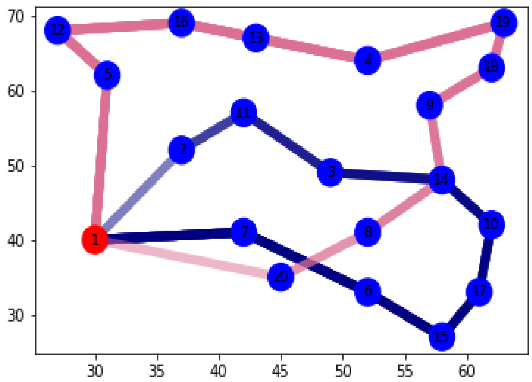

Figure 9: *Tour with 19 locations and 2 route*

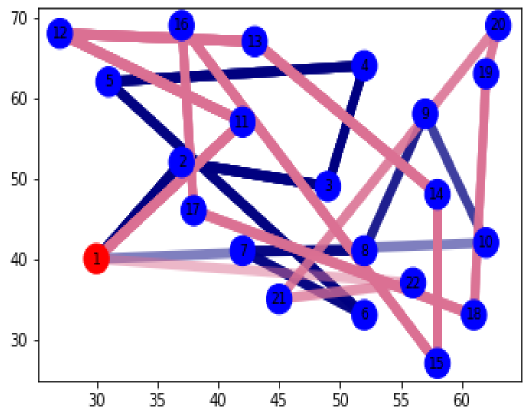

Figure 10: Tour with 22 locations and 2 route

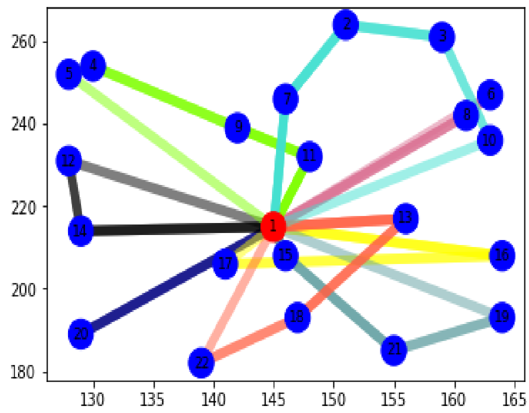

Figure 11: *Tour with 22 locations and 8 route*

Figures (8) - (11) above shows the routes for each of the instance considered in this experiment. Each locations have also been shown in table (4).

VI. Conclusion

So far, we considered only two variants of the vehicle routing problem which are; travelling salesman problem and capacitated vehicle routing problem. For the travelling salesman problem, we used branch and cut algorithm to solve the problem and applied it on some instances. We considered the following TSPLIB instance; Wi29, DJ38, Berlin52, Pr76, KroA100, pr136, pr144, ch150, qa19 and KroA200. Since technique produced an optimal solution, which ascertain the fact that exacts methods give optimal solutions. The results are shown in (1). For the capacitated vehicle routing problem, we used column generation technique to solve this mathematical model and applied it on Augerat et. al (E) data. These technique produced optimal solutions as well. The results are given in (3).

In this research work, we applied some exact solution methods of solving VRP. Where our concentration was based on the Branch and Cut algorithm for Travelling Salesman Problem and Column generation technique for the Capacitated Vehicle Routing Problem.
These techniques; branch-and-bound and column generation produced optimal solutions for travelling saleman problem and capacitated vehicle routing problem respectively. Other variants of Vehicle Routing Problem can also be solved using other exact algorithms.

Acknowledgements

All authors listed here are well appreciated.

References

[1]. A. A. Ibrahim, N. Lo,R. O. Abdulaziz, and J. A. Ishaya. CAPACITATED VEHICLE ROUTING PROBLEM. International Journal of Research - Granthaalayah, 7(3):310–327, 2019.
[2]. Paolo Toth and Daniele Vigo, An overview of vehicle routing problems, The vehicle routing problem, SIAM, 2002, pp. 126.
[3]. Bruce L Golden, Subramanian Raghavan, and Edward A Wasil, The vehicle routing problem: latest advances and new challenges, vol. 43, Springer Science & Business Media, 2008.
[4]. Gilbert Laporte, The vehicle routing problem: An overview of exact and approximate algo- rithms, European journal of operational research 59 (1992), no. 3, 345358.
[5]. Frdric Semet, Paolo Toth, and Daniele Vigo, Chapter 2: Classical exact algorithms for the capacitated vehicle routing problem, Vehicle Routing: Problems, Methods, and Appli- cations, Second Edition, SIAM, 2014, pp. 3757.
[6]. Edward J Beltrami and Lawrence D Bodin, Networks and vehicle routing for municipal waste collection, Networks 4 (1974), no. 1, 6594.
[7]. Brian Kallehauge, Jesper Larsen, Oli BG Madsen, and Marius M Solomon, Vehicle routing problem with time windows, Column generation, Springer, 2005, pp. 6798.
[8]. Paola Cappanera, Lus Gouveia, and Maria Grazia Scutell, The skill vehicle routing prob- lem, Network optimization, Springer, 2011, pp. 354364.
[9]. Chetan Chauhan, Ravindra Gupta, and Kshitij Pathak, Survey of methods of solving tsp along with its implementation using dynamic programming approach, International Journal of Computer Applications 52 (2012), no. 4.
[10]. Christopher Edwards and Sarah Spurgeon, Sliding mode control: theory and applica- tions, Crc Press, 1998.
[11]. George B Dantzig and John H Ramser, The truck dispatching problem, Management science 6 (1959), no. 1, 8091.
[12]. Christel Erna Geldenhuys, An implementation of a branch-and-cut algorithm for the travelling salesman problem, Ph.D. thesis, University of Johannesburg, 1998.
[13]. Manfred Padberg and Giovanni Rinaldi, A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems, SIAM review 33 (1991), no. 1, 60100.

[14]. Martin Desrochers, Jacques Desrosiers, and Marius Solomon, A new optimization algorithm for the vehicle routing problem with time windows, Operations research 40 (1992), no. 2, 342354.
[15]. Niklas Kohl, Jacques Desrosiers, Oli BG Madsen, Marius M Solomon, and Francois Soumis, 2-path cuts for the vehicle routing problem with time windows, Transportation Science 33 (1999), no. 1, 101116.
[16]. Eunjeong Choi and Dong-Wan Tcha, A column generation approach to the heterogeneous fleet vehicle routing problem, Computers & Operations Research 34 (2007), no. 7, 2080 2095.
[17]. Michel Gendreau, Francois Guertin, Jean-Yves Potvin, and Ren Sguin, Neighborhood search heuristics for a dynamic vehicle dispatching problem with pick-ups and deliveries, Transportation Research Part C: Emerging Technologies 14 (2006), no. 3, 157174.
[18]. Lars M Hvattum, Arne Lkketangen, and Gilbert Laporte, Solving a dynamic and stochastic vehicle routing problem with a sample scenario hedging heuristic, Transportation Science 40 (2006), no. 4, 421438.
[19]. Luce Brotcorne, Gilbert Laporte, and Frederic Semet, Ambulance location and relocation models, European journal of operational research 147 (2003), no. 3, 451463.
[20]. GDH Claassen and Th HB Hendriks, An application of special ordered sets to a periodic milk collection problem, European Journal of Operational Research 180 (2007), no. 2, 754769.
[21]. Kevin M Curtin, Gabriela Voicu, Matthew T Rice, and Anthony Stefanidis, A com- parative analysis of traveling salesman solutions from geographic information systems, Transactions in GIS 18 (2014), no. 2, 286301.
[22]. GPT van Lent, Using column generation for the time dependent vehicle routing problem with soft time windows and stochastic travel times, Masters thesis, 2018.
[23]. Inc. gurobi optimization. gurobi optimizer reference manual, 2019, url, http://www.gurobi. com, 2019.