Journal of Advances in Computer Research
Quarterly pISSN: 2345-606x eISSN: 2345-6078
Sari Branch, Islamic Azad University, Sari, I.R.Iran
(Vol. 7, No. 1, February 2016), Pages: 1-22
www.jacr.iausari.ac.ir

# A Meta-heuristic Approach to CVRP Problem: Local Search Optimization Based on GA and Ant Colony

**Arash Mazidi**[✉]**, Mostafa Fakhrahmad, Mohammadhadi Sadreddini**
*Department of Computer Engineering, Shiraz University, Shiraz ,Iran*
mazidi@cse.shirazu.ac.ir; mfakhrahmad@cse.shirazu.ac.ir; sadredin@shirazu.ac.ir

**Received: 2015/03/15; Accepted: 2015/07/07**

**Abstract**
*The Capacitated Vehicle Routing Problem (CVRP) is a well-known combinatorial optimization problem that holds a central place in logistics management. The Vehicle Routing is an applied task in the industrial transportation for which an optimal solution will lead us to better services, save more time and ultimately increase in customer satisfaction. This problem is classified into NP-Hard problems and deterministic approaches will be time-consuming to solve it. In this paper, we focus on enhancing the capability of local search algorithms. We use six different meta-heuristic algorithms to solve VRP considering the limited carrying capacity and we analyze their performance on the standard datasets. Finally, we propose an improved genetic algorithm and use the ant colony algorithm to create the initial population. The experimental results show that using of heuristic local search algorithms to solve CVRP is suitable. The results are promising and we observe the proposed algorithm has the best performance among its counterparts.*

***Keywords:*** *Vehicle Routing Problem, Capacitated Vehicle Routing Problem, Meta-Heuristic Algorithms, Local Search, Genetic Algorithm.*

## 1. Introduction

In recent decades, one of the major issues that are widely used to increase the effectiveness and efficiency of the system of transportation is vehicle routing problem. This problem includes a number of vehicles and the number of customers at different sites. The vehicles should respond to all requests of the costumers in the shortest possible time. An example is shown in Figure 1. Some of mathematicians and computer scientists have done researches to solve such problem by using optimal solution.

Objective of solving the problem is to find a better solution in order to reduce the path length that should be used by vehicles, the spent time by vehicles, the number of required vehicles, and the overall required time to serve all the requests. We can add constraints to the problem by considering some circumstances. If we add the time constraint to this problem, each vehicle has a limited time to respond to request of customers and will be penalize in case of any delay in service. We can mention picking up garbage and industrial waste, distribution of fuel in fuel stations and cash to bank branches as examples of vehicle routing with time limits.

Another limitation is the capacity constraint. Each vehicle has a limited capacity to serve its customers; in this case, the problem will be a capacitated vehicle routing problem.

Another routing problem has time and capacity constraints. In this problem, vehicles should provide services to customers in specified time and limited capacity as customer's request. In fact, the objective is to minimize the vehicle's travel time, also the total demand of commodities for each route may not exceed the capacity of the vehicle that serves that route. In addition, a solution is feasible if the total quantity assigned to each route does not exceed the capacity of the vehicle that services the route [1].

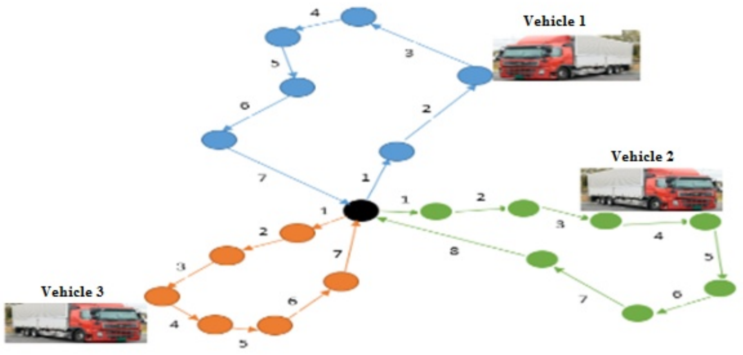

*Figure 1. Vehicle Routing*

Vehicle routing problem is one of the NP-Hard problems that deterministic algorithms take too much time to solve it. We recommend using meta-heuristic methods to save the time of problem solving.

Various approaches have been presented to solve the CVRP in the last decades, such as linear programming [2], meta-heuristics methods [3,4], and hybrid heuristics with variable neighborhood search or constructive heuristic methods [5-8]. The overview of methods presented in [4] shows that at least 29 different methods for the CVRP exist, all of which achieve more or less comparable performance. Although several methods can produce good solutions, the computational time is long when the scale of instances is large. Meanwhile, an abundance of methods for the CVRP is a population-based algorithm and the parameter setting of the algorithm is important; however, the parameter setting of many meta-heuristics is not considered in the literature.

Kanthavel et al. [9] have provided nesting idea of Particle Swarm Optimization to solve the problem of vehicles routing. In their method, the optimization algorithm is used to identify the routes of vehicles in the nested function and origin. In this approach, local optimization techniques had been used and the results show that the proposed method has a better computational performance in comparison with other methods.

Tan et al. [10] have combined the ant colony method with some heuristic approaches in order to solve the vehicle routing problem. They tried to improve the path of ants. In the proposed method, a new mechanism is presented for evaporation of the pheromone left by ants on the paths that improves the search of optimized paths by the ant.

Lei et al. [11] analyzed the problem of vehicle routing with time constraint for delivery to each customer with random capacity. Also, they proposed an adaptive large neighborhood search heuristic for the solution. Modified Solomon benchmark instances are used in the experiments. The computational results clearly show the superiority of the proposed heuristic over an alternative solution approach.

Marinakis et al. [12] have combined the Genetic and Particle Swarm Optimization algorithms to achieve optimum results. In their work, PSO algorithm and its principles are used for evolving solutions in the population of the genetic algorithm. It produces better solutions and better population and makes better choices in selection of parents in the genetic algorithms. Thus, the result of this technique will result in better exploration in the problem space.

Mazzeo et al. [13] have implemented the ant colony algorithm for vehicle routing problem with capacity constraints using meta-heuristic techniques. They compared the results with other algorithms. The results show a relatively better performance.

Yu et al. [14] offered an improved ant colony optimization (IACO), which possesses a new strategy to update the increased pheromone, called ant-weight strategy, and a mutation operation, to solve VRP. The basic idea is enhancement of the pheromones. The computational results for fourteen benchmark problems are reported and compared to other meta-heuristic approaches.

Gounaris et al. [15] presented the robust capacitated vehicle routing problem under demand uncertainty to address the minimum cost delivery of a product to geographically dispersed customers using capacity-constrained vehicles. Robust rounded capacity inequalities are developed and it is shown how they can be separated efficiently for two broad classes of demand supports.

Baldacci et al. [16] described a new integer programming formulation for the CVRP based on a two-commodity network flow approach. Also, they presented a lower bound derived from the linear programming (LP) relaxation of the new formulation which is improved by adding valid inequalities in a cutting-plane fashion. Moreover, they presented a comparison between the new lower bound and lower bounds derived from the LP relaxations of different CVRP formulations proposed in the literature. A new branch-and-cut algorithm for the optimal solution of the CVRP was described in their work.

Shang et al. proposed a hybrid of GA and ACO as a new algorithm to solve TSP [17]. GA is benefited to initiate the matrix of pheromone in ACO and to recombine the route from ACO. The authors had claimed that the hybrid algorithm is more effective compared to GA and ACO.

Duan and Yu suggested the use of memetic algorithm to find the combination of parameters in the ACO [18]. The authors had claimed that this hybridization would simplify selection of the adjustable parameter in the ACO where human experience is needed and in most cases depending on coincidence.

Jin-Rong et al. developed two sub-algorithms based on GA and ACO hybridization covering up the weaknesses of both algorithms [19]. The ACO is employed to help GA to eliminate the appearance of invalid tour, while GA is used to overcome the dependency on the matrix of pheromone in the ACO.

In this paper, we review and analyze various algorithms, including simulated annealing, stochastic hill climbing, genetic, tabu, ant colony and particle swarm. Then, we will propose an improved genetic algorithm for solving the Capacitated Vehicle Routing Problem (CVRP). Finally, we have conducted experimental evaluation using standard test data, in addition to comparing the results together; we will compare the results with optimal solutions. The proposed algorithm is a hybrid of Genetic algorithm and Ant Colony. The main idea of the proposed algorithm is to define an appropriate approach to hybridize GA and ACO to find CVRP solution. The Genetic algorithm is a population-based algorithm and its work is on population of solutions. The Ant Colony algorithm aims to search for an optimal path in a graph, based on the behavior of ants seeking a path between their colony and a source of food. Therefore, the ant colony algorithm creates appropriate solutions and the genetic algorithm uses the generated population of solutions. The proposed algorithm combines GA and ACO algorithms and aims to include benefits of both algorithms. Moreover, local search is applied on the achieved solutions in the ant colony algorithm in order to create a better initial population and it is applied on GA algorithm in order to find better solutions.

The advantages of the proposed algorithm include faster convergence and lower risk of getting stuck in local optimum in some applications. It can solve any optimization problem that can be described with the chromosome encoding. It solves problems with multiple solutions. Since the proposed algorithm is not dependent on the error surface, we can solve multi-dimensional, non-differential, non-continuous, and even non-parametrical problems by this method.

The rest of this paper is organized as follows. Section 2 describes the algorithms that can be used for solving CVRP. Section 3 presents the proposed algorithm (Improved Genetic Algorithm). The experimental evaluation results as well as the details of algorithms are reported in Section 4. Finally, Section 5 concludes the paper.

## 2. Methods for Solving CVRP

In this section, we introduce and apply different approximate methods for solving the capacitated vehicle routing problem. In all methods, we consider a vector for each vehicle containing its customers that requested some services. Any solution for this problem might result in the sample matrix that shown in relation 1.

$$Solution = Matrix[Trucks.Count, ServiceList.Count] \tag{1}$$

The purpose of solving this problem is to minimize the length of the route traveled by the vehicles, as shown in relation 2.

$$\mathrm{Cos}\,t = Min\Big(\sum_{k\in Trucks}\ \sum_{i\in ServiceList_k} Dis\tan ce(Matrix[k,i], Matrix[k,i+1])\Big) \tag{2}$$

When we determine a random solution as the initial solution, we must make lists for each vehicle as completely random list of customers according to the capacity of the vehicle. Services for vehicles should not be more than the capacity of vehicles. When we need solutions of neighbors, we use two mechanisms for producing these solutions. The first mechanism, move the customers in the list of a vehicle and the second mechanism is used for removing a customer from the list of a vehicle and adding it to the list of another with respect to the capacity of vehicles.

We have used the local search algorithm to find the best neighbors in the neighborhood of a solution. The local search algorithms use the two mechanisms mentioned to make a neighbor. Termination condition for all algorithms is to achieve the best result or achieve considered time for completion of the program (which is considered 30 seconds in this paper). In the following subsection, we will discuss meta-heuristic algorithms.

## *2.1 Simulated Annealing Algorithm*

Simulated annealing (SA) is a generic probabilistic meta-heuristic for the global optimization problem of locating a good approximation to the global optimum of a given function in a large search space. It is often used when the search space is discrete. The method was independently described by Scott Kirkpatrick, C. Daniel Gelatt and Mario P. Vecchi in 1983 [20], and by VladoCerny in 1985 [21]. The SA algorithm is inspired of Monte Carlo model (Monte Carlo model is relation between the atomic structure, entropy and temperature during the cooling process of a material). The SA is shown in Figure 2. At each step, the SA heuristic considers some neighboring state s' of the current state s, and probabilistically decides between moving the system to state s' or staying in state s. These probabilities ultimately lead the system move to states having lower energy. Typically, this step is repeated until the system reaches a state that is good enough for the application, or until a given computation budget has been exhausted. In this model, at first, material has very high temperature and then temperature will down to reach the balance. Speed of lowering the temperature of the material is very important and has an essential role in algorithm. If the temperature decreases rapidly, the balance will not be reached and the material will encounter some problems [20].

```
1.  Procedure Simulated Annealing
2.  Begin
3.    t ← 0, initialize T
4.    Select a current point Vc at random and evaluate Vc
5.    Repeat
6.      Repeat
7.        Select Vn from the neighborhood of Vc
8.  If cost(Vc) < cost(Vn) then
9.  Vc←Vn
10. Else if random[0,1) <$e^{(\frac{cost(Vn)-cost(Vn)}{T})}$
11. Vc←Vn
12.     Until( termination-condition)
13.     T ← g(T,t)
14.     t ← t+1
15.   Until (halting-criterion)
16. End
```

***Figure 2. Simulated Annealing Algorithm [22]***

We use the idea of minimizing the temperature in optimizing the vehicles routing. At the first of implementation, we need initial solution that we create a solution randomly. In the next step, we need to create neighbors of the initial solution (manufacturer of neighbors described earlier) and then use the local search on them to find best solution. We use possibility in line 10 of algorithm in Figure 2, to select a neighbor solution as the current solution. We do this loop to meet the requirements of the end of program.

The algorithm uses a function, called cooling temperature, g(T,t). This function is used to decrease the temperature of material to find solution in this problem. The proposed function can be observed in equation 3.

$$g(T,t) = 4 \times \sqrt{\frac{MatIteration}{t}} \tag{3}$$

MaxIteration is the number of iterations of the algorithm (considered 5000, in this paper), and t is for the counter of the algorithm.

### *2.2 Stochastic Hill Climbing Algorithm*

Stochastic hill climbing is a variant of the basic hill climbing method. While basic hill climbing always chooses the steepest uphill move, stochastic hill climbing chooses at random from among the uphill moves, the probability of selection can vary with the steepness of the uphill move. Figure 3 shows the Stochastic Hill climbing. This algorithm maintains a risk factor (line 7 of Figure 3) in order to go out the local optimal solutions [23].

```
1.  Procedure Stochastic hill-climbing
2.  Begin
3.    t ← 0
4.    Select a current string Vc at random and evaluate Vc
5.    Repeat
6.      Select Vn from the neighborhood of Vc
7.      Select Vn with probability $\frac{1}{1+e^{(\frac{cost(Vc)-cost(Vn)}{T})}}$
8.      t ← t+1
9.    Until t = MAX
10. End
```

***Figure 3. Stochastic Hill Climbing [22]***

In the implementation of this algorithm, in contrast to the SA algorithm, there is no function for changing the temperature (T). In the stochastic hill climbing algorithm, we consider constant value for temperature (T) equaling five. The initial solution is random and we calculate track of solution using equation 2. Then, we create a neighbor of the current solution using the neighbor creation mechanism described earlier. We use local search on new solution for finding better solution and by considering probability in the algorithm, we select the next solution and do this process until the termination condition of the algorithm is met. At the end of the algorithm, we will receive the best solution.

### *2.3 Ant Colony Algorithm*

Ant Colony Optimization Algorithm (ACO) is a probabilistic technique for solving computational problems that aim at finding good paths through graphs. Initially proposed by Dorigo in 1992 in his PhD thesis [24], the first algorithm was aiming to search for an optimal path in a graph, based on the behavior of ants seeking a path between their colony and a source of food. The ant colony algorithm is shown in Figure 4, simulates the optimization process inspired by the behavior of real ants in searching food.

```
1. Procedure Ant Colony
2. Begin
3.     Set Parameter, initialize pheromone trails
4.     While termination condition not met Do
5.         Construct Ant Solution
6.         Update Pheromones
7.     End While
8. End
```

***Figure 4. Ant Colony Algorithm***

An example of how a real ant colony manages to find the shortest route from its nest to the food, based on different quantities of pheromone deposited, is presented in Figure 5.

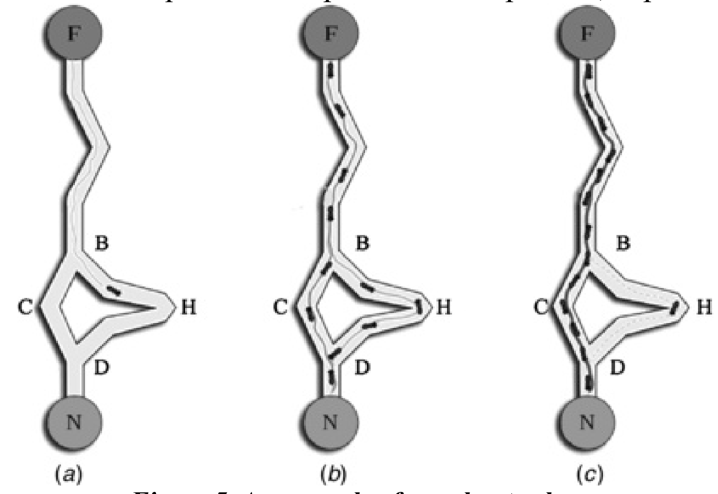

***Figure 5. An example of a real ant colony.***

The ants move along the path from food source F to the nest N. At point B, all ants walking to the nest must decide whether to continue their path from point C or from point H (Figure 5 (*a*)). A higher quantity of pheromone on the path through point C provides an ant a stronger motivation and thus a higher probability to follow this path. As no pheromone was deposited previously at point B, the first ant reaching point B has the same probability to go through either point C or point H. The first ant following the path BCD will reach point D sooner than the first ant that followed path BHD, due to its shorter length. The result is that an ant returning from N to D will trace a stronger trail on path DCB, caused by the half of all the ants that by chance followed path DCBF and by the already arrived ones coming via BCD. Therefore, they will prefer path DCB to path DHB.
Consequently, the number of ants following path DCB will increase with time unlike those following path BHD. This causes the quantity of pheromone on the shorter path to grow faster than that of the longer one. Hence, it is more likely that a typical ant chooses the shorter path.

To implement this algorithm for CVRP, we use the ant colony system algorithm [25]. We use $\tau_0 = 0.0001$as the initial value for pheromone of every customer. Each ant is a solution of the problem in this simulation. We considered 30 ants in our implementation and selected the initial points for every ant, randomly. Initial points represent the first customers for each vehicle. Then, in order to select the next customer for each ant (solution), we use pseudo-random-proportional formula that is shown in equation 4 by parameters α=1 and β=2.

$$p_k(r,s) = \frac{[\tau(r,s)]^{\alpha} \times [\eta(r,s)]^{\beta}}{\sum_{u \in Customers} ([\tau(r,u)]^{\alpha} \times [\eta(r,u)]^{\beta})} \tag{4}$$

τ is the pheromone that is putted by ants between customers and η is the attractiveness of the path between customers, as defined in equation 5.

$$\eta(i,j) = \frac{1}{Dis\tan ce(i,j)_{i,j \in Customers}} \tag{5}$$

After completing the path of ants (solutions), we should update the pheromones. We update them by two mechanisms. The first mechanism is local update and the second one is global update.

Pheromones are going to be updated periodically in order to accentuate the better solutions and increase their attraction for other ants. The global and local update formulas are given in equations 6 and 7, respectively.

$$\tau(bes\tan t) = 0.9 \times \tau(bes\tan t) + \frac{2}{\mathrm{Cos}\, t(bes\tan t)} \tag{6}$$

$$\tau(r,s) = 0.9 \times \tau(r,s) + 0.1 \times \tau_0 \tag{7}$$

After updating pheromones, new solutions are generated by new pheromones and the algorithm runs recurrently, until meeting the termination condition. The optimal solution will be reached at the end of the algorithm.

### *2.4 Tabu Search Algorithm*

Tabu Search algorithm (as shown in Figure 6) is one of the meta-heuristic approaches which was firstly introduced by Glover [26]. Tabu search enhances the performance by using memory structures that describe the visited solutions or user provided sets of rules. If a potential solution has been previously visited within a certain short-term period or if it has violated a rule, it is marked as tabu (forbidden) such that the algorithm does not consider that possibility, repeatedly. The tabu search algorithm uses a local search to move from one potential solution X to an improved solution X' in the neighborhood of X.

The used memory structures in tabu search can roughly be divided into three categories:

I. Short term: The list of solutions that have recently been considered. If a potential solution appears on the tabu list, it cannot be revisited until it reaches an expiration point.

II. Intermediate term: Intensification rules intended to bias the search towards promising areas of the search space.

III. Long term: Diversification rules that drive the search into new regions (i.e. regarding resets when the search becomes stuck in a plateau or a suboptimal dead-end).

```
Procedure Tabu Search
Begin
    l ← Maximum tabu list length
    N ← number of t-weaks
    S ← some initial candidate solution
    Best ← S
    L ← {} a tabu list of maximum length l
    Enqueue S into L
    Repeat
      If (length(L) > l ) then
          Remove oldest element from L
      R ← t-weak(copy(S))
      For n-1 times do
         W ← t-weak(copy(S))
         If (W∉L and (Quality(W)>Quality(R) or R∈L))
             R ← W
         If ( R ∉ L  and Quality(R) > Quality(S))
             S ← R
         Enqueue R into L
       If (Quality(S) > Quality( Best) ) then
             Best ← S
    Until Best is the optimum or time is out
End
```

***Figure 6. Tabu Search Algorithm***

In the implementation of this algorithm, we use two different tabu lists. In fact, we implement this algorithm with two approaches. In the first approach, we use the list of solutions and in second approach; the tabu list is a two-dimensional or square matrix with number of customers.

In the first approach, after creating a track or a solution randomly, we should compare it to available solutions in the tabu list and if it is not present in the tabu list and the value of track is less than the previous solutions, we choose it and put it in the tabu list. However, in the second approach, there is a value between two customers for each track. This value shows the use of track by vehicles. While we generate a solution, we should select a path that used less than others. We select a path between vehicles from customer A to customer B with less value in the matrix. When a path is selected, its value is incremented by one. For each step, we select the best solution and repeat this task to achieve the optimum solution or the termination condition of the algorithm. At the end of the algorithm, we will have the best solution.

### *2.5 Particle Swarm Algorithm*

Particle Swarm Optimization (PSO) algorithm is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. PSO algorithm optimizes a problem by having a population of candidate solutions, here named particles, and moving these particles around in the search space according to simple mathematical formulae over the particle's position and velocity. Figure 7 shows the PSO algorithm which is inspired by the behavior of fishes and birds in the nature. These creatures have group work and this algorithm uses this feature of creatures. PSO is a meta-heuristic method as it makes few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions. However, meta-heuristic methods such as PSO do not guarantee to find the optimal solution. PSO can be used for optimization problems that are partially irregular, noisy, change over time, etc. [27].

```
1.  Procedure Particle Swarm
2.  Begin
3.      For each particle
4.          Initialize particle
5.      For each particle
6.          Calculate cost
7.              If the cost is better than the best cost (pbest) in history
8.                  Set current value as the new pbest
9.              Choose particle with best cost of all the particles as gbest
10.             Update particle position
11.     While maximum iterations or optimum result
12. End
```

***Figure 7. Particle Swarm Algorithm***

We used the idea presented in [28] to solve the CVRP by PSO algorithm. In implementation of this algorithm, we generated the initial solution greedy and randomly. We have two methods to encrypt and decrypt the solutions in this implementation. In the encryption method, the paths are encrypted to a matrix A[customer.count + 1, customer.count + 1]. The value of A[i, j] is the possibility of moving vehicles from customer i to customer j. This matrix completed according to the initial solutions that are generated at the first of the algorithm.

At the end of the algorithm, another method should be run. In the decryption method, we choose the customer by considering the probabilities of each row of the matrix A and should take the most likely customers for vehicles; we should pay attention to capacity of vehicles and choose appropriate customers for each vehicle. Also, we consider two probability matrices for the best path to each vehicle and the best public path (best path among all vehicles) denoted as P and G, respectively. By using these matrices, we guide movements of the solution (particles) towards the optimal solution. To update the particle position (solution), equation 8 and 9 are used.

$$A_i^t = w_1^t A_i^{t-1} + w_2^t P_i^{t-1} + w_3^t G_i^{t-1} \tag{8}$$

$$w_1^t + w_2^t + w_3^t = 1 \tag{9}$$

The following values are used in our implementation; $w_1^t = 0.5, w_2^t = 0.2, w_3^t = 0.3$ are considered.

### *2.6 Genetic Algorithm*

Genetic Algorithm (GA) is a search heuristic that mimics the process of natural selection. This meta-heuristic method is routinely used to generate useful solutions to optimization and search problems. Genetic algorithms belong to the larger class of evolutionary algorithms (EA), which generate solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection and crossover. The basic idea of the genetic algorithms is transmission of inherited characteristics through genes. In the GA, the solutions are known as chromosomes. Two processes occur in this algorithm, a mutation process that is random changes in the chromosomes, and the crossover process that is combination of the two chromosomes to produce new chromosomes. Figure 8 shows pseudo-code of the GA [29]. The evolution usually starts from a population of randomly generated individuals, and is an iterative process, with the population in each iteration called a generation. In each generation, the fitness of every individual in the population is evaluated; the fitness is usually the value of the objective function in the optimization problem being solved.

```
1.  Procedure Genetic
2.  Begin
3.      Choose initial population
4.      Repeat
5.          Evaluate the individual cost of the population
6.          Select pairs of individuals to reproduce
7.          Apply crossover operator
8.          Apply mutation operator
9.      Until terminating condition
10. End
```

***Figure 8. Genetic algorithm [19]***

The GA is a population-based algorithm and it works on population of solutions. At the first of the implementation, we generate 30 randomly initial solutions as population. Then the path of each solution is calculated using equation 2. We use the following three mechanisms in order to choose parents.

I. Choosing two best solutions as parents.
II. Choosing two solutions randomly among five best solutions as parents.
III. Choosing two solutions randomly among the whole population.

After choosing chromosomes as parents, we should crossover the parents to create a chromosome in all problem space. In this process, we select two points in two selected chromosomes (in fact divide customers of vehicles to two parts), then the first part of the first chromosome is combined with the second part of the second chromosome and the result is a new chromosome as the child chromosome. The new child chromosome should satisfy the capacity condition. After generating the child chromosome, mutation process should be performed on the new chromosome to probably find the best solution around the current solution. It should be mentioned that after using crossover operator for four times, the chromosomes will be affected once by mutation operator. In other words, crossover operator is going to be used four times more than mutation operator. Figure 9 shows the process of the GA.

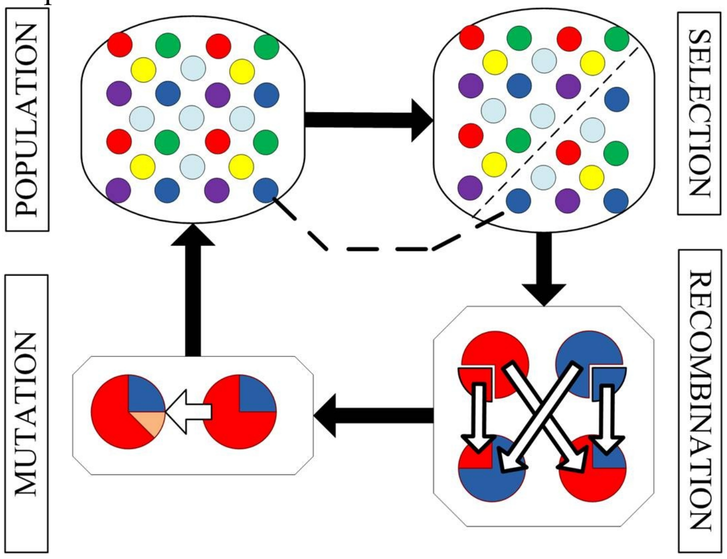

***Figure 9. A diagram of the operation of a simple Genetic Algorithm***

In each iteration, or generation, a population of possible solutions is evaluated and the two solutions are selected as parents of the next generation. The process is repeated so that the fitness of subsequent populations increases. In this way, the likelihood that the population contains the optimal solution also increases.

## 3. Improved Genetic Algorithm

There are many improvements on genetic algorithm. In this paper, in order to improve this algorithm we make the initial population by ant colony algorithm. In fact, the proposed algorithm is a hybrid of Genetic algorithm and Ant Colony algorithms. The reason for improvement of genetic algorithm is that it is better than conventional AI schemes in that it is more robust. Unlike older AI systems, they do not break easily even if the inputs changed slightly, or in the presence of reasonable noise. Also, in searching a large state-space, multi-modal state-space, or n-dimensional surface, a genetic algorithm may offer significant benefits over different typical optimization techniques. Figure 10 shows the flowchart of the proposed algorithm.

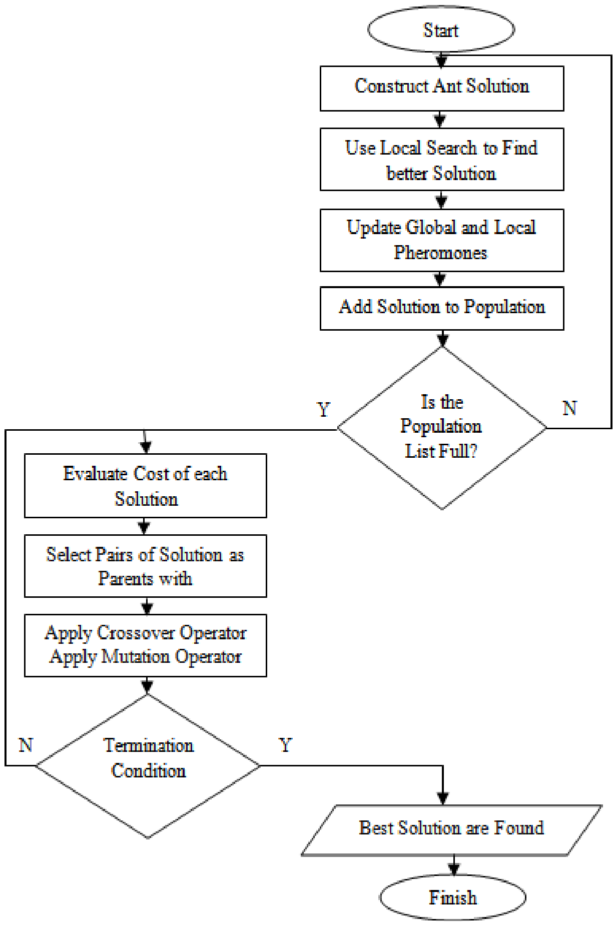
Start
Construct Ant Solution
Use Local Search to Find better Solution
Update Global and Local Pheromones
Add Solution to Population
Is the Population List Full?
Y
N
Evaluate Cost of each Solution
Select Pairs of Solution as Parents with
Apply Crossover Operator Apply Mutation Operator
Termination Condition
N
Y
Best Solution are Found
Finish

***Figure 10. The Flowchart of the Proposed Algorithm***

In the proposed algorithm, we need an initial population to start the genetic algorithm. We use the ant colony algorithm shown in Figure 12. In this procedure, we set different parameters for α and β. In addition, we set different initial pheromones and create ants as solutions. Each ant is a solution for CVRP. After generating solutions, the solutions should be added to the population list. Then, we run genetic algorithm on this population as described in section 2.6. The improved genetic algorithm is shown in Figure 11.

```
1.   Procedure Improved Genetic
2.   Begin
3.         Use Ant Colony algorithm to create initial population (Figure 9)
4.         Repeat
5.              Evaluate the individual cost of the population
6.              Select pairs of individuals to reproduce
7.   Apply crossover operator
8.              Apply mutation operator
9.          Until terminating condition
10.  End
```

***Figure 11. Improved Ant Colony Algorithm***

```
1.  Procedure Ant Colony Algorithm to Create Genetic Population
2.  Begin
3.          Set Parameter, initialize pheromone trails
4.          While population list is not full Do
5.              Construct Ant Solution
6.  Use local search to find better solution
7.              Use global update pheromones
8.              Use local update pheromones
9.  Add created solution to population list
10.         End While
11. End
```

***Figure 12. Ant Colony Algorithm to Create Genetic Population***

In the proposed algorithm, we use local search for detecting the best path around the detected path. The Proposed algorithm uses global and local updates shown in equations 6 and 7 for updating pheromones.

Initial population that is created by the ant colony algorithm has suitable exploration and exploitation that are controlled by its parameters. Hence, this population is better than a randomly generated population. Moreover, in the genetic algorithm where we aim to find the best solution, by use of crossover and mutation operators, exploration and exploitation are controlled, respectively. There are several methods for crossover and mutation operators that are introduced in section 2, can be used in the proposed algorithm. When the operators are more powerful in exploration and exploitation, the proposed algorithm will be more accurate. Therefore, these operators have important roles in the proposed algorithm.

The representation of a solution we use here is an integer string of length n. Each gene in the string is the integer node number assigned to that customer originally. The sequence of the genes in the string is in the order of visiting these customers. For example, consider we have the following solution (as represented in Figure 13):
Route No. 1: $0\rightarrow2\rightarrow3\rightarrow1\rightarrow4\rightarrow0$
Route No. 2: $0\rightarrow5\rightarrow6\rightarrow10\rightarrow0$
Route No. 3: $0\rightarrow8\rightarrow9\rightarrow7\rightarrow11\rightarrow12\rightarrow0$

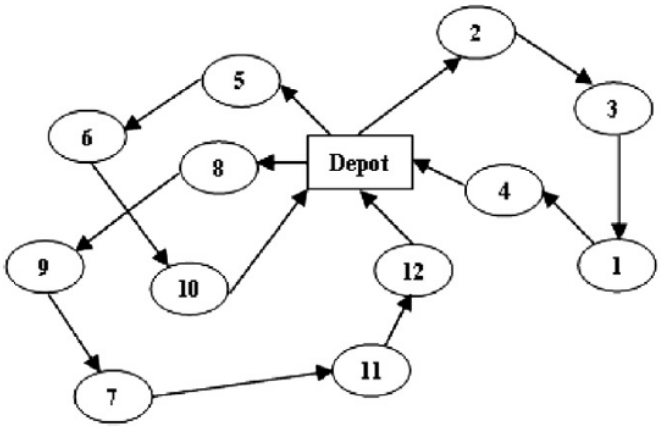

***Figure 13. A Solution for Vehicle Routing Problem***

Crossover operator is usually applied to the selected parents with a crossover probability pc, which could be chosen between 0.60 and 0.95 based on the situation. In this paper, the best results are attained by choosing a 0.75 for pc parameter.

Two crossover operators are applied, i.e., one-point crossover and two-point crossover. A single cross-over point on both parents organism strings are selected. All data beyond that point in either organism string is swapped between the two parent organisms. Two-point crossover calls for two points to be selected on the parent organism strings. Everything between the two points is swapped between the parent organisms, rendering two child organisms.

The mutation operator used in the proposed algorithm consists of applying with equal probability two different mutation operators called inversion and swap sequence operators. The inversion operator reverses the visiting order of the customers between two randomly selected points, while the swap sequence operator consists of randomly selecting two sub-strings of customers and exchanging them. The example of the two mutation operators are given in Figure 14.

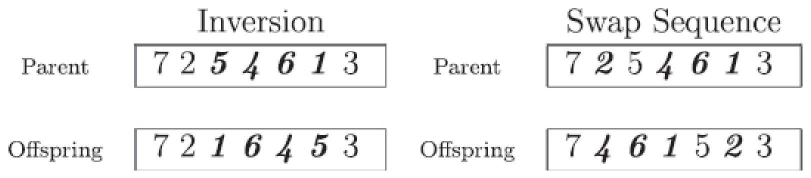

***Figure 14. An Example of two Mutation Operators***

Advantages of the proposed algorithm is made of advantages of GA and ACO algorithms which are faster convergence and smaller danger of getting stuck in local optimum in some applications. It solves problems with multiple solutions. Since the proposed algorithm is not dependent on the error surface, we can solve multi-dimensional, non-differential, non-continuous, and even non-parametrical problems. The proposed algorithm is a method that is very easy to understand and it practically does not require the knowledge of mathematics. The experimental results that will be given in the next section narrate from an improvement by this algorithm on benchmarks data.

## 4. Implementation and Result

In this section, implementation process of the introduced algorithms for solving CVRP is described. All results are obtained by system with frequency 2.5 GHz, core 2 Duo T9300 Intel processor and 4GB of memory that runs final version of windows 7. Also, introduced algorithms have been implemented with the programming language C# in Visual studio 2012.

Vectors are considered for routs of each vehicle and each entry of the vector is customer that should be served. The customers are sorted and should be met one by one. The results for the different algorithms are average often times run of the algorithm within 30 seconds, on standard datasets.

Two factors are important in the implementation to achieve better results. Search the whole problem space (Exploration) and search around locally relevant solutions (Exploitation). In all of the introduced algorithms which are described in this paper, if there was balance between these two factors (exploration and exploitation), we will achieve to better results.

In the simulated annealing algorithm, at the start of the algorithm, temperature is intended to be high, but over time temperature according to the cooling function that is defined in the algorithm will be decreased gradually. This process due to have suitable exploration at the first of the algorithm with high temperature and then suitable exploitation by losing the temperature. Initial temperature is considered T=100 in this implementation. When temperature is decreased, we will have enhancement and reduction on exploitation and exploration, respectively.

In the stochastic hill climbing algorithm, temperature is fixed to T=5. This value is considered empirically. Important points that should be notice are exploration and exploitation of algorithm. We selected T=5 and have suitable exploration and exploitation for the implementation of this algorithm.

There are two parameters in the ant colony algorithm that control exploration and exploitation. These parameters $\alpha$ and $\beta$ experimentally are considered 1 and 2, respectively. In this algorithm, each ant is a solution. An ant is a matrix of vehicles and customers. This algorithm starts from a random customer and then selects the next customers of vehicles by pseudo-random-proportional. Each customer has more percent, will be selected as next customer for serving the vehicles and will be added to matrix. This algorithm has global and local update to control exploration and exploitation.

In the genetic algorithm, crossover and mutation operators have essential roles for solving problems. In this algorithm, crossover process controls exploration and mutation operator improves exploitation by searching local solutions and find best solution around current solution. In the particle swarm algorithm, exploration and exploitation are controlled by changing the weight matrices that are used in the algorithm. Also, this algorithm is usually used for problems with continuous problem space.

The proposed algorithm in this paper is hybrid of genetic algorithm and ant colony algorithm. At the first of the proposed algorithm, we use ant colony algorithm to generate initial ants or solutions for CVRP. In this step, we select different parameters for $\alpha$ and $\beta$ to find different solutions in problem space. We add generated solutions to the population of genetic algorithm, then use of selection mechanism for parent selection. We choose two best solutions as parents. Then use crossover operator and find a point in solution to create a new child solution. At the end, we use mutation operator to improve exploitation of the algorithm and find the best solution around the current solution. Then, we change generated solution to worst solution in population and repeat this procedure again.

Table 1 shows the results of implementation. In order to more easily analyze the results, we made bold and shadow the best and worst results, respectively.

***Table 1. Results of Implementation of the Meta-heuristic Algorithms on CVRP***

| Test Case | Num of Vehicles | Num of Customers | Capacity of Vehicles | Optimal | SA | SHC | AC | TS | PSO | GA | Improved GA |
|---|---|---|---|---|---|---|---|---|---|---|---|
| A-n32-k5 | 5 | 32 | 100 | 784 | **787** | **787** | **787** | **787** | 790 | **787** | **787** |
| A-n45-k7 | 7 | 45 | 100 | 1146 | 1167 | 1163 | 1155 | 1151 | 1157 | 1150 | **1148** |
| A-n55-k9 | 9 | 55 | 100 | 1073 | 1098 | 1102 | 1085 | **1075** | 1085 | **1075** | **1075** |
| A-n61-k9 | 9 | 61 | 100 | 1034 | 1094 | 1167 | 1170 | **1045** | 1110 | 1065 | 1050 |
| A-n63-k9 | 9 | 63 | 100 | 1616 | 1663 | 1665 | 1680 | **1640** | 1700 | 1655 | 1650 |
| A-n80-k10 | 10 | 80 | 100 | 1763 | 1868 | 1870 | 1690 | 1825 | 1830 | 1820 | **1814** |
| P-n19-k2 | 2 | 19 | 160 | 212 | **212** | **212** | **212** | **212** | 220 | **212** | **212** |
| P-n45-k5 | 5 | 45 | 150 | 510 | 542 | 545 | 515 | 530 | 540 | **514** | **514** |
| P-n55-k15 | 15 | 55 | 70 | 989 | 1025 | 1030 | 1250 | 1150 | 1174 | 1050 | **1020** |
| P-n101-k4 | 4 | 101 | 400 | 681 | 892 | 915 | 800 | 990 | 980 | **725** | **725** |

The improved genetic algorithm has the best results in the eight test cases and has not worst results in the tests. It shows that the proposed algorithm has suitable exploration and exploitation using crossover and mutation operators. Also, the results show that the population of the genetic algorithm is important and having appropriate solutions in the population will lead to better results. The results show that the less number of cities, the better result are achieved by the proposed method. Moreover, it can be seen that the topology of customer network is important to get optimal results. It could be improved of original genetic algorithm. Therefore, we select the proposed algorithm as best algorithm among all of the introduced algorithms for solving capacitated vehicle routing problem.

The performance of GA largely depends on the proper selection of its parameters values; including crossover mechanism, probability of crossover, population size and mutation rate and selection percent. Also, the parameters of the ACO algorithm determine the behavior of each ant and are critical for fast convergence to near optimal solutions of the problem. The basic parameters that are used in the ACO algorithms are the relative importance (or weight) of pheromone, the relative importance of heuristics value, initial pheromone value, evaporation rate, and a parameter to control the exploration or exploitation sides. We evaluate the proposed algorithm with different value of the mentioned parameters. Table 2 to Table 5 show the sensitivity analysis of the improved GA with different parameters.

***Table 2. The population size evaluation***

| Test Case | Optimal | Size of Population | α | β | Crossover Probability | Mutation Probability | Improved GA |
|---|---|---|---|---|---|---|---|
| A-n45-k7 | 1146 | 10 | 1 | 2 | 0.75 | 0.25 | 1159 |
| | | 20 | 1 | 2 | 0.75 | 0.25 | 1155 |
| | | **30** | 1 | 2 | 0.75 | 0.25 | **1148** |

The population of chromosomes consists of some solutions. As seen in Table 2, when the population has more solutions, the probability of finding better solutions will be increased. Consequently, the best result will be obtained through having bigger population.

***Table 3. The α value evaluation***

| Test Case | Optimal | Size of Population | α | β | Crossover Probability | Mutation Probability | Improved GA |
|---|---|---|---|---|---|---|---|
| A-n45-k7 | 1146 | 30 | **1** | 2 | 0.75 | 0.25 | **1148** |
| | | 30 | 3 | 2 | 0.75 | 0.25 | 1162 |
| | | 30 | 5 | 2 | 0.75 | 0.25 | 1176 |

Based on the obtained results by the experiments that are shown in Table 3, it is clear that as α parameter increases, the greater possibility of ants choose the path belong to the previous tour will be, in the new tour. In the ant system α indicates the importance of the collected pheromones on the edges.

***Table 4. The β value evaluation***

| Test Case | Optimal | Size of Population | α | β | Crossover Probability | Mutation Probability | Improved GA |
|---|---|---|---|---|---|---|---|
| A-n45-k7 | 1146 | 30 | 1 | 1 | 0.75 | 0.25 | 1163 |
| | | 30 | 1 | **2** | 0.75 | 0.25 | **1148** |
| | | 30 | 1 | 3 | 0.75 | 0.25 | 1159 |

As the β parameter increases, the possibility of choosing the local shortest path by ants will be increased as it is shown in table 4. To get the optimal solution, algorithm must have a tradeoff between global optimization and fast convergence.

***Table 5. The crossover and mutation probabilities evaluation***

| Test Case | Optimal | Size of Population | α | β | Crossover Probability | Mutation Probability | Improved GA |
|---|---|---|---|---|---|---|---|
| A-n45-k7 | 1146 | 30 | 1 | 2 | 0.25 | 0.75 | 1185 |
| | | 30 | 1 | 2 | 0.50 | 0.50 | 1161 |
| | | 30 | 1 | 2 | **0.75** | **0.25** | **1148** |

The crossover and mutation operators adjust the exploration and exploitation of the algorithm. As seen in the Table 5, crossover probability value 0.25 does not have a suitable exploration. In contrast, the best values for crossover probability and mutation probability are 0.75 and 0.25, respectively.

Another experiment is comparison of the proposed algorithm with related algorithms that were introduced in the section 1. The results that are reported for each instance are computed according to the following equation:

$$RPD = \frac{Method_{Solution} - Optimal_{Solution}}{Optimal_{Solution}} \times 100\% \qquad (10)$$

Where, RPD is the relative percentage deviation of the solution obtained (denoted as $Method_{solution}$) from the best published solution (denoted as $Optimal_{solution}$).
Table 6 presents the computational results. For each instance in Table 6, the results are presented in the form of RPD for the best solution obtained. A RPD value of 0.00% indicates that the optimal solution is obtained.

***Table 6. Comparison of computational results of the proposed algorithm and related algorithms***

| Test Case | Num of Vehicles | Num of Customers | Capacity of Vehicles | Optimal | Improved GA | Chen [9] | Tan [10] | Marikanis [12] |
|---|---|---|---|---|---|---|---|---|
| A-n32-k5 | 5 | 32 | 100 | 784 | 0.38 | 0.72 | 0.37 | 0.43 |
| A-n45-k7 | 7 | 45 | 100 | 1146 | **0.17** | 0.54 | 0.31 | 0.33 |
| P-n19-k2 | 2 | 19 | 160 | 212 | **0.0** | 0.0 | 0.0 | 0.0 |
| P-n45-k5 | 5 | 45 | 150 | 510 | **0.78** | 1.2 | 0.85 | 0.91 |
| B-n77-k10 | 10 | 77 | 100 | 1221 | **0.89** | 1.4 | 0.93 | 0.92 |
| F-n134-k7 | 7 | 134 | 2210 | 1162 | 2.1 | 4.5 | 2.0 | 2.31 |
| C-n75-k10 | 10 | 75 | 140 | 835 | **0.83** | 1.9 | 1.44 | 1.35 |
| C-n100-k11 | 11 | 100 | 200 | 866 | **0.75** | 1.15 | 0.77 | 0.82 |

In order to more easily analyze the results, in Table 6, we made bold and shadow the best and the worst results, respectively. As seen in Table 6, the Improved Genetic algorithm has the best results in the six test cases out of eight test cases. The results are compared with the three algorithms that are proposed by Chen [9], Tan [10] and Marikanis [12]. The results are compared with the Best Known Solution (Optimal Solution) of the data set. In this data set, the number of cities varies from minimum of 19 to maximum of 134 and the number of vehicles varies from minimum of 2 to maximum of 11. The comparison shows, the algorithm proposed by Chen et al. [9] has the most deviation from the optimal solution results in this comparison.

The Improved Genetic algorithm has deviation but it is less than others. The deviation from the optimal result depends on some parameters such as topology of cities, number of cities, capacity and so on. By increasing the number of cities, deviation value will be increased. Therefore, the best deviation value is achieved by testing the proposed method with least number of cities as well as the highest deviation value is achieved by the maximum number of cities. Comparison graphs of those results are depicted in Figure 13.

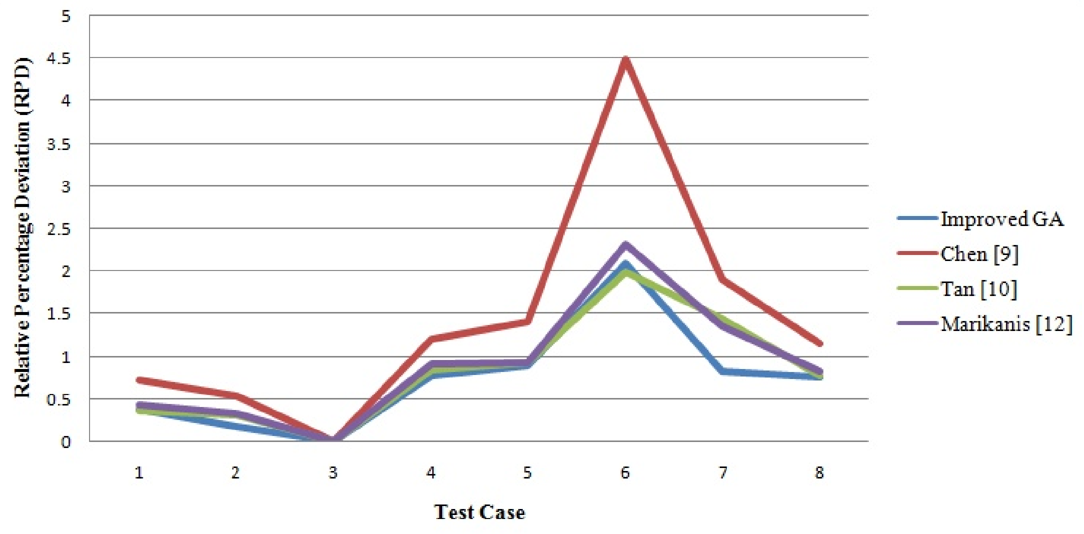

***Figure 13. Comparison Graphs of the Results***

Figure 13 shows the comparison graphs of the results of algorithms for solving CVRP. The RPD of the best result is equal to 0. The results of Improved GA, Chen [9], Tan [10] and Marikanis [12] are shown with blue, red, green and purple, respectively. The red curve for most parts is lower than other curves. It shows that it has less deviation of the optimal results.

## 5. Conclusion and Future work

CVRP is a VRP in which a fixed fleet of delivery vehicles of uniform capacity must service known customer requests for a single commodity from a common depot at minimum transit cost. That is, CVRP is like VRP with the additional constraint that every vehicle must have uniform capacity of a single commodity. There are many algorithms to solve this problem. In this paper, we analyze, compare, implement and evaluate six meta-heuristic algorithms to solve CVRP. The genetic algorithm was shown to be the best algorithm among all meta-heuristic algorithms and we proposed as improved version of this algorithm, as well. In the original genetic algorithm, initial population is randomly generated, while we proposed in this work the use of ant colony algorithm to generate the initial population. All of the meta-heuristic methods as well as the proposed algorithm were tested on ten standard benchmarks. Also, three related works are tested on benchmarks. We used local search in all of the implementations and got improvements on the results. Some algorithms are suitable for continuous problems and others are for discrete problems. Since the CVRP is a discrete problem, the combination of GA and AC algorithms was shown to be suitable for this problem. On the other hand, some algorithms like PSO are suitable for continuous problems. The experimental results show improvement of the proposed genetic algorithm. The proposed algorithm had the best results in eight test cases against six meta-heuristic algorithms and obtained best results in the six test cases out of eight test cases against the three related algorithms. Non-deterministic algorithms like the genetic algorithm and the improved genetic algorithm proposed in this paper were shown to be suitable to solve NP-Hard problems such as CVRP, in an acceptable time.

We want to solve CVRP by bee colony algorithm and other improvements on meta-heuristic algorithms in the future works. Also, the algorithms could be evaluated on other standard test cases and benchmarks for greater certainty.

## References

[1] S. Nanda Kumar, and R. Panneerselvam, "A Survey on the Vehicle Routing Problem and Its Variants.," *Intelligent Information Management,* Vol. 4, No. 3, pp. 66-74, 2012.

[2] P. Toth, and A. Tramontani, "An integer linear programming local search for capacitated vehicle routing problems," in The Vehicle Routing Problem: Latest Advances and New Challenges, Springer, New York, USA, pp. 275-295, 2008.

[3] H. Nazif, and L. S. Lee, "Optimised crossover genetic algorithm for capacitated vehicle routing problem," Applied Mathematical Modelling, Vol. 36, No. 5, pp. 2110-2117, 2012.

[4] W. Y. Szeto, Y. Wu, and S. C. Ho, "An artificial bee colony algorithm for the capacitated vehicle routing problem," European Journal of Operational Research, Vol. 215, No. 1, pp. 126-135, 2011.

[5] P. Chen, H.-K. Huang, and X.-Y. Dong, "Iterated variable neighborhood descent algorithm for the capacitated vehicle routing problem," Expert Systems with Applications, Vol. 37, No. 2, pp. 1620-1627, 2010.

[6] E. Emel. Yurtkuran, “A new hybrid electromagnetism-like algorithm for capacitated vehicle routing problems,” Expert Systems with Applications, Vol. 37, No. 4, pp. 3427-3433, 2010.

[7] A. Juan, J. Faulin, R. Ruiz, B. Barrios, and S. Caballé, “The SR-GCWS hybrid algorithm for solving the capacitated vehicle routing problem,” Applied Soft Computing Journal, Vol. 10, No. 1, pp. 215-224, 2010.

[8] H. Q. Zheng, Y. Q. Zhou, and Q. F. Luo, “A hybrid cuckoo search algorithm-GRASP for vehicle routing problem,” Journal of Convergence Information Technology, Vol. 8, No. 3, 2013.

[9] K. Kanthavel, and P. Prasad, "Optimization of Capacitated Vehicle Routing Problem by Nested Particle Swarm Optimization.," *American Journal of Applied Sciences,* Vol. 8, No. 2, pp. 107-112, 2011.

[10] W. F. Tan, L. S. Lee, Z. A. Majidi, and H. W. Seow, "Ant Colony Optimization for Capacitated Vehicle Routing Problem.," *Journal of Computer Science,* Vol. 8, No. 6, pp. 846-852, 2012.

[11] H. Lei, G. Laporte, and B. Guo, "The Capacitated Vehicle Routing Problem with Stochastic Demands and Time Windows.," *Computers & Operations Research,* Vol. 38, No. 12, pp. 1775-1783 2011.

[12] Y. Marinakis, and M. Marinaki, "A Hybrid Genetic–Particle Swarm Optimization Algorithm for the Vehicle Routing Problem.," *Expert Systems with Applications,* Vol. 37, No. 2, pp. 1446-1455, 2010.

[13] S. Mazzeo, and I. Loiseau, "An Ant Colony Algorithm for the Capacitated Vehicle Routing.," *Electronic Notes in Discrete Mathematics,* Vol. 18, pp. 181-186, 2004.

[14] Yu, Z. Z. Yang, and B. Yao, "An Improved ant Colony Optimization for Vehicle Routing Problem.," *European Journal of Operational Research,Vol.* 196, No. 1, pp. 171-176, 2009.

[15] E. Gounaris, W. Wiesemann, and C. A. Floudas, “The robust capacitated vehicle routing problem under demand uncertainty.,” *Operations Research*, Vol. *61, No.* 3, pp. 677-693,2013.

[16] R. Baldacci, E. Hadjiconstantinou, and A. Mingozzi, “An exact algorithm for the capacitated vehicle routing problem based on a two-commodity network flow formulation.,” *Operations Research*, Vol. *52, No. 5*, pp. 723-738, 2004.

[17] G. Shang, J. Xinzi,and T. Kezong, “Hybrid Algorithm Combining Ant Colony Optimization Algorithm with Genetic Algorithm”, in Proc. of the 26th Chinese Control Conf, 2007.

[18] H. Duan, and X. Yu, “Hybrid Ant Colony Optimization Using Memetic Algorithm for Traveling Salesman Problem”, in Proc. IEEE Int. Symposium. Dynamic Programming and Reinforcement Learning, 2007.

[19] X. Jin-rong, L. Yun, L. Hai-tao, and L. Pan, "Hybrid Genetic Ant Colony Algorithm for Traveling Salesman Problem", *Journal of Computer Applications*, Vol. 28, No. 8, pp. 2084-2112, 2008.

[20] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi, "Optimization by Simulated Annealing". *Science, Vol.* 220, No. 4598, pp. 671–680, 1983.

[21] V. Cerny, "Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm.," *Journal of Optimization Theory and Applications,* Vol. 45, No. 1, pp. 41–51, 1985.

[22] S. Luke, Essentials of Metaheuristics., http://cs. gmu. edu/∼ sean/book/metaheuristics, 2010.

[23] Z. Michalewicz, and D. B. Fogel, How to Solve it: Modern Huristics., Springer, 2004.

[24] M. Dorigo, *Optimization, Learning and Natural Algorithms*, PhD thesis, Politecnico di Milano, Italy, 1992.

[25] M. Dorigo, and L. M. Gambardella, "Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem.," *Evolutionary Computation, IEEE Transactions, Vol.* 1, No. 1, pp. 53-66, 1997.

[26] F. Glover, "Future Paths for Integer Programming and Links to Artificial Intelligence.," *Computers & Operations Research, Vol.* 13, No. 5, pp. 533-549, 1986.

[27] J. F. Kennedy, J. Kennedy, and R. C. Eberhart, Swarm Intelligence., Morgan Kaufmann, 2001.

[28] B. I. Kim, and S. J. Son, "A probability Matrix Based Particle Swarm Optimization.," *Journal of Intelligent Manufacturing, Vol.* 23, No. 4, pp. 1119–1126, 2012.

[29] R. R. Sharapov, "Genetic Algorithms: Basic Ideas, Variants and Analysis.," *Vision Systems: Segmentation and Pattern Recognition,* pp. 407-422, 2007.