Proceedings of the IEEE
International Conference on Automation and Logistics
August 18 - 21, 2007, Jinan, China

A Genetic Algorithm for the Capacitated Arc Routing Problem*

Xin Deng , Zhengyu Zhu, Yong Yang, Xiaohua Li, Yunyan Tian, Mengshuang Xia

Institute of Computer Science
Chongqing University
Chongqing, 400044, P.R.China

dx168@yeah.net

***Abstract* - In this paper, the Genetic Algorithm(GA) is used to resolve the Capacitated Arc Routing Problem(CARP), the case of which stems from assigning the routing of sprinkler cars in real life. And the paper presents basic components of GA combined with the crossover, mutation and loach search operation. These new methods are helpful for the global search of CARP problem. The test data is coming from real world, offered by the Sanitation Department-the owner of the sprinkler cars. Furthermore, the contrastive experiment which has been done in this paper illustrates that this GA method is more excellent than other existing algorithms in resolving this kind of CARP. As for the pratical meaning of this research, the routing plan proposed may provide high efficiency for the department and will bring about significant economic benefits.**

Index Terms - CARP, Genetic Algorithm, Sprinkler Car.

I. INTRODUCTION

A. The CARP

The Capacitated Arc Routing Problem (CARP), which was introduced by Golden & Wong (1981)(Ref.[1]), is a distribution problem with many practical applications, just as postal mail delivery, school bus routing, road weeping, winter gritting and electricity wire examination.

The CARP can be described as follows. We are given an undirected, connected graph $G=(V, E)$ with vertex set V and edge set E. Every edge $e \in E$ has a non-negative cost c_e and a non-negative demand for service q_e. The edges with positive demand make up the subset of the required edges E_R which have the number of ε .Given a vehicle capacity W , the CARP consists of finding a set of vehicle routes T_1, T_2,.., T_m ($m \le K$ where m is a decision variable and K is given by application), called its solution, with minimum total cost. In the solution, every required edge is serviced by exactly one vehicle, each route starts and ends at a pre-specified vertex v_0 (the depot), the total demand serviced by a route does not exceed the vehicle capacity W and $m \le K$.

The CARP belongs to a series of network optimization problems, and it is a NP-hard problem. Ref.[1] showed that even the 0.5 approximate CARP (find a solution whose cost is more than 1.5 times the optimal cost) is NP-hard. Furthermore, finding a feasible solution of the CARP with a given number of vehicles is also NP-hard. So the interest focuses on the development and implementation of heuristic algorithm.

B. Conversion

In this pager, to optimize the routes of the sprinkler cars is our objective. It's an extended CARP problem. As our cognition, existent published papers have no algorithm to deal with the sprinkler cars' routing problems. So many papers focus on the routing problem with time windows and the original CARP which result in no ready-made algorithms suit for it(see Ref.[2,3,4,5,6]). We must change the existed algorithm to suit for the problem, and it's the first time to attempt to the sprinkler cars' routing problem. The sprinkler cars which belonged to one Sanitation Department of the Chongqing city in China have the task to sprinkle the roads of the district. In the real road condition, one road must be sprinkled twice by cleaning the left side and then the right side. So we should construct a CARP solution in a directed graph $G'=(V, E')$ instead of the original undirected graph $G=(V, E)$. The reason is that the direction of traveling along an arc is uniquely defined but for the edges. The graph G' has the same vertex set V as G, the arc set E' contains edge $e=(v_i, v_j) \in E$. The arcs $a=(v_i, v_j)$ and $a^R=(v_j, v_i)$, both with cost c_e and demand q_e, are called representatives of edge e. Supposed $a=(v_i, v_j)$ is an arc, we define a function *reversal()* such that, $reversal(a)=a^R$ and $reversal(a^R)=a$.

So, in G, a solution of CARP can be described as a set of routes T_1, T_2,.., T_m, each route T_i is represented by (σ_0, σ_1, σ_2, . . . σ_t, σ_0), where v_0 is the depot vertex; each σ_j (j=1, 2, . . ., t) denotes an arc.

Since the road with smallest cost are uniquely existed by the order of the arcs, so here and later we will always represent a route with a sequence of the arcs of all serviced edges without appearing any deadheading.

II. MATHEMATICAL MODE

$$Min\sum_{i=1}^{K} Cost(T_i) \qquad (1)$$

s.t.

$$\text{load}(T_i)=\sum_{j=1}^{|T_i|} q(T_{ij}) \le W_i \qquad (2)$$

* Project supported by the Specialized Research Fund for the Doctoral Program of Higher Education of China(Grant No. 20030611016).

$$\text{cost}(T_i) = D(\sigma, T_{i1}) + \sum_{j=1}^{|T_i|-1}(\omega(T_{ij}) + D(T_{ij}, T_{i,j+1})) + \omega(T_{i,|T_i|}) + D(T_{i,|T_i|}, \sigma)) \quad (3)$$

$$0 \le |T_i| \le \varepsilon \ (i=1,2,\ldots,K) \quad (4)$$

$$\sum_{i=1}^{K} |T_i| = \varepsilon \quad (5)$$

$$\begin{cases} R_k = \{T_{ki} \mid T_{ki} \in \{T_{k1}, \ldots, T_{k|T_k|}\}, k = 1,2,\ldots,K\} \\ R_i \cap R_j = \phi (i \ne j; i, j = 1,2,\ldots,K) \end{cases} \quad (6)$$

(1)There are K routes in a feasible result, T_i stands for the route i.(2) The total demand of each route aught to less than the capability W of a vehicle.(3)The Cost Function depict the cost of each route.(4)The total number of each route's arcs can't exceed the total number of service arcs ε .(5)Make sure that every service arcs must be serviced.(6)The set R_k assembles the arcs in the route k. Different route can't service the same arcs.

III. A NEW GA METHOD WITH LOCAL SEARCH FOR CARP

A. Initialization

To initialize the data, we should input the graph G which includes the arcs' deadheading cost C and demand q firstly, then the fleet size K, vehicle capacity W. Chromosome number are inputted into the computer too. After that, the Floyd Algorithm is used to get the shortest distance array D between each pair of node and the predecessor array P which notes the routes.

1) Population and its chromosomes: Each possible solution *S* for CARP, containing m routes, T_1, T_2,.., T_m, ($m \le \boldsymbol{K}$), can also be regarded as containing $\boldsymbol{K}$ routes, T_1, T_2,.., T_h, if allowed empty routes existed in it. Here an empty route means the route (σ_0, σ_0) contained no serviced arcs. Supposed $R_i=(\sigma_0, \sigma_{i_1}, \sigma_{i_2}, \ldots, \sigma_{it_i}, \sigma_0)$, for i=1, 2, …, K, we can encrypted the solution S as a chromosome. Use an array, directly in the form of Chrom = $(\sigma_{11}\sigma_{12}\ldots\sigma_{1t_1}\sigma_{21}\sigma_{22}\ldots\sigma_{2t_2}\ldots\sigma_{h1}\sigma_{h2}\ldots\sigma_{ht_h})$, to depict all serviced arcs. Constraint: since each chromosome is corresponding a feasible solution S of CARP, so it must such that, the number of routes contained in $\boldsymbol{S}$ is always less than $\boldsymbol{K}$ and the total capabilities serviced by each route of $\boldsymbol{S}$ aught to less than the capability $\boldsymbol{W}$ of a vehicle.

2) Population and initialization: In this paper, the population $\boldsymbol{P}$ is always composed of $\boldsymbol{cn}$ chromosomes, where $\boldsymbol{cn}$ is the size of population and is a control parameter, which will be adjusted in our experiments. Commonly $5 \le cn \le 30$ .The initial population P is composed of $\boldsymbol{cn}$ chromosomes. Each of chromosome is produced randomly by the order of the arcs. For example, if there is a graph with 5 service arcs, after the initialization process, the chromosome may be (2,1,3,5,4). But it means nothing before the feasible process in chapter B of III.

3) Fitness values: Each chromosome has it's own cost value, in order to let the data fit for the statistic, this paper induce the fitness value obtained by Fit[i]=1/Cost[i]. The chromosome i has the fitness value Fit[i] which is the reciprocal of cost value of the chromosome i. The bigger the fitness value is, the better performance of the chromosome has.

B. Feasible process

Algorithm Split which refers to the Ref.[7] has been changed in this paper with an $O(\tau^2)$ time complexity working on a given chromosome S. It minimizes total cost (subject to the sequence of tasks defined by the chromosome) and, as a secondary objective, the number of vehicles. It runs in $O(\tau)$ space only, by avoiding an explicit generation of the auxiliary graph. Two labels are used for each node i of G: Vi (cost of the shortest path from 0 to i in G) and Predi (predecessor arcs array).

```
Procedure Split(Chrom[Num])
  V[Num,0]:=0, Pred [Num,0]:=0;
  for i:=1 to n  do  V[Num,i]:=65535;
  for i:=1 to n do
    load:=0,cost:=0;j:=i;
    repeat
      load:=load+q[Chrom[Num,j]];
      if i=j then
        cost:=D[σ,chrom[Num,j]]+
              (chrom[Num,j])+D[Chrom[Num,j],σ]
      else
        cost:=cost-D[Chrom[Num,j-1],σ]+
              D[Chrom[Num, j-1],chrom[Num,j]]+
              C(chrom[Num,j])+D[Chrom[Num,j],σ];
      endif
      if (load<=W) then
                  VNew:=V[Num,i-1]+cost;
      if (VNew<V[Num,j]) or ((VNew=V[Num,j])
                  and (Pred [Num,i-1]+1<
                  Pred [Num,j])) then
        V[Num,j]:=VNew;
        Pred [Num,j]:=i-1;
        endif;
        j:=j+1;
      endif;
     until (j>n)or(load>W);
    end;
end;
```

After the Split procedure, a chromosome is decoded. Once transfer the array Pred, we can get the routes. The last member of the array V records the cost of the chromosome.

C. Selection Procedure

According to the Ref.[7] the "Binary tournament method" which includes two steps is used to select two chromosomes from the population. Step 1: select two chromosomes from the population randomly and then let the highest chromosome to be the first operated one. Step 2: repeat step 1 to find out the second one.

D. Crossovers

Given two parents P1 and P2 with τ tasks(Ref.[8,9]), both crossovers draw two cutting sites p and q with $1 \le p \le q \le \tau$. To get the first child C1, copie P1(p), . . . , P1(q) into C1(p), . . . , C1(q). P2 is gained by scanning from left to right and filling the tasks missing in C1 into C1(q+1), . . . , C1(τ), and then C1(1), . . . , C1(p-1). For this crossover operation, the other child C2 is obtained by exchanging the roles of P1 and P2. The example as the follow:

P1:8 4 5 7 9 6 1 3 2 ⟶ P1:8 4 | 5 7 9 6 |1 3 2
P2:4 6 8 3 2 5 9 1 7 ⟶ P2:4 6 8 3 2 5 9 1 7 ⟶

C1:* * | 5 7 9 6 |* * * ⟶ C1: 2 1 |5 7 9 6 |4 8 3
P2:4 6 8 3 2 5 9 1 7

E. Mutation

Given a parent P with τ tasks, mutation draws two cutting sites p and q with $1 \le p \le q \le \tau$. To get the child C, mutation exchanges P (p) and P (q).

F. Local Search (LS)

LS(see Ref.[10,11]) performs successive phases that exams all pairs of tasks (u,v) to try the following moves (the five LS operators) in O(τ^2). Each phase ends by the first improving move detected or when all pairs (u,v) are examined. LS stops when a phase reports no improvement.

N1: check arcs u and v, if *reversal(u)=v* then exchange u and v ,else exit N1.

N2: move u after v.

N3: Move the sub-sequence between the two arcs(u, x) after v. x is a random number and u＜x＜v, but u can't next to v.

N4: See Fig.1.

Case1: u and v are in the same route. If the selected arcs are u(x next to u) and v(v next to y), let v after u, v connect to x, not change the arcs between x and y, then connect y to the arc which after v before the operation. Note that if u connects to v or u is the second final arc, then N4 is forbid.

Case2: u and v are not in the same route. Arcs u(x next to u) and v(y next to v), connect u to y and v to x. Note that if u and v are both the last arcs in theirs routes then discard N4. If u(v) is the last arc in one route, we should consider x(y) as the depot.

N5: See Fig.2.

Case1: u and v are in the same route. Arcs u(x next to u) and v(v next to y), let u connects to v, v connects to y, reverse the arcs between y and x. At last connect x to the arcs which originally after v.

Case2: u and v are not in the same route. Arcs u(x next to u) and v(y next to v), connect u to v, then connect v to it's front arcs which must be conversed. Converse the arcs after v and then connect y to x. Note that if v is the first or the last arcs in one route then discard N5.

During the LS operation, it may cause illegal chromosomes. If this happens, then exit the current phase, discard the illegal chromosome and to do the next phase.

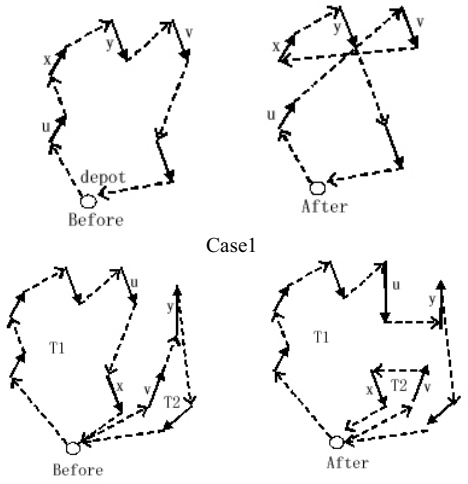

Fig.1 The phase of N4 in Local Search

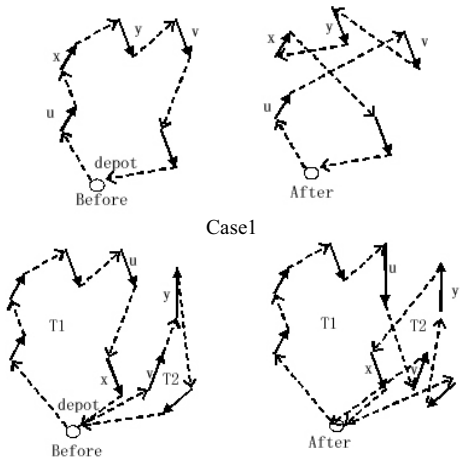

Fig.2 The phase of N5 in Local Search

G. The main procedure for CARP

```
Main( )
{
   initiate( );/ *see chapter III A*/
   Iterate_number:=1;
  Termination:=False;
  While not Termination do
   {
     No_change:=1;
     While No_change<NoC and Not Termination do
       { Selection();  /*see chapter III C*/
         OX( ); (crossover operators)
                /* probabilities 0.7, see chapterIII D*/
         Mutation();   /* probabilities 0.2*/
         Iterate_number++;
         If no_improvement( ) then
             No_change++;
         Else
             LS( ); (Local search)
```

```
            If LB_has_been_reached( ) then
                Termination:=True;
             Endif
          Endif
          If Iterate_number=ItN then       /* ItN=20000 */
             Termination:=True;
           Endif
        }   /* While */
        if  No_change=NoC then
            restart( );
        endif
    }       /* While */
  output( );
}         /* Main() *
```

Algorithm 1 The main structure of the GA for CARP

The function **no_improvement( )** is used to detect if any improvement in the current population P has been got after an operator has been executed. The function **LB_has_been_reached( )** is used to detect if a known lower bounder has been reached. The function **restart( )** is used to restart the system from its beginning when the population P has no change during a large number of iterations. The **function output( )** is used to output the result of the algorithm, including the routes of its optimized solution, the ordered arcs of each route.

IV. CASE STUDY

This study refer to a real life problem of sprinkler cars routes programming in a Sanitation Department of Chongqing City in China. We abstract the real road map to the topology figure. There are 37 vertexes and vertex 1 is the depot point. See Fig.3. The experiment condition is AMD 800MHz CPU, 128M SDR Memory, Windows2000 Professional and Delphi 7.0.

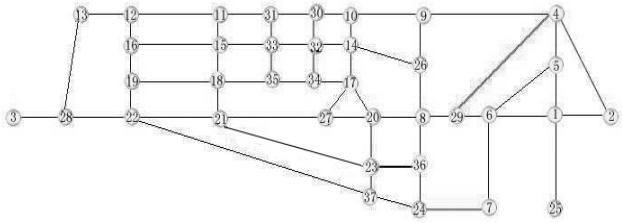

Fig.3 The map of the district

The role of the Sanitation Department is to water the street in the district. For the department, there are 8 sprinkler cars which have the same capability of 8-ton. Before the case study, the cars' routes is assigned by the experience of the car drivers. Everyday they will cost 95 KM. But after our experiment, its cost reduces to about 64 KM one day. The decrease is 33%. For them , they can save 40 thousand Yuan every year.

We also test the capability of the algorithm based on the data. Firstly we test the probability of the LS. See Fig.4. The abscissa stands for the iteration number and the y-axis stands for the cost of the Chromosome. From the figure we can conclude that the probability of the LS is 70% should to be the best.

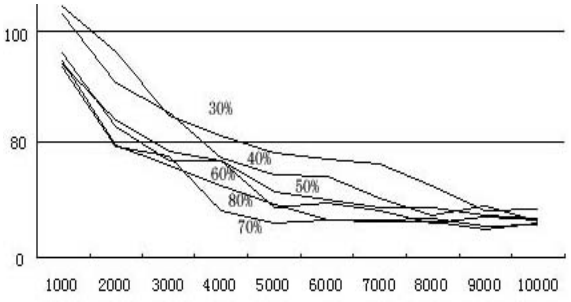

Fig.4 The influence of the LS probability

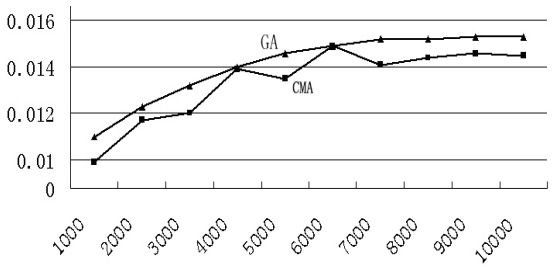

Fig.5 The fitness value in different iteration number

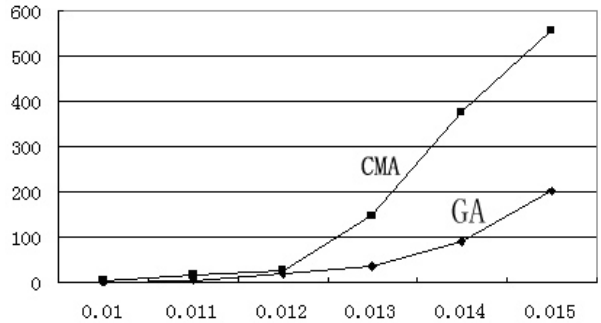

Fig.6 The cost of the time by two algorithms

TABLE I
THE EVOLUTION SPEED OF THE CHROMOSOM

	1	2	3	4	5	6	7	8	9	10	Average
1000	90.71	94.48	92.31	94.58	94.02	94.54	93.68	92.54	92.76	91.94	93.16
2000	84.56	78.43	76.59	79.60	82.54	81.72	80.73	83.13	80.87	81.135	80.93
3000	75.18	75.04	74.12	78.26	72.50	78.29	77.92	75.77	74.25	75.45	75.68
4000	69.35	67.78	70.25	73.25	70.59	70.20	73.02	73.26	72.437	72.69	71.28
5000	67.81	68.14	67.72	70.43	68.86	71.91	67.58	66.20	68.98	67.03	68.47
6000	65.85	66.30	64.88	64.77	69.94	67.56	66.23	65.04	68.31	67.77	66.67
7000	65.11	64.77	66.69	65.21	65.26	65.65	65.53	65.66	64.37	68.00	65.63
8000	66.48	66.17	66.69	63.77	64.86	66.10	66.44	67.31	65.40	65.73	65.90
9000	66.64	68.52	65.68	64.66	66.00	64.42	66.23	64.59	63.98	64.83	65.56
10000	64.56	67.57	65.95	64.50	63.97	66.99	65.040	65.09	65.94	65.21	65.48

TABLE II
THE CONSTRAST RESULT

		1	2	3	4	5	6	7	8	9	10	Average
GA(to get 64)	Iteration number	7120	7052	7031	8061	7654	7304	8037	6891	7210	7697	7406
	Running time	12m9s	12m2s	12m19s	12m12s	12m9s	10m3s	12m24s	8m53s	9m43s	9m7s	11m13s
CMA(to get 67)	Iteration number	9695	9423	8990	10383	10023	9845	9632	9127	10165	9856	9714
	Running time	30m45s	29m23s	24m52s	35m34s	28m34s	27m32s	29m58s	28m1s	31m17s	27m21s	29m18s

Secondly, we test the evolution speed of the chromosome. See TABLE I. The x-axis stands for the test number of the algorithm and the y-axis stands for the different iteration number. From this table we can get that: (1) The capability of getting the best result is increasing followed the adding of the iteration number. (2)The more the iteration number is, the better result it will get, and the less cost of the chromosome followed by the fitness value increase.

In order to illustrate this algorithm's powerful performance, we contrast it to the "Competitive Memetic Algorithms"(See Ref.[7] , shorts for CMA) based on our data for the excellent outperform of CMA than other former algorithms in CARP. By computing the data in the same computer circumstance, we get the better performance by our algorithm. In our algorithm(short for GA), the lowest cost is about 64 Km contrast to the lowest cost about 67 Km which gotten by the CMA. After that we test the fitness value in different iteration number(See Fig.5) by two kinds of algorithms. The x-axis is the different iteration number from 1000 to 10000 step by 1000.The y-axis depicts the fitness value of the chromosome which is the average value of ten times running. From the picture we can conclude that the GA has the better performance than the CMA. We also test the cost of the time by two algorithms(See Fig.6).In the figure, the x-axis is the different fitness value of the chromosome, the y-axis is the cost of the running time(second) of the computer. From this figure we can conclude that to get the same fitness value the CMA costs more time than the GA. At last we test the time and the iteration number of the two kind of algorithms to get the best results of their own. See TABLE II. The first row with iteration number and the second row with running time stand for the GA algorithm to get its best result with 64 Km. Either the third row and the last row depict the iteration number and the running time of the CMA algorithm with its lowest cost of 67 Km. From the table we know that the average iteration number is 7046 and the average running time is 11m13s of the GA algorithm contrasting to the CMA with 9714 average iteration number along with 29m18s running time. So we can conform that the GA is better than the CMA in resolving this kind of CARP.

V. CONCLUTION

The Genetic Algorithm for CARP presented in this paper well performs the real life routing assigning problem of the sprinkler cars. When designing our method, we based on the GA method given by Ref.[7] sometimes. But we also introduce some new techniques to improve it. The excellent performance of our method results from two aspects: 1) Settle and modify the CARP mathematical model. 2) Improve its Split procedure to fit into this kind of extended CARP. 3) A different design of the local search operators. 4)Contrast to the best-known algorithm in our data and get better result than it. From the response of the Sanitation Department, the program helps them save much fuel fee. For us, the next step is to extend this technology to big company, such as logistics corporations, post office, sanitation department and so on, to obtain more economic benefit.

ACKNOWLEDGMENT

The authors wish to thank the Chongqing Software Engineering Key laboratory who provides the computers. Furthermore, thank the sprinkler car drivers of the sanitation department in Chongqing Yubei district who collect the data and have the test.

REFERENCES

[1] B.L. Golden , R.T. Wong , "Capacitated arc routing problems,"Networks, vol.11 , pp. 305–315, 1981.

[2] Patrick Beullens, Luc Muyldermans, Dirk Cattrysse, Dirk Van Oudheusden , "A guided local search heuristic for the capacitated arc routing problem," European Journal of Operational Research, vol 147, pp. 629–643, 2003.

[3] Belenguer, J.M. and E. Benavent , "A Cutting Plane Algorithm for the Capacitated Arc Routing Problem," Computers and Operations Research, vol.30, pp. 705–728, 2003.

[4] DeArmon, J.S. , "A Comparison of Heuristics for the Capacitated Chinese Postman Problem," Master's Thesis, The University of Maryland at College Park, MD, USA. , 1981.

[5] Eglese, R.W , "Routing Winter Gritting Vehicles," Discrete Applied Mathematics, vol.48(3), pp. 231–244 , 1994.

[6] Peter Greistorfer , "A Tabu Scatter Search Metaheuristic for the Arc Routing Problem," Computers & Industrial Engineering, vol. 44, pp. 249–266, 2003.

[7] Philippe Lacomme, Christlan Prins and Wahiba Ramdane-Cherif, "Competitive Memetic Algorithms for Arc Routing Problems," Annals of Operations Research, vol.131, pp. 159–185, 2003.

[8] Oliver, I.M., D.J. Smith, and J.R.C. Holland , "A Study of Permutation Crossover Operators on the Traveling Salesman Problem," In J.J. Grefenstette (ed.), Proceedings of the 2nd Int. Conf. on Genetic Algorithms. Hillsdale, NJ: Lawrence Erlbaum, pp. 224–230,1987.

[9] Holland, "Adaptation in Natural and Artificial Systems," University of Michigan Press , 1987.

[10] Moscato , "Memetic Algorithms: A Short Introduction," In D. Corne, M. Dorigo, and F. Glover (eds.), New Ideas in Optimization. McGraw-Hill, P, pp. 219–234, 1999.

[11] Barrie M. Baker, M.A. Ayechew , "A genetic algorithm for the vehicle routing problem," Computer&Operation Research, vol.30, 787–800, 2003.