Multi-chromosome Genetic Algorithm for Multi-object Planning Problem

ShouKang Cui
517030910371
csk151183@sjtu.edu.cn

PengXiang Du
517030910372
dpx96956@sjtu.edu.cn

RuoFeng Liu
517030910377
liuruofeng@sjtu.edu.cn

***Abstract*—In this project, we design a new genetic algorithm to solve multi-depot and multi-type vehicle routing problem(MDMTVRP) which is a representative problem in multi-object planning. We design multiple chromosome method to encode the problem, and come up with new selection, crossover and mutation operation to accommodate multi-chromosome method. We do experiment on small data set and larger data set, and finally get acceptable results.**

I. Introduction

Multi-object planning problem is very practical in our daily life.Some industries need proper planning to maximize profit and minimize cost such as take-out,express delivery and public traffic.To research multi-object planning problem, We decide to choose multi-type vehicle routing problem(MDMTVRP) as a example.We know that MDMTVRP is a variation of VRP, and has more constraints and more objects that taken into consideration than VRP.So MDMTVRP is clearly a NP hard problem.Heuristics are usually used in solving this kind of probelms, genetic algorithm is one of them.

We know that traditional genetic algorithm can be effective in solving traditional VRP. But in our MDMTVRP, conditions vary.Starting points are not only one. And vehicle's types become various. Traditional one chromosome genetic may run into dilemma in MDMTVRP, because it is difficult to express abundant information on one chromosome.So in our experiment,we think about develop a more effective method to solve it.At the beginning, we think about other methods to solve it. We find that other methods may get optimal in each vehicle's route, but the global solution may not be the optimal. So we take improvement of genetic algorithm into consideration since it tends to get global optimal.

II. Idea motivation

Before we solve MDMTVRP,we do some research about other existing methods to deal with this problem. But we find the widely used method is mutation ant colony algorithm. It is used for shortest finish time. And other methods apply simple one chromosome traditional genetic algorithm, which are not very effective. So we think about if one chromosome limits the expression of information,why not adopt multi-chromosome to represent multi-object?This method is clear and easy to understand.

III. MDMTVRP

A. Problem description

There are multiple depots, each depot has some numbers of vehicles. And vehicles have multiple types. Now depots need to provide distribution service for customers to satisfy their need. How to plan routing to minimize transportation cost.

And there are some concrete details:Multiple depots can have multiple types of vehicles.Different types of vehicles may have different load capacity .Different types of vehicles may have different transportation costs(including fixed cost and variable cost).One vehicle can be arranged at most one time, starting from one depot and finally going back to the same depot.One customers need is less than one vehicles load.One customer can be only served by one vehicle.

B. Formulation

There are total H vehicles in different depots and N customers. Vehicles and customers have their own location coordinates. Vehicle's load capacity is $Q_h(h = 1, 2, ..., H)$. The fixed cost of vehicle is c_{1h}, the variable cost about distance is c_{2h}. The need of each customer is $g_n(n = 1, 2, ..., N)$.

We use a graph $G(V, E)$ to solve this problem. $V = \{1, 2, ..., N + H\}$,first N nodes are customers, the latter H nodes represent vehicles. The distance set $E\{d_{ij}|i, j \in V\}$,d_{ij} is the distance from node i to j. We design a decision variable x_{ijh},when vehicle h goes from i to j, it is set to 1. Otherwise it is 0.

Under a series of constraints, our goal is to calculate:

$$minz = \sum_{h=1}^{H} \sum_{j=1}^{N+H} \sum_{i=N+1}^{N+H} c_{1h} x_{ijh} + \sum_{h=1}^{H} \sum_{j=1}^{N+H} \sum_{i=1}^{N+H} c_{2h} d_{ij} x_{ijh}$$

IV. Traditional GA

Genetic Algorithm: a computational model of biological evolution process that simulates natural selection and genetic mechanism of Darwin's biological evolution theory. It is a method to search the optimal solution by simulating natural evolution process.

Genetic algorithm starts with a population representing the possible solution set of the problem, and a population consists of a certain number of individuals encoded by genes. Each individual is actually a chromosome entity with characteristics.

Chromosome is the main carrier of genetic material, i.e. the collection of multiple genes. Its internal expression (i.e. genotype) is a certain combination of genes, which determines the external expression of an individual's shape. At the beginning, mapping from phenotype to genotype, i.e. coding, needs to be realized. Because the work of imitating gene coding is very complicated, we often simplify it, such as binary coding.

After the generation of the first generation population, according to the principle of survival of the fittest, generation by generation evolution produces better and better approximate solutions. In each generation, individuals are selected according to the fitness of individuals in the problem domain, and the population representing the new solution set is generated through combination crossover and mutation by means of genetic operators of natural genetics.

The whole process of genetic algorithm is shown in fig. 1. We can know that it has 6 key components:

- **Initialization.** Set the maximum evolution algebra T, and randomly generate M individuals as initial population P(0).
- **Individual evaluation.** Calculating the fitness of each individual in the population P(t).
- **Selection.** The selection operator is applied to the population. The purpose of selection is to pass the optimized individuals directly to the next generation or to generate new individuals through pairing and crossing and then pass them on to the next generation. The selection operation is based on the fitness evaluation of individuals in the group.
- **Crossover.** The crossover operator is applied to the population. Crossover operator plays a key role in genetic algorithm.
- **Mutation.** The mutation operator is applied to the population. That is, the gene values on some loci of individual strings in a group are changed. Population P(t) is selected, crossed and mutated to obtain the next generation population P(t+1).
- **Termination condition judgment.** If t=T, the individual with maximum fitness obtained in the evolution process is taken as the optimal solution output, and the calculation is terminated.

Usually,The traditional genetic algorithm (GA) adopts the single-chromosome coding method. But if we use this classical way in solving multi-object and multi-type planning problems, such as multiple traveling salesman problem(mTSP) and multi-depot and multi-type vehicle routing problem(MDMTVRP), It will have many disadvantages:

- **Complexity.** Due to the planning of multiple objects, it is difficult and complex to code with only one chromosome at this time.
- **Low quality.** Since the problems usually have multiple constraints,the solution quality will not be high.
- **Infeasible solution.** Due to the multiple constraints, a large number of infeasible solutions or even illegal solutions will be generated.

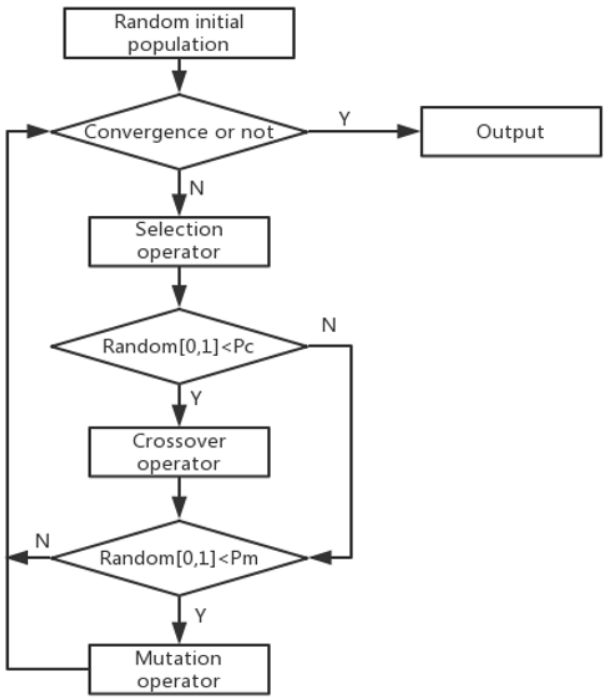

Fig. 1. Flow chart of genetic algorithm

- **Large searching space.** If we carry out various operations on only one chromosome, it will often has a large searching space and low searching efficiency.

But if we use a chromosome to represent each object or type, and the number of chromosomes is equal to the number of objects, a new multi-chromosome genetic algorithm is formed, which is what we want to say today. In the multi object planning problem, MDMTVRP is a very complex representative problem. Therefore,we will take this problem as the main case to illustrate our algorithm.

V. Multiple-Chromosome Genetic Algorithm

A. Design multi-chromosome

We firstly design the encoding method for the problem. In traditional genetic algorithm, one chromosome represents one solution to the problem. But in our problem, we have different type of vehicles to provide service. If we use one chromosome to present all answers, it may increase complexity because we mix up different type of vehicles and all customers. Thus we use multi-chromosome to represents one answer to the problem.

Firstly, we create two basic classes. One class is vehicle class, which contains the information of one vehicle such as vehicle id, vehicle type, vehicle capacity, vehicles depot position and vehicles cost. Another class is customer class, which contains the information of one customer such as customer id, customer position and customers need.

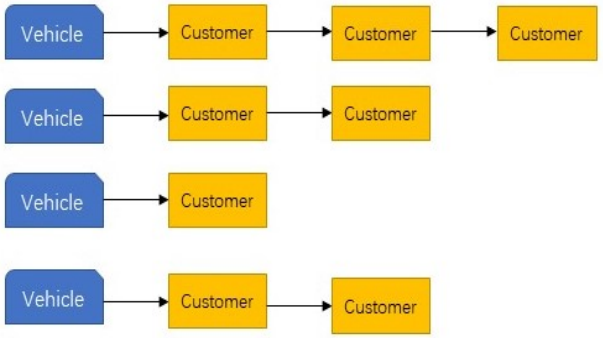

Fig. 2. Multi-chromosome

After we create these classes, we create a list, which contains one vehicle and the customers served by this vehicle. In our project, this list represents one chromosome.

The first gene on the chromosome is vehicle gene, and the latter genes are customer genes. One chromosome is only the scheme of one vehicle. But we have many different types of vehicles.

To get a complete solution, we need multiple chromosomes. The number of chromosomes is equal to the number of total vehicles. Thus we finally create a multi-chromosome class. This class contains a list of chromosomes. The length of this list is equal to the number of chromosomes. So one multi-chromosome class represents one solution to the question.

B. Randomly initialize the population

In the traditional genetic algorithm, because there is only one chromosome and all customers are on the same chromosome, population initialization is very simple and intuitive. But we use a chromosome to represent each vehicle, and the number of chromosomes is equal to the number of vehicles, create a new multi-chromosome genetic algorithm. Accordingly, we need to create a population initialization method suitable for multi-chromosome genetic algorithm.

An initial population with a population size of S is randomly generated, and the specific process is as follows.

Randomly select a customer who has not been assigned, then randomly select a vehicle (i.e. a chromosome), and try to assign this customer to this vehicle. If it is not overloaded, this customer will be assigned to this vehicle, otherwise, randomly select one of the other vehicles and judge whether it is overloaded again. By analogy, until the customer is assigned to a certain vehicle, the remaining customers are assigned to the vehicles (chromosomes) in turn according to the above procedure. The order in which the customers are assigned to the vehicles is the order in which they are served. This procedure is equivalent to randomly creating an initialization individual.

Repeat the above process for S times, then we will get an initial population with size S.

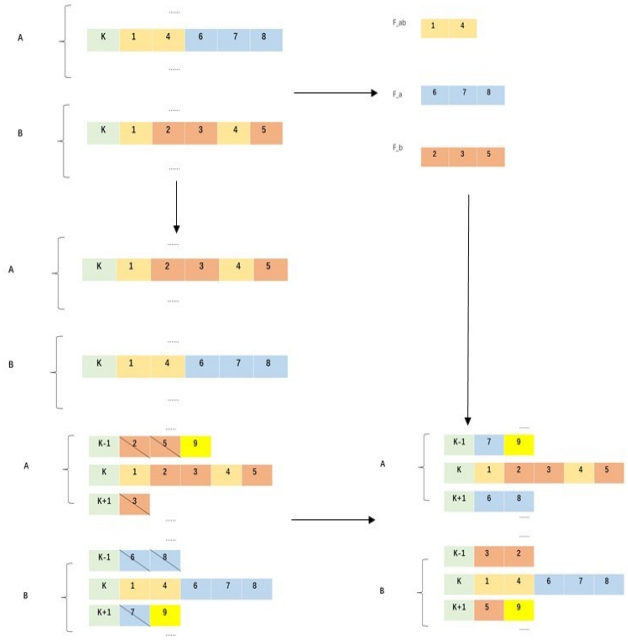

Fig. 3. crossover operation

C. Crossover

There are six usual crossover operations in traditional genetic algorithm: partial-mapped crossover, order crossover, position-based crossover, order-based crossover, cycle crossover and subtour exchange crossover. These methods are widely applied in one chromosome genetic algorithm. But in our multi-chromosome genetic algorithm, we need do some changes.

Since crossover operation in traditional genetic algorithm is not suitable for our multi-chromosome method, so we need to come up with a new one. We know that unfixed point crossover and multiple point crossover may lead to the overload of vehicle, resulting the generation of invalid solution. We design a multi-chromosome fixed one point crossover, here are the concrete steps:

1) : in one problem, we have H vehicles. We randomly choose one number ranging from 1 to H. This number is the number of chromosome we will operate. We mark this number k.

2) : We compare the k_{th} chromosome of individual A and individual B, and we create three gene pools, one gene pool F_{ab} stores common genes of k_{th} chromosome of A and B, one gene pool F_a stores unique genes of A, one gene pool F_b stores unique genes of B.

3) : We swap all customer genes on k_{th} chromosome of A and B.

4) : Compare all genes on other chromosomes of A except k_{th} chromosome with genes in gene pool F_b. If there exists

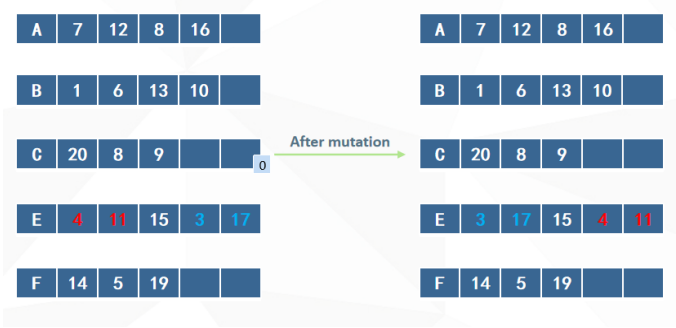

Fig. 4. **Flip** mutation operator

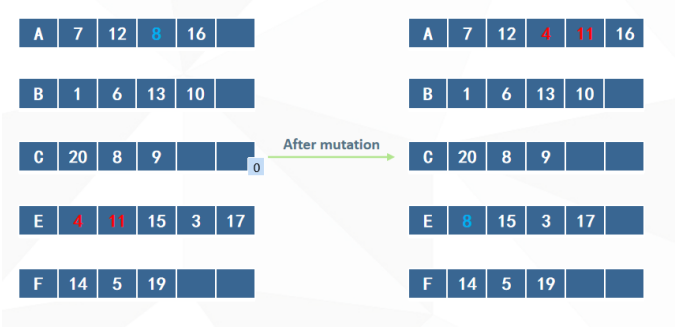

Fig. 5. **Swap** mutation operator

same genes, delete same genes on As chromosomes. Similarly, we do it for B.

5) : Finally, we randomly insert genes in F_a into individual As random chromosomes random position after first vehicle gene. Then, we check whether this operation will lead to overloading. If overloading, do this operation again randomly. Finally we distribute all genes in gene pool F_a. We do similar operation on individual B.

D. Mutation

Since we use a new coding method, we must design new genetic evolution operator operators. Most of the current operators are derived from other operators, such as constructing multiple chromosomal mutation from a single chromosomal mutation sequence. The operators used in this paper are complex operators designed by combining simple and direct operators, which speeds up the evolution process and improves the efficiency of genetic algorithm.

In the following, we will introduce three mutation operators, of which the first two are multi-chromosomal mutation operators derived from single chromosomal mutation method, but both operators have disadvantages. Finally, we combined these two operators to create a new mutation operator, which is more suitable for our multi-chromosome genetic algorithm.

1) Flip operator: As shown in fig. 2, any two gene fragments in a single chromosome are exchanged with each other, and mutation operators only operate in one chromosome, **but only** two gene sequence fragments in the same chromosome are exchanged in this way.

2) Swap operator: As shown in fig. 3, any two chromosomes exchange two randomly selected gene fragment sequences, which forms a simple "Swap" mutation mode. If one of the gene sequences is empty, the operator will be implemented to insert the non-empty sequence into a randomly selected position in another chromosome.

However, this kind of operator may destroy the constraints and produce illegal solutions.

3) New combined gene-based operator: In order to avoid the shortcomings of the above two operators, we combined them to create a new complex gene-based operator.

For all genes on all chromosomes, we generate a random number to determine whether this gene is mutated. If a

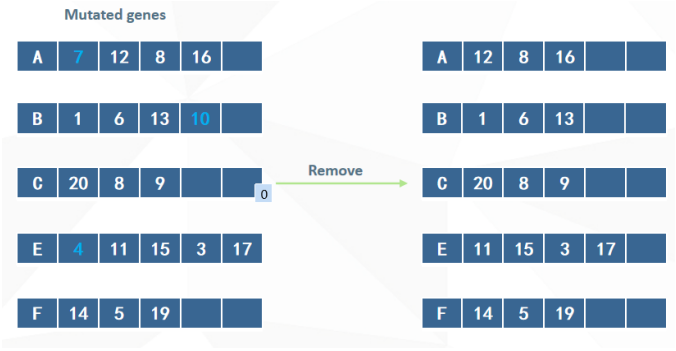

Fig. 6. The process of forming the mutant gene pool in the **new combined** gene-based operator

mutation occurs, we remove the gene from its chromosome and add it to the pool of mutant genes. As shown in fig. 4, its mutation gene pool is [7,10,4].

Then the genes in the mutant gene pool are randomly inserted into any position after the first gene of any chromosome of the individual one by one. Then we judge whether the vehicle is overloaded: if overloaded, we randomly insert the next chromosome and judge whether it is overloaded again, and so on, until all the genes in the mutant gene pool are distributed out. Thus we complete a mutation operation.

This new operator **not only** operates on every chromosome, **but also** does not destroy the constraints of the problem! Therefore, we use this new operator which we create as the mutation operator in our algorithm.

E. Selection

There are many dfferent methods for selection.For example, proportion selection(according to the proportion of individual fitness in total population), sort selection(sort the individuals by their fitness and select the superior individual first), etc. Here we use a new method with the **combination** of these two methods.

First the fitness of the individuals will be transformed by the following formula,

$$f = \frac{f_{max} - f_i + \gamma}{f_{max} - f_{min} + \gamma}$$

f_{max} means the max fitness of all individuals, f_{min} means the min fitness of all individuals. γ is a positive number which is used to prevent equation from zero division in the processes

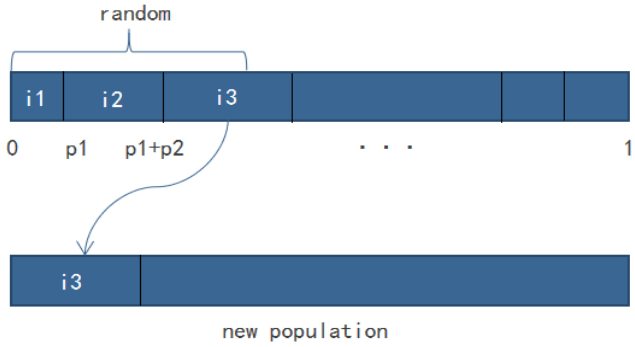

Fig. 7. The process of **select** one individual to next generation

of computation. In this project we use the fixed cost of the vehicles as γ.

After the transformation, the indiciduals with lower cost have higher fitness. **What's more**, the individuals with the best fitness will not take up to much proportion, and in the earlier iterations this will prevent from converging too fast and falling into one local optimum.

Then we use proportion selection to select individuals for next generation. According to the proportion of the fitness in total fitness, the individuals are allocated into a interval from 0 to 1. In each time, use a random number to define which one to select as shown in fig.5.After N times of repeating, we will get a new population of size N.

However, because of the mutation or other uncontrollable factors, the best individuals might disappear in the next generation. This is disadvantage to the iteration. Thus we combine the method with the idea of **sort selection**. We will select the **best** individual with the bset fitness before the operations. And after preliminary selection, mutation and cross over, check whether this individual is still in the next generation. If not, we will replace the worst one in the population with the previous best one. By this, the superior individual will be better and better and the fitness will monotonically increasing.

VI. EXPERIMENTS

A. Example verification

In order to verify the feasibility and effectiveness of the Multi-chromosome Genetic Algorithm, we solve a easy example. In this example, we use the straight-line distance, which is more convenient to explain the problem. The location of the depots are manually set far away from each other.The location of the customers is randomly distributed in the area of $100km \times 100km$. There are three depots and three types of vehicles (4 in total) with 10 passengers, refer to table I and table II for the information of customers and depots. It is required to arrange appropriate vehicles and their driving routes to minimize the total cost.

The parameters of Multi-chromosome Genetic Algorithm are set as follows: scale of population is 100, number of generations is 1000, probability for mutation is 0.1, probability for cross-over is 0.5. The result is shown in table III, and the final cost is 3342. The vehicle route is shown in fig.8. The fitness variation curve is shown in fig.9.

TABLE I
CUSTOMER INFORMATION

ID	1	2	3	4	5
Need(kg)	24	22	24	25	22
Location(x_km)	70	21	99	27	20
Location(y_km)	20	38	55	39	20

ID	6	7	8	9	10
Need(kg)	24	37	37	40	33
Location(x_km)	65	87	15	22	90
Location(y_km)	60	59	70	90	35

TABLE II
DEPOT/VEHICLE INFORMATION

Vehicle ID	1	2	3	4
depot ID	1	2	3	2
type	1	2	2	3
category(kg)	90	100	100	110
fixed cost	10	12	12	13
variable cost	6	7	8	9
Location(x_km)	30	85	30	85
Location(y_km)	30	55	80	88

As you can see, the routes of different vehicles don't cross and vehicles service for customers who is close to its depot. This means that our solution is pretty good. $fig.9$ shows that in the early iterations, the cost is falling fast and the optimal solution is get at $361th$ generation.

However, because of the randomness of the operations, we might not always find the best solution and we just try to find the better solution as we can.

VII. CONCLUSION

In this project, we design a Multi-chromosome Genetic Algorithm to solve multi-depot and multi-type vehicle routing problem(MDMTVRP) which is a representative problem in multi-object planning. We use a list to represent a chromosome and the amount of chromosomes is equal to the amount of vehicle. Due to the change of individual expression from traditional genetic algorithm, we need to come up with new cross-over and mutation operation to accommodate multi-chromosome method. This is the same idea of traditional genetic algorithm but the implementation is actually different. As for the selection, we use proportion selection combined with optimal selection. The results show that the multi-chromosome genetic algorithm designed in this paper has high quality and stability but speed can be further improved.

TABLE III
VEHICLE SCHEDULING SCHEME

vehicle ID	Route
1	A-5-2-4-A
2	B-1-10-3-B
3	B-6-7-B
4	C-8-9-C

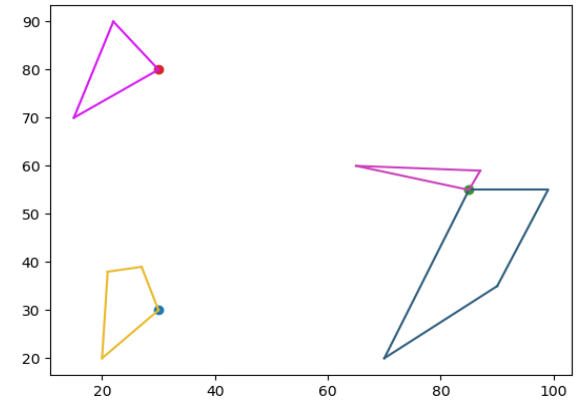

Fig. 8. Schematic diagram of vehicle path. The spots represent three depots.

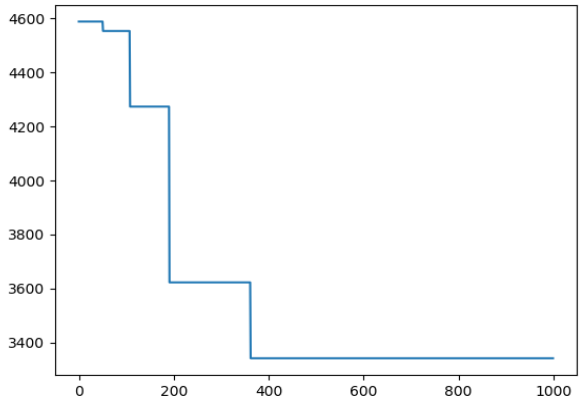

Fig. 9. Evolution curve

In further research, we might use a linked list to represent a chromosome. This will be helpful for the speed. And adding constraints of time will be more closer to reality.

REFERENCES

[1] CHENG R,GEN MEvolution Program for Resource Constrained Project Scheduling Problem [J]Evolutionary Computation,1994,11(3): 274-287.