Optimization for Coal Heavy Haul Transportation Assembly Scheme Problem Using Genetic Algorithm

Xuesong Han
School of Transportation and Logistics
Southwest Jiaotong University
Chengdu, China
e-mail: hxs2004@126.com

Zhenghong Gu
School of Mines
China University of Mining and Technology
Xuzhou, China
e-mail: kdgzh62@126.com

***Abstract*—This paper presents a Coal Heavy Haul Transportation Assembly Scheme Problem (CHASP), in which the time consuming functions, assembly number constraints and assembly weight constraints etc are considered. The time consuming costs consist of residence time and disassembly time. The disassembly time functions are usually nonlinear functions of unit train departure directions. Then, a nonlinear 0-1 programming is formulated for the problem and solved by lingo mathematical solver. Considering the complexity of the problem, a kind of Genetic Algorithm is proposed to solve it. Extensive computational experiments are taken on randomly generated data, the detailed results are given and the genetic algorithm is shown to be efficient.**

Keywords- Coal heavy haul transportation; Assembly scheme; Genetic algorithm;

I. INTRODUCTION

While the freight railroad industry has been in existence for over two centuries, the fundamental concept of aggregating freight railcars based on different attributes to form outbound trains has not changed. For some coal mine areas in China, coal is loaded into freight railcars and then transport to the destinations. During this process, some inbound trains need to be assembled together to become outbound train with heavier weight in order to transport more coal at the same time. The place where all the inbound trains assembled together is called classification yards [1].

Generally speaking, the modes of assembly operation are as follows: inbound trains coming from coal mine loading areas with different weights and different destinations get to the classification yards. There are some operation choices for them at this time, they may stop on the empty tracks and departure when the departure time is due, or waiting on the track to be assembled with other inbound trains coming later. they can also find some proper trains waiting on the tracks and assemble with them directly. If unit trains are assembled to outbound trains, the train weight and number should be less than their upper bounds. All the outbound trains must departure on time without any delay. And if unit trains are assembled together with different destinations, outbound train should be separated when arriving at some point near the destinations to make sure that different unit trains can go to their own destinations and some disassembly time consuming is needed.

During this operation process of unit trains grouped together to form outbound trains, how to reduce the total consuming time is one of the key problems in coal transportation planning. Coal can be transported to the destinations more quickly if decreasing consuming time. Therefore, our goal is to determine the assembly scheme of inbound trains and pursuit that all the inbound trains can get to their destinations as quickly as possible. The assembly scheme should determine whether the inbound trains should stop on the empty tracks waiting to departure alone or assemble with other trains and which trains should be assembled, then when to departure and so on.

Although many papers paid attention to railway optimization[2]. Literatures on assembly are scarce, literature [3] considers the real time dispatch problems of trams in storage yards. Literature [4] studies the marshalling problem which rearranges the train carriages in a station to group them together by destination. The goal is to rearrange the carriages using minimum tracks. In this paper, we will consider a coal heavy haul transportation assembly scheme problem.

II. PROBLEM DESCRIPTION

Suppose that during a period of time, a set of inbound unit trains, $I = \{1 \cdots \mathrm{i} \cdots \mathrm{m}\}$ with different attributes get to the classification yards at arriving time set, $\{td_1, td_2 \cdots td_m\}$ and a set of departure trains, $J = \{1 \cdots j \cdots n\}$ will departure from the station at departure time set $\{tf_1, tf_2 \cdots tf_n\}$. There are three kinds of weights for unit train set I, the weight set is $\mathrm{W} = \{5000,10000,20000\}$, and the total number of 5000 tons trains is m_1, the total number of 10000 tons trains is m_2, the total number of 20000 tons trains is m_3. So we have $m = m_1 + m_2 + m_3$. Each unit train has their own destination and train weight. The objective is to find the assembly scheme to minimum unit train consuming time.

III. MODELING

A. Model Assumptions

The problem is formulated based the following assumptions:

- The parameters like inbound train number, arriving time, train weight etc are known.

- The departure time, departure train number etc are known.
- Locomotives number and capacity are enough for all the departure trains.
- Tracks capacity is enough to operate the assembly process.

B. Symbol Description

In order to describe the problem and model, the symbols of description are as follows.

nb: minimum number of unit trains that a outbound train can consist.

nu: maximum number of unit trains that a outbound train can contain.

wl: minimum weight of outbound train.

wu: maximum weight of outbound train.

t_{df} : standard operation time for unit train to arrive and departure.

t_{fen} : standard disassembly time to separate assembly train into unit trains.

t_{zu} : standard assembly time for separating unit trains to assembly train .

Q_i: weight of unit train i.

The decision variables are as follows:

$$x_{ij} = \begin{cases} 1 \text{ if train i is assembled to outbond train j} \\ 0 \qquad \text{otherwise} \end{cases}$$

$$b_{ii'} = \begin{cases} 1 \text{ if train i has the same destination with train i}' \\ 0 \qquad \text{otherwise} \end{cases}$$

C. Model Formulation

CHASP with consuming time can be formulated as:

$$\min Z = \sum_{i=1}^{m}\sum_{j=1}^{n}(tf_j - td_i)x_{ij} + \sum_{i=1}^{m-1}\sum_{i'=1}^{m}\sum_{j=1}^{n} x_{ij}\cdot x_{i'j}\cdot b_{ii'}\cdot t_{\text{fen}} \quad (1)$$

$$\sum_{j=1}^{n} x_{ij} = 1 \ \forall i \in I \quad (2)$$

$$nb \le \sum_{i=1}^{m} x_{ij} \le nu \ \ \forall j \in J \quad (3)$$

$$wl \le \sum_{i=1}^{m} x_{ij}Q_i \le wu \quad \forall j \in J \quad (4)$$

$$x_{ij}\cdot x_{i'j}\cdot(td_{i'} + t_{df} + t_{zu}) \le tf_j \ \ \forall i, i' \in I, j \in J \quad (5)$$

$$x_{ij}(td_i + t_{df}) \le tf_j \quad \forall i \in I, \forall j \in J \quad (6)$$

The objective(1) minimizes the total consuming time of all the unit trains, it include two parts, the former one is the sum of consuming time that all unit trains spend at classification yards, the later one is the sum of disassembly time that all outbound trains with different destinations are disassembled into unit trains. Constraint (2) ensures that each inbound train can only be assembled into an assembly train. Constraint (3) guarantees that each outbound train can only be assembled by a certain number of unit trains within the range [*nb,nu*]. Constraint (4) states that each outbound train's weight should be within the range [wl,wb]. Constraint (5) express that if some unit trains assemble together, all of them should finish all the operation before the departure time. Constraint(6) ensures that if some unit trains departure alone, they should also finish all the operation before their departure time.

IV. Solution Based on Genetic Algorithm

CHASP is a nonlinear 0-1 programming model, it's easy to get its best solution within reasonable time when the problem scale is not very large. With the problem scale increasing, solution time increases quickly and sometimes the time consuming is unacceptable. In order to get the feasible solution within reasonable time, some heuristic solution methods should be developed for this problem. In this paper, we adopt a kind of genetic algorithm to solve it. Solution time and solution quality will be compared with results of mathematical solver.

Genetic Algorithms are an efficient optimization technique for many problems, it was first proposed by John Holland in his paper "Adaption in natural and artificial systems[5]".

A. Chromosome Encoding Method

Chromosome encoding method has a great influence on the speed of solving the problem. In this paper, we adopt matrix encoding method and chromosome length is equal to the number of matrix elements (as shown in Figure 1).

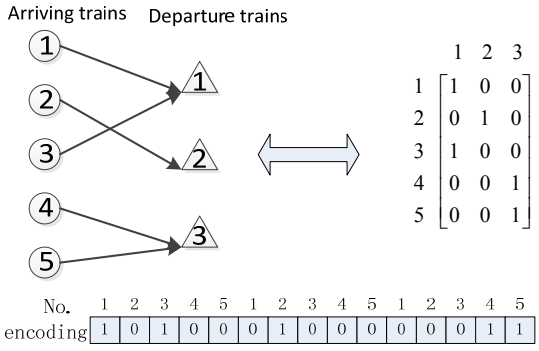

Figure 1: Chromosome encoding method

The columns of matrix represent inbound trains and rows of matrix represent outbound trains. If arriving unit train i will be assembled to departure train j, the element in the matrix will be 1, otherwise the element will be 0. Then connect all the columns together to get the chromosome, the number of genes in the chromosome is equal to the number of element in the matrix.

B. Fitness

The probability of survival of any individual is determined by its fitness. The objective of this problem is to minimize the total consuming time. Therefore, the objective value can be viewed as fitness directly.

$$Fitness = Z \quad (7)$$

C. Genetic operators

In the genetic algorithm, there are three main genetic operators, selection, crossover and mutation. By using different probabilities for applying these operators, the speed of convergence can be controlled.

- Selection Operation
  In this paper, The GA in this study uses an elitist selection based on the roulette wheel spin method [6] and elitism strategy [7-8].
- Crossover Operation
  The crossover operator mixes parts of two parent solutions to create two different offspring solutions. In this paper crossover is carried out by the one-point cutting method [6]. One-point cutting method is the simplest crossover method but sometimes is very efficient. The operation process is like this: a cutting point is chosen randomly and the "offsprings" are obtained by taking the first part from one parent and the second from the other. And the crossover probability is 0.6 in this paper.

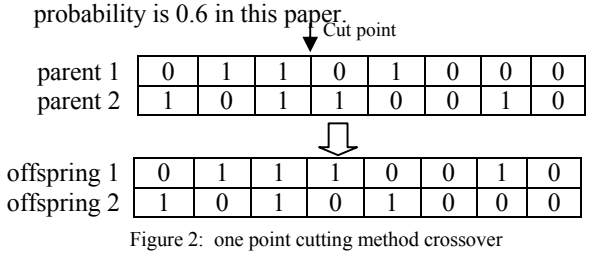

Figure 2: one point cutting method crossover

Through the above crossover operation, new chromosome will be created.

- Mutation Operation
  Similar to crossover operation, mutation operation is carried out to prevent the premature convergence and explores new solution space. A new individual is created by making modifications to one selected individual, which is different from crossover. An integer mutation position along the chromosome is selected uniformly at random, mutated from 1 to 0 or from 0 to 1 with a mutation probability. This kind of mutation is called swap mutation.

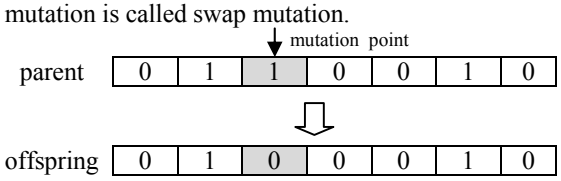

Figure 3: swap mutation method

- Constraint Dealing
  In the model of CHASP, there are five constraints, it's not easy for the chromosomes to satisfy all the constraints. Here, we take some measures to limit the initial solution and create penalty functions to punish it if some constraints are not satisfied.

a) Initial Chromosome

Step 1: When the initial chromosome is created, The sum of every row must be equal to 1. So constraints (2) can be satisfied.

Step 2: we also let the sum of every column in the matrix be less than or equal to the upper number nu and greater than or equal to the lower bound nb. Then, constraints (3) are satisfied.

Step 3: For each column, plus all the weight that corresponding gene is equal to 1, then the total weight is acquired. We make sure that the total weight of all unit trains must be less than or equal to the upper number wu and greater than or equal to the lower bound wl. Then, constraints (4) are satisfied.

b) Algorithm Flow Chart

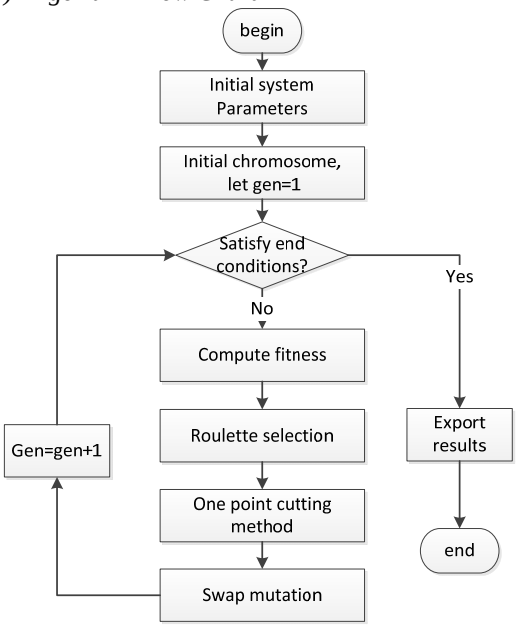

Figure 4: Algorithm flow chart

c) penalty functions

Penalty functions are developed to influence the fitness.

$$Z_1 = g_1 \cdot [x_{ij} \cdot x_{i'j} \cdot (td_{i'} + t_{df} + t_{zu}) - tf_j] \quad (8)$$

$$g_1 = \begin{cases} 1 \ if \ x_{ij} \cdot x_{i'j} \cdot (td_{i'} + t_{df} + t_{zu}) - tf_j \geq 0 \\ 0 \qquad otherwise \end{cases} \quad (9)$$

$$Z_2 = g_2 \cdot [x_{ij}(td_i + t_{df}) - tf_j] \quad (10)$$

$$g_2 = \begin{cases} 1 \ if \ x_{ij}(td_i + t_{df}) - tf_j \geq 0 \\ 0 \qquad otherwise \end{cases} \quad (11)$$

then fitness can be expressed as:

$$Fitness = Z + Z_1 + Z_2 \quad (12)$$

d) End Conditions

Here, we design two end conditions for this genetic algorithm. The first one is that the algorithm will stop if generation is greater than the maximum number. Another one is that, if the solution will not change for some time, we think the solution get its

acceptable level, algorithm will also terminate in order to save some computation time.

V. COMPUTATIONAL EXPERIENCE

The Genetic Algorithm implementations discussed in the previous section is used to solve some instances and programmed by matlab 7.0. These instances are generated randomly and also solved by mathematical solver lingo, and the solutions are compared with those got from genetic algorithm in this paper(as table I shows).

TABLE I. COMPUTATION INSTANCES

No.	I(m)	m_1	m_2	m_3	J(n)	LC(s)	GC(s)	ΔC
1	6	2	3	1	4	18	1020	0%
2	8	3	4	1	6	33	1460	2.1%
3	10	4	5	1	7	46	1867	3.2%
4	12	4	6	2	9	190	2183	3.6%
5	14	4	8	2	10	1864	2472	3.8%
6	16	5	9	2	12	2390	2620	4.5%
7	18	6	9	3	13	>14400	3145	5.3%
8	20	7	10	3	15	+∞	4031	—
9	24	9	11	4	16	+∞	4765	—
10	28	10	13	5	20	+∞	5640	—
11	32	12	14	6	23	+∞	6986	—
12	35	13	15	7	25	+∞	7683	—
13	40	15	16	9	29	+∞	9846	—

Note:
LC : computation time that using lingo solver to solve the instances .
GC: computation time that using Genetic Algorithm to solve the instances.
ΔC: the solution gap between the Genetic Algorithm and lingo solver.

TABLE I gives the computation results for genetic algorithm and lingo solver. We can see that computation time increases quickly with the increment scale of the instances (as shown in Figure 5).

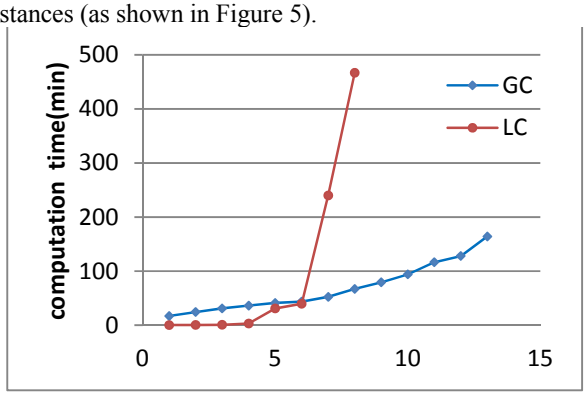

Figure 5: computation time compared chart

When the scale is larger than 18 unit trains, it's unacceptable to solve the problem using mathematical solver lingo. Much more time is needed, and many unit trains have come before computation is finished.

Compared with the lingo solver, Genetic Algorithm proposed in this paper has a better performance for solving CHASP. When the scale is over 18 unit trains, Genetic Algorithm can also get feasible solutions within acceptable computation time.

VI. CONCLUSION

In this paper, we proposed a new problem call CHASP, which is got from Chinese coal heavy haul transportation. And a nonlinear 0-1 programming model is formulated. For the small scale instances, it can be solved by mathematical solver and for larger scale instance, one kind of genetic algorithm is proposed to solve it. Solution quality and computation time are also compared through different random instances.

From the above analysis, we can get the following conclusions:

Genetic algorithm is useful for solving CHASP, it performs better than mathematical solver when applying to larger scale instances.

On the other hand, from the solution gap analysis, we can see that the gap is over 5% sometimes. Therefore, other more effective algorithm should be studied in the future to improve the solution quality.

REFERENCES

[1] Yue Y, Zhou L, Yue Q. Multi-route railroad blocking problem by improved model and ant colony algorithm in real world[J]. Computers & Industrial Engineering, 2011, 60(1): 34-42.

[2] Caprara A (2010). Almost 20 years of combinatorial optimization for railway planning: from lagrangian relaxation to column generation. Paper presented at: Proceedings of the 10th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems.

[3] Winter T, Zimmermann U T. Real-time dispatch of trams in storage yards[J]. Annals of Operations Research, 2000, 96(1): 287-315.

[4] Dahlhaus E, Horak P, Miller M. The train marshalling problem[J]. Discrete Applied Mathematics, 2000, 103(1): 41-54.

[5] Holland J H. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence[M]. MIT press, 1992.

[6] D.F. Goldberg, Genetic algorithms in search optimization and machine learning. Addison Wesley, Reading, Massachusetts (1989).

[7] A. Yamamoto, A quantitative comparison of loading pattern optimization methods for in-core fuel management of PWR, J. Nucl. Sci. Technol. 34 (1997), pp. 339–347.

[8] Deb K, Pratap A, Agarwal S. A fast and elitist multiobjective genetic algorithm: NSGA-II[J]. Evolutionary Computation, IEEE Transactions on, 2002, 6(2): 182-197.