Depot shunting scheduling using combined genetic algorithm and PERT

N. Tomii & Li J. Zhou

Railway Technical Research Institute JAPAN

Abstract

Shunting scheduling problems at railway depot can be regarded as a sort of resource constrained project scheduling problem (RCPSP). Unlike the normal RCPSPs, however, shunting scheduling problems require that the number of activities which form a project has to be dynamically changed in the problem solving process and some of the activities have to be performed at a prescribed timing, for example. We propose an efficient algorithm for depot shunting scheduling problems combining genetic algorithm (GA) and PERT. GA and PERT are combined so that the candidates for answers in GA are evaluated by PERT. This enables us to reduce the search space of GA to a great extent. We have confirmed the effectiveness of our algorithm through several experiments using practical train schedule data.

1 Introduction

Decision support systems such as train scheduling, vehicle scheduling and crew scheduling have been widely used in the area of railway transportation planning. Nowadays, it is strongly desired to make such systems more advanced to improve the process of transportation scheduling.

To this end, we focus on the following two points. One is to develop algorithms with higher functions such as to make transportation schedules automatically. Although little attention seems to have been paid to automatic scheduling at the moment, it is helpful to improve the efficiency of transportation scheduling.

The other is to develop algorithms that are applicable in various kinds of circumstances irrespective of the differences caused by the specific conditions of
the site of application. That is, if we develop algorithms for each site from scratch to reflect the specific conditions, it entails prohibitively high costs.

In this paper, we introduce an algorithm that makes depot shunting schedules automatically. One of the characteristics of this algorithm is that it is not dependent upon the specific conditions of depot such as the track layout and other factors. Hence, we can use the algorithm flexibly for different types of depots without giving a significant change.

A depot shunting schedule prescribes a series of tasks for one day at a railway depot. As for the tasks, there exist several types of inspections, maintenance, cleaning of rolling stock together with the shunting of rolling stock. A workforce assignment schedule is also included.

A basic depot shunting schedule is made when the train schedule is revised. Then, it is modified daily based upon the change of the everyday train schedule or delay caused by accidents etc.

Depot shunting schedules are currently made totally by hand by experts. It takes a couple of hours even for them to make a schedule, because depot shunting scheduling is a complicated work in which a lot of constraints have to be considered.

For station shunting scheduling problems, we proposed an algorithm combining probabilistic local search and PERT[1]. Depot shunting scheduling problems, however, are more complicated compared with the station shunting scheduling problems with regard to the following. Namely, we have to cope with:

- A larger number of task types.
- A larger number of workforce types.
- More complicated criteria.

This means that we have to tackle over complicated optimization problem. Recently, genetic algorithms (GA), which are stochastic search techniques based on the mechanism of natural selection and natural genetics [2], have received considerable attention regarding their potential as an optimization technique for complex problems and have been successfully applied in the area of industrial engineering [3].

For these reasons, we consider to adopt the genetic algorithms (GA). But there occurs a problem that the search space becomes too large because in solving the depot shunting scheduling problem, exact timings for the start and end of each task have to be decided. To solve the problem, we propose an idea to combine the genetic algorithm with PERT (Program Evaluation and Review Technique). We introduce a two-stage algorithm, in which a rough shunting schedule without explicitly considering the timings of each task is produced by GA, and the timings of tasks and shunting are decided by using the PERT technique.

In this paper, we introduce the algorithm together with its experimental results. In Chapter 2, we describe an outline of the depot shunting scheduling problems. Then, in Chapter 3, we introduce our algorithm for depot shunting scheduling problems combining genetic algorithm and PERT. In Chapter 4, we report the results of our experiments.
2 Depot shunting scheduling problem

2.1 An outline of a depot shunting scheduling problem

Fig. 1 is an example of a depot shunting schedule at a railway depot whose track layout is shown in Fig. 2. This diagram is called a shunting diagram. A shunting diagram illustrates a schedule of movements of rolling stock at the depot, which is depicted along the horizontal axis. Workforce assignment for tasks is also shown in the shunting diagram. As for tasks, there exist coupling, decoupling, shunting and several types of inspections and cleaning.

For example, Train X arrives at Track 10 and it is moved to Track 11 via Track T2. This is because there is no direct route between Track 10 and Track 11. On Track 11, a monthly cleaning is done by the Cleaning team A. After that, the train is shunted to Track 15 via Track T1, where a daily inspection is performed by Inspection team B.

A depot shunting schedule consists of the three factors:
1. Schedule of tasks
2. Schedule of shunting
3. Schedule of workforce assignment

In the schedule of tasks, the type, place (track) where the task is performed, start and end times are prescribed. In the schedule of shunting, origin and
destination tracks for shunting and their execution timings are specified. Workforce assignment contains assignment of workforce for each task together with the time for rest.

2.2 Constraints of shunting scheduling problems

Various kinds of constraints have to be considered in making depot shunting schedules.

(1) Constraints concerning train schedules
   A solution must not have inconsistency with planned times prescribed by the train schedule.

(2) Constraints caused by facility conditions
   - Existence of routes: There must exist a physical route between the starting point of shunting and the destination.
   - Length of tracks: Trains have to use a track whose length is larger than its length.
   - There must not exist another trains on the route of a shunting.
   - Existence of catenary: For emus and electric locomotives, there must be catenary on their routes.

(3) Temporal constraints
   - Minimum dwell times on tracks to complete assigned tasks have to be retained.
   - Shunting running time: We call the running time of trains between tracks the shunting running time. The shunting running times have to be as prescribed for particular starting and destination points. If the time is shorter, the shunting schedule is physically impossible to realize. The time shall not be longer, either, as the route is blocked for shunting during the running time and cannot be used by other trains.
- Crossover conflicts: At railway depots, two routes sometimes intersect each other. Two routes which are intersecting cannot be used simultaneously. If one of the two intersecting routes is used, trains which use the other route have to wait for a certain time, which is called the crossover conflict time, before it becomes open. This is called a crossover conflict constraint, a crucial condition that shall strictly be observed from the viewpoint of security.

(4) Constraints about tasks
- All the necessary tasks are performed.
- Tasks are done on the right tracks where appropriate facilities exist.

(5) Constraints about workforce
- Time for rest or meal is assured for each worker.
- Tasks are assigned to a worker within his working time.
- Enough time for transference between working places are retained.
- Enough time to prepare for each task is retained.
- Qualified workers are assigned for tasks.

2.3 Criteria for evaluation of shunting scheduling

We set the criteria of a shunting schedule as follows:
- Total walking distance for workers is minimum.
- Total idle time for workers is minimum.
- Workload for workers are balanced.

2.4 Depot shunting scheduling problem as a resource constrained project scheduling problem

The depot shunting scheduling problem can be regarded as a kind of resource constrained project scheduling problems (RCPSPs)[3]. But when we compare the depot shunting scheduling problem with RCPSPs commonly dealt with, the former has the following unique features.

(1) The number of activities which form a project is not given a priori and has to be dynamically changed during the process of problem solving.

(2) Not only orders of tasks but also their precise execution times have to be included in the solution. This is because we have to examine temporal constraints such as minimum dwell times, cross over conflict times, etc.

(3) In usual RCPSPs, constraints of interval times between activities are not assumed. In contrast, the shunting scheduling problem has a strong constraint on the intervals between activities. This is because shunting running times shall definitely be fixed.
3 Depot shunting scheduling algorithm combining GA and PERT

3.1 Basic idea

The overall construction of the algorithm is shown in Fig. 3. Basically, it is based on the procedure of the genetic algorithms.

It starts with an initial set of random solutions called population. Each individual in the population is called a chromosome, representing a solution to the problem. In our algorithm, one chromosome (ie. individual) corresponds to one shunting schedule. Each individual is evaluated and a fitness value is computed. Some of the individuals are selected stochastically based on their fitness value (this procedure is called a roulette wheel selection), then genetic operators such as the crossover and the mutation are applied to produce new chromosomes. The population of the next generation is thus produced by repeating this process.

Some of the characteristics of our algorithm are as follows:
- Usually, chromosomes are expressed by bit strings. In our algorithm, however, we use a kind of PERT network called an extended shunting scheduling network as a chromosome. Genetic operators are directly applied to the networks. This is because if we express complicated schedules by bit strings, this will cause a serious problem such as the result of genetic operators quite probably produces infeasible schedules.
- Chromosomes do not have explicit timing information. They only contain the information about the orderings concerning the usage of tracks and the assignment of the workforce. The exact timings of each task are determined in the evaluation phase using the PERT calculation technique.
3.2 Resource allocation by GA

(1) Shunting scheduling network
We express a shunting schedule using a kind of PERT network. We call the network an extended shunting scheduling network. An extended shunting scheduling network is described as \((N, A)\), where \(N\) is a set of nodes; \(A\) is a set of arcs and \(A \in N \times N\). A weight \(w \geq 0\) is assigned to each arc. \(N = N_1 \cup N_2\) \((N_1 \cap N_2 = \emptyset)\). These definitions mean the following.
- A node \(n \in N_1\) corresponds to an event of arrival or departure of a train.
- A node \(n \in N_2\) corresponds to an event of start or end of a task.
- An arc expresses the time base order of execution between nodes. The weight of an arc means the minimum time necessary between the occurrence of the two events on each side of the arc. Some of the arc types are shown in Table 1.

An example of an extended shunting scheduling network is shown in Fig. 4. This network corresponds to the schedule depicted in Fig. 1 (Please note that only the major arcs are shown and the weights of arcs are not shown to secure visibility).

<table>
<thead>
<tr>
<th>Arc Type</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation</td>
<td>Between arrival and departure nodes of a train</td>
</tr>
<tr>
<td>Shunting</td>
<td>Between departure and arrival nodes of a shunting work</td>
</tr>
<tr>
<td>Track</td>
<td>Between departure and arrival nodes of consecutive trains</td>
</tr>
<tr>
<td>Crossover</td>
<td>Between nodes which use intersecting routes</td>
</tr>
<tr>
<td>Planned time</td>
<td>Between the Start node and nodes which have planned time</td>
</tr>
<tr>
<td>Task</td>
<td>Between start and end nodes of a task</td>
</tr>
<tr>
<td>Transfer</td>
<td>Between an end node of a task and a start nodes of another task.</td>
</tr>
</tbody>
</table>

Fig. 4 An example of shunting scheduling network.
(2) GA operators
A chromosome in our algorithm can be considered as a combination of the schedules of each train. Each train has its own schedule such as the tracks they stay on, the tasks to be performed and the workforce assignment to the tasks.

The crossover operator is realized as an exchange of the schedule of trains. An example of a crossover operation is depicted in Fig. 5. Two children which partially inherit characteristics of their parents are produced by the crossover operation.

One of the merits of this crossover operator is that it is unlikely to produce a physically infeasible child such as a child containing a movement of trains between tracks where no physical route exist.

We also have the mutation operator, which stochastically modifies a part of a chromosome, such as the track, workforce assignment etc.

3.3 Calculation of execution times of shunting and tasks
We calculate execution time of each node for the shunting scheduling network of the candidate solutions. The time for rest of each worker (or team) is also set automatically. The calculation method is almost the same as the conventional PERT method. However, the conventional method does not assure that shunting running times become equal to prescribed ones. By adding backtracking, the

![Fig. 5. Crossover operator.](image-url)
algorithm produces an answer which satisfies shunting running time constraints. Thus, calculation of shunting times for shunting scheduling network guarantees a feasible schedule which satisfies all physical constraints imposed by facility conditions and temporal conditions as far as possible.

3.4 Evaluation

Based on the criteria we set in Section 2.3, we evaluate each chromosome using the following formula.

\[
Eval = \sum \text{walking time} + \sum \text{idle time} + a \times \text{frequency of shunting} + \sum \text{difference of working time}
\]

Here, the walking time means the time spent for transference by the workforce. The idle time means the time the workforce spend waiting for their tasks outside their office. \(a\) is an average time required for one shunting work. The difference of working time is included in the formula so that a schedule in which the workload is unbalancing among the staff who are qualified to do the same types of tasks is not produced.

4 Results of experiments

We performed experiments using actual train schedule data. The target depot is a middle size depot whose track layout is shown in Fig. 2. As for the workforce, there exist two inspection teams, four shunting drivers and a couple of cleaning teams.

We conducted ten experiments for the daytime shunting schedule of this depot, where seven trains and 13 tasks for them are included. From preliminary examination, we set the number of iteration 250.

We show the results of ten experiments in Table 2. \(\text{Train Sch.}\) denotes the number of trains departures which do not satisfy the constraints concerning the train schedule. \(\text{Workforce}\) means the number of tasks which do not satisfy the constraints of workforce, \(\text{Shunt}\) is the frequency of shunting. \(\text{Value}\) is the evaluated value for the generated schedule. \(\text{Generation}\) shows the number of iteration required until the answer was found.

In each of the ten experiments, the algorithm found solutions that satisfy all the constraints described in Chapter 2. Moreover, it has succeeded in finding solutions which can be evaluated fairly good from the practical point of view. The algorithm was implemented on a PC (Pentium III, 550MHz) in Java and it took five to thirty minutes to find the solutions.
Table 2. Results of Experiments.

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Train Sch.</th>
<th>Workforce</th>
<th>Shunt</th>
<th>Value</th>
<th>Generation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>29</td>
<td>623</td>
<td>232</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>29</td>
<td>572</td>
<td>110</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>29</td>
<td>619</td>
<td>102</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>29</td>
<td>603</td>
<td>128</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>29</td>
<td>588</td>
<td>40</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>29</td>
<td>633</td>
<td>234</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>29</td>
<td>603</td>
<td>117</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>29</td>
<td>591</td>
<td>179</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
<td>29</td>
<td>643</td>
<td>102</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>29</td>
<td>589</td>
<td>68</td>
</tr>
</tbody>
</table>

5 Conclusions

We have introduced an algorithm combining the genetic algorithm and PERT for depot shunting scheduling problems. This algorithm is applicable to various kinds of track layouts and train schedules, since no knowledge or conditions specific to each depot are employed. We have proved that the algorithm works well and confirmed its effectiveness through experiments using actual train schedule data.

References