Knowledge-Based Railway Scheduling System

Te-Wei Chiang and Hai-Yen Hau
Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan 10617, R.O.C

Abstract - In railway system, the timetable containing the arrival and departure times of all trains at each station is most essential schedule for train operation. Due to the size of the problem, conventional operation research's optimization procedures are not applicable. In this paper, we propose a knowledge - based approach to railway scheduling. The simulation result the resulting schedule in local scheduling has satisfactory performance.

I. INTRODUCTION

The operation of railway systems requires solving various types of scheduling problems, such as timetable preparation and resource allocation. Because they are large-scale combinatorial problems and have many conflicting criteria, railway scheduling problems, therefore, still depend heavily on the flexible information-processing and decision-making abilities of human experts.

In this paper, we proposed an architecture for railway scheduling based on knowledge-based technology. We divide the scheduling problem into two parts : the global scheduling and the local scheduling. In global scheduling, the tracks between two stations is assigned to trains and the initial train diagram will be established. In local scheduling, the tracks in a station are assigned to trains safely according to the result of global scheduling. The local scheduling is very similar to conventional manufacture scheduling in manufacturing applications. In this paper, only local scheduling is discussed.

Section two surveys the railway scheduling problems. Based on this observation, we propose a new problem solving architecture in section three. Section four describes the local scheduling algorithm. Section five simulates the proposed local scheduling architecture. Finally, section six summarizes our architecture.

II. PROBLEM DESCRIPTION

In railway system, the timetable that contains the arrival and departure times of all trains at each station is most essential schedule for train operation. The three components of the train scheduling problem are : the time table, the criteria that measure the performance of operations, and the physical limitations of resources.

A. Timetable : for each train, the timetable contains the following information

- 1) the arrival time at each station,
- 2) the departure time at each station, and
- 3) the assigned track/platform at each station.

B. Criteria : the goals of train scheduling, or measures that indicate the quality of the train schedule.

- 1) Optimization of passenger's service
- 2) Minimization of operation costs

C. Physical constraints : include the safety regulations for train operations and the limitation of station configuration. These constraints are :

- 1) The minimum running time between two stations (this indicates the shortest time needed for a train to travel from one station to the next station)
- 2) The minimum stopping time at each station (the time needed for passengers to get on or off the train)
- 3) Minimum time between arrivals at each station (Due to the safety concern, after one train arrive, the following arrival train have to keep a headway time)
- 4) Minimum time between departures at every station (Due to the safety concern, after one train depart, the following departure train have to keep a headway time)
- 5) Minimum time of turning back at the terminal (At the terminal, the train have to turn back in order to run backward)
- 6) Maximum number of trains running between two stations (Between two stations, the maximum number of trains is limited for safe reason)

Typically, such problems can be formulated as an integer programming problem. However, in railway scheduling problem, the size of the problem makes it impossible to be solved by such approach (For example, the main North-South railway in Taiwan has about 100 stations each has it

own rail configuration, and about 200 trains of different speed are to be scheduled daily). For this problem, there will be $(200!)^{100}$ possible schedules. To further complicate the matter, the criteria for measuring performance are often contradictory. For example, an increase in service frequently results in high quality service, but also high operation costs.

Our approach to solving this problem is to divide the schedule problem into two parts : the global scheduling and the local scheduling. In local scheduling, the tracks in a station is assigned to trains safely according to the result of global scheduling without violating any constraint.

In this paper, we make two assumptions. First, there are a pair of tracks between any two stations, one for northbound trains and one for southbound trains. Second, the trains are operating at the same speed. With these two assumptions, we can focus on the local scheduling problem and ignore the global scheduling problem in real world (such as crossing and overtaking).

III. PROPOSED ARCHITECTURE

The proposed architecture for local scheduling is shown in Figure 1. The input to the local scheduling algorithm are the result of the global scheduling and routing tables. The local scheduling algorithm use rules and constraint propagation technique to generate a feasible local schedule.

Each station has two routing tables, one for northbound routes and one for southbound routes. Given the station layout as shown in Figure 2, user need to transform the station layout into two routing tables, the southbound routing table and the northbound routing table, in which all possible routes will be listed. The routing tables for the station shown in Figure 2 are shown in Table I and Table II. Note that a track may be contained in different routes. The southbound routing table describes all possible routes in the southbound direction, and the northbound routing table describes all northbound routes. According to the routing tables, we

Fig. 1 System Architecture of Local Scheduling

Fig. 2 A sample station configuration

can find the relationship between every pair of routes. Besides, the routing table contains the information for the detection of type III conflicts. The Before_Track_Points column and the After_Track_Points column in the routing table indicates fork points along the route. Positive fork point indicates the route branch to the fork, while negative fork point indicates the route bypass the fork.

TABLE I
SOUTHBOUND ROUTING TABLE

Route Priority	Before Track Points	Track ID	After Track Points	Pass Platform or not
1	6,5	2	4,-7	YES
2	6,5,-8	1	7	YES
3	6,-5	3	-4,-7	YES

TABLE II
NORTHBOUND ROUTING TABLE

Route Priority	Before Track Points	Track ID	After Track Points	Pass Platform or not
1	-3,4	3	5,6	YES
2	3,7,-4	3	5,6	YES
3	3,4	2	8,5,-6	YES
4	-3,-1	4	-2,5,6	YES

There are five basic types of conflict that may occur in local scheduling:

Type I: A track is assigned to different trains at the same time.

Type II : The time interval between the two trains which have been allocated to the same track is violation of minimum time lag request (the legal time lag in this case is 6 mins.).

Type III : If the time lag of some pairs of trains are not enough, then violate the minimum intersection time lag (the permissible time lag in this case is 3 mins.).

Type IV : Minimum time between arrivals is violated. (assume the minimum headway time for arrival is 2 mins.)

Type V : Minimum time between departures is violated. (assume the minimum headway time for departure is 2 min.)

Figure 3 illustrates the type III conflict. Train A bypasses fork point 6 and branches to fork point 5, then enters the track 3. Train B bypasses fork point 5 and branches to fork point 6. So if the time lag between train A's arrival time and train B's departure time is not enough, then type III conflict occurs. According to the previous observation, we find that if any two trains pass through the same fork point (no matter branch to the fork or bypass the point) and one of the two trains branch to the fork point, then type III conflict occur.

Fig. 3 An example of type III conflict

From the above observation, we can infer rules which can be used for conflict detection. The purpose of the following rules is to check the given train due to the selected route conflicts or not.

RULE C1

IF (the arrival time of the train $< T_k$)

THEN type I or type II conflict occurs

RULE C2

IF (any two trains pass through the same fork point **AND** one of the two trains branch to the fork point)

THEN type III conflict occurs

RULE C3

IF ((the train is southbound)**AND**(the arrival time of the train $< S_a$))

THEN type IV conflict occurs

RULE C4

IF ((the train is southbound)**AND**(the departure time of the train $< S_d$)

THEN type V conflict occurs

In order to determine the route's priority, we use the following priority rules:

RULE P1

IF (the number of crossover points route A run through $>$ the number of crossover points route B run through)

THEN the priority of route A $>$ the priority of route B

RULE P2

IF (the number of crossover points route A run through is equal to the number of crossover points route B run

through) **AND** (the number of bypass points of route A $>$ the number of bypass points of route B)

THEN the priority of route A $>$ the priority of route B

RULE P3

IF (the number of crossover points route A run through is equal to the number of crossover points route B run through) **AND** (the number of bypass points of route A is equal to the number of bypass points of route B) **AND** (the number of Before_Track_Points of route A $>$ the number of Before_Track_Points of route B)

THEN the priority of route A $>$ the priority of route B.

IV. LOCAL SCHEDULING ALGORITHM

The local scheduling algorithm uses the output from global scheduling and routing tables as the input data. The output of the local scheduling algorithm is the desired feasible local schedule. The main job of local scheduling algorithm is to assign tracks to trains in a conflict-free manner.

The local scheduling algorithm is described as below:

STEP1: Initialization (read input data from the global schedule). If the input train violate the minimum headway time (i.e. type IV or type V conflict occurred), then rectify the input data by shifting the arrival time and departure time of that train.

STEP2: Select and delete one train from the set of schedulable trains. According to the type (northbound or southbound) and the arrival and departure time of the train, select the highest priority route from routing table with no conflict.

STEP3: If no route has been selected in STEP2, then assign the highest priority route to the train as early as possible and propagate constraints according to the resources (such as tracks and crosspoints) availability, else propagate constraints according to the resources availability in STEP2.

STEP4: If the set of schedulable trains are not empty, then goto STEP2. else stop.

In order to resolve conflicts, we used a technology called *constraint propagation*. we use five variables and a constraint set to propagate constraints and make schedule according to these constraints so that the five types of conflict can be avoided. These variables and constraint set are defined as follows: T_i is the earliest available time of track i . It is used to prevent type I and type II conflict. S_a and N_a are the earliest available arrival time for south bound and north bound. They are used to prevent type IV conflict. S_d and N_d are the earliest available departure time for south bound and north bound. They are used to prevent type V

conflict. C_T is a constraint set using to prevent type III conflict, which contains a number of constraint elements. Each element is of the form (start_time, end_time, constraint_element_1, ..., constraint_element_n), which denotes the fork points (constraint_element_1, ..., constraint_element_n) has been affected between during the time interval (start_time, end_time).

For example, if there is a train occupying track K from time t_1 to t_2 , then the earliest available time for track K is $t_2 + 6$ minutes. So if a train has been assigned a route passing through track K from time t_1 to t_2 , then the variable T_K is equal to $t_2 + 6$, and if there is a later train pass through T_K before $t_2 + 6$ then type I conflict occurs. The constraint propagation by other four variables is similar. Furthermore, as shown in Figure 3, if there is a train A arriving the station at time t_0 then the fork points 5 and 6 are constrained during the time interval $(t_0 - 3, t_0 + 3)$ due to type III conflict. Therefore, we can add $(t_0 - 3, t_0 + 3, -5)$ to the constraint set C_T by looking up south bound routing table. So if there is another train passing through fork point 6 or bypass fork point 5 between $t_0 - 3$ and $t_0 + 3$, then type III conflict occurs. Be aware of the sign of the fork point's value. If the sign of a fork point's value is negative, then no matter the other train is passing through or bypass the fork point, conflict will occur. Conversely if the sign of a fork point's value is positive, then only the train pass through the fork point will be affected. For example, in Figure 3, we can get the Before_Track_Points of the route of train B is 5 and -6, which means when train B arrives the station, it bypass fork point 5 and branch to fork point 6. Therefore, if the train B's arrival time is between $t_0 - 3$ and $t_0 + 3$, then there is a conflict between train A and train B. The assignment of the five variables described previously and the addition of $(t_0 - 3, t_0 + 3, 6, -5)$ to C_T can be looked upon as a kind of constraint propagation.

V. SIMULATION

The input data to local scheduling is from global scheduling, and the data format is (southbound/northbound, train_id, arrival_time, departure_time).

The input data from the global schedule of station s3, as shown in Figure 4, are

(s,T1,0,5), (n,Ta,3,8), (s,T2,10,15), (n,Tb,10,15), (s,T3,15,20), (n,Tc,20,20), (n,Td,25,20), (s,T4,30,30), (n,Te,30,35), (s,T5,35,40), (s,T7,45,55), (n,Tf,50,50), (s,T6,48,48), (n,T8,51,51), (n,Tg,55,60). For convenient, we only consider the local scheduling between 7:00 and 8:00, and we set the start time 7:00 to zero in input data.

According to the routing table, we can find the relationship between every pair of routes (see Fig. 5).

Fig. 4 Another example of global schedule

From the priority rules, user can infer the needed routing tables, as shown in Table I and Table II. Apply the local scheduling algorithm described in section 3 and use the example global schedule as input. We can obtain the resulting local schedule (as shown in Figure 6).

Southbound Routes:

- 1.D → 6 → 5 → 2 → 4 → 7 → B
- 2.D → 6 → 5 → 8 → 1 → 7 → B
- 3.D → 6 → 5 → 3 → 4 → 7 → B

Northbound Routes:

- 1.A → 3 → 4 → 3 → 5 → 6 → C
- 2.A → 3 → 7 → 4 → 3 → 5 → 6 → C
- 3.A → 3 → 4 → 2 → 8 → 5 → 6 → C
- 4.A → 3 → 1 → 4 → 2 → 5 → 6 → C

□ : Track
○ : Runthrough Point
● : Bypass Point

Fig. 5 All possible routes in the example

Fig. 6 The track assignment of Figure 7's station 3

VI. CONCLUSION

We divide the scheduling problem into two parts : the global scheduling and the local scheduling. In global scheduling, the tracks between two stations is assigned to trains and the train diagram will be established. In local scheduling, the tracks in a station is assigned to trains according to the result of global scheduling without violating any constraint. In this paper, only local scheduling is discussed. We proposed an architecture for railway scheduling based on knowledge-based technology and constraint propagation technique. We use human's knowledge to select adequate route and exert constraint propagation to resolve conflict. The simulation shows the resulting schedule has satisfactory performance and efficiency .

REFERENCE

- [1] K. Komaya and T. Fukuda, "A Knowledge-based Approach for Railway Scheduling," *CAIA*, pp.405-411,1991.
- [2] B. Fringuelli,"Knowledge-Based Technology for Controlling Railway Stations," *IEEE EXPERT*, pp.45-52, December 1992.
- [3] K. Baker: "Introduction to Sequencing and Scheduling", John Wiley & Sons, 1974.
- [4] S. J. Noronha; V. V. S. Sarma : "Knowledge-Based Approaches for Scheduling Problems: A survey", *IEEE Trans. on Knowledge and Data Engineering*, vol. 3, No. 2, June 1991, pp. 160-171.
- [5] J. R. Slagle; H. Hamburger : "An Expert System for a Resource Allocation Problem", *Communication of ACM*, vol.28, no. 9, pp. 994-1004, 1985.
- [6] J. E. Cury, F. A. C. Gomide, and M.J. Mendes : "A Methodology for Generation of Optimal Schedules for an Underground Railway System", *IEEE Trans. on Automatic Control*, vol. AC-25, No.2, April 1980.
- [7] Mark S. Fox and Stephen F. Smith : "ISIS - A Knowledge-based System for Factory Scheduling", *EXPERT SYSTEMS*, pp.25-49, 1984, vol.1, No.1.