한국철도학회논문집
제8권 제5호 2005년 ▌pp. 391-397

# 한국고속철도를 위한 차량운용 및 할당시스템

# KTX Trainset Maintenance Routing and Allocation System for Korea High-Speed Rail

홍순흠* · 김성호†
Soon-Hum Hong · Seongho Kim

## Abstract

In this paper we present a model for the maintenance routing construction and the trainset allocation (maintenance routing problem). The model solves the maintenance routing problem using column generation algorithm which was used to combine constraint programming and linear programming. Ilog-cplex was used to solve the linear programming model and Ilog-solver was used to solve the constraint programming model. The computational experience is also provided.

***Keywords*** : Maintenance Routing(유지보수운용), Trainset Allocation(차량할당), Set Partitioning Problem(집합분할문제), Column Generation Technique(열생성기법)

## 1. 서 론

Train schedule is usually determined based on traffic forecasts for the demand variations, and is developed manually or by the help of decision support system. Once the schedule is determined, the next step is to decide how to efficiently allocate resources to serve the train schedule. Korea High-speed railway (KHR) is constantly investigating new ways to reduce the costs and improve the service.

One good alternative is to increase the utilization of the rolling stock by a better assignment of the available trainsets to the scheduled trains. In this paper we use the term train to refer to a particular service between connecting points along an itinerary called a train path.[1)]

The dispatcher has the daily task of assigning available trainsets from the fleet to individual routes. It is obvious the trainset can be assigned to the routes in different ways. In addition, the number of hours (or days) is known that each trainset can serve before going for a mandatory technical checkup and parts replacement.

KHR has two technical bases where this work is done. It is certain that the railway is interested in keeping trainsets in service as long as possible before they go to the technical base for maintenance. The dispatcher encounters the following practical problem on a daily basis: Assign the available trainsets from the fleet to specific routes so that the trainsets are kept in normal operation for as long as possible before going to the technical base. This trainset assignment to routes (making a maintenance routing) is combinatorial by nature. Although skilled dispatchers can make these assignments rather successfully, it is clear that the combinatorial nature of the problem requires the utilization of certain models and modern computing algorithms in order to achieve the best possible effects.

Booler(1980) developed a heuristic method based on multi-commodity flows in a network as a trainset routing construction and a allocation method. Wright(1989) proposed a series algorithms that solve the locomotive

† 책임저자 : 회원, 인하대학교 경영대학 경영학부, 조교수
E-mail : shk7768@inha.ac.kr
TEL : (032)860-7768 FAX : (032)123-1234
* 한국철도기술연구원 운영시스템연구그룹, 책임연구원

routing problem using variants of the Hungarian method. However Wright's method does not take into account of the locomotive maintenance. In this paper we present a system for the maintenance routing construction and the trainset allocation.

## 2. Model Development and Design

The basic approach is to develop a mathematical model to determine optimal routes for all trainsets in a given schedule that meet the defined maintenance requirements for a specific planning horizon. The output of the model is a series of N-day trainset routings for each trainset in a schedule which satisfy the maintenance requirements.

Before the model can be described in detail, specific terms need to be defined. These include the followings: One-Day Routings (ODR) is a series of trains that do not span more than one day. Series of One-Day Routings (SOR) is a series of linked ODRs that meet the location-time constraint. The meaning of this constraint is that the starting location of an ODR in a path must be the end location of a previous ODR of the path and the starting time of the next ODR has to be greater than the end time of the next ODR. Feasible maintenance SOR is a SOR that meet given maintenance requirements. Maintenance Routing is a complete set of feasible maintenance SORs that cover the planning period for each trainset.

The objective of our system is to create the cheapest maintenance routing. We use the column generation algorithm (Barnhart et al. 1998) to solve the maintenance routing problem. This algorithm has been widely accepted as an attractive technique to solve large scale optimization problems in transportation such as our problem. In the model formulation, the following notation is used:

$k$ is an index for a trainset;
$K$ is set of all trainsets;
$P$ is a feasible maintenance SOR;
$\Omega_k'$ is a set of feasible maintenance SORs for trainset $k$;
$c_P^k$ is cost coefficient for a feasible maintenance SOR $P$ for trainset $k$;
$\delta_{pP}$ is an incidence coefficient for ODR $p$ for a feasible maintenance SOR $P$. $\delta_{pP}=1$ if ODR $p$ is included in the feasible maintenance SOR $P$.
$P^R$ is set of all ODRs;
$\delta_{qP}$ is an incidence coefficient for maintenance activity $q$ for a feasible maintenance SOR $P$;
$P^M$ is set of all type of maintenance activity; $q \in P^M$ is defined by maintenance station, by maintenance type, and by maintenance type specific duration. $\delta_{qP}=1$ if maintenance activity $q$ is included in the feasible maintenance SOR $P$.
$b_q$ is the number of trainsets (capacity) which can be covered by the maintenance station;
$x_P^k$ is binary decision variable for a feasible maintenance SOR $P$ for trainset $k$.

Our maintenance routing problem can be modeled as a generalized set partitioning problem as follows.

(SPP)

$$\text{Minimize} \quad \sum_{k \in K} \sum_{P \in \Omega_k'} c_P^k x_P^k \tag{1}$$

$$\text{subject to} \quad \sum_{k \in K} \sum_{P \in \Omega_k'} \delta_{pP} x_P^k = 1 , \quad \text{for } p \in P^R \tag{2}$$

$$\sum_{k \in K} \sum_{P \in \Omega_k'} \delta_{qP} x_P^k \le b_q , \quad \text{for } q \in P^M \tag{3}$$

$$x_P^k \in \{0,1\}, \quad \text{for } \quad k \in K, \quad P \in \Omega_k' \tag{4}$$

The column generation algorithm proceeds as follows: The master problem (MP) is the linear programming relaxation of the model (A). The simplex method is used to solve the master problem for a set of current feasible maintenance SOR (columns), in the process calculating dual variables to be used for pricing out new columns. The subproblem (SP) is used to check whether any new columns with a negative reduced cost exits. If so, they are added to the current linear programming relaxation of the model (A) which is then re-optimized; if not, the solution to the current linear programming relaxation of the model (A) is optimal. The subproblem is a constrainted shortest path problem. There is a specific subproblem (a network)

1) In this paper we use the term train to refer to a particular service between connecting points along an itinerary called a train path.

associated with each trainset. The paths (SORs) generated in each network are feasible maintenance SOR for a specific trainset. Given the dual variables provided by the current master problem solution, the cost associated with a path represents the reduced cost of a feasible maintenance SORs in the master problem.

## 2.1 The Master Problem

Constraints that apply to the master problem are followings.

- The first constraint of the model (A) ensures that each ODR need to be covered by exactly one column (feasible maintenance SOR).
- The second constraint of the model (A) ensures that each maintenance activity can not exceed the capacity of the maintenance station.
- The third constraint of the model (A) ensures that exactly one column is assigned to each trainset.

In the column generation algorithm the linear programming relaxation of the model (A) is solved repeatedly with a given set of columns. we solve the linear programming relaxation using the linear programming library Ilog-Cplex (Ilog 2001).

## 2.2 The Sub-problem

It should be noted that each column (feasible maintenance SOR) is generated independently. By decomposing the problem in two phases using the column generation algorithm it is possible to model difficult real life constraints. The reason for this is that by decomposing it is possible to consider the constraints in two phases. Some constraints only apply to individual SOR and these can be enforced in the generation phase as each column is generated independently. At the end of the generation phase all the feasible maintenance SORs (columns) will be legal with respect to all the constraints that apply to the individual column.

Constraints that apply to the subproblem are followings.

- The starting location of an ODR in a feasible maintenance SOR (column) must be the end location of a previous ODR of the column.
- The starting time of the next ODR has to be greater than the end time of the current ODR.
- Each maintenance activity has [minDays, maxDays] and [minDistance, maxDistance] constraint to indicate: the gap between the end date of first maintenance activity and start date of second maintenance activity must be within [minDays, maxDays], the total traveling distance between two consecutive maintenance activities must be within [minDistance, maxDistance]. The maintenance activity should be scheduled as late as possible within both ranges.
- Users can add some flexibilities by defining [minDays, maxDays+relaxedDays] and [minDistance, maxDistance+relaxedDistance] in the sense to relax the constraint. However, the cost will be different.
- There is a hierarchy for the maintenance activities. Each maintenance activity may have 0/1 parent maintenance activity. If a higher level maintenance activity is scheduled, it means it includes all the lower level maintenance activities (descendents). For instance doing 24 (Limited Inspection: LI) also means it has done 23 (Systematic Work on Wheel: SWW), 22 (Comfort Examination and Running Gear Inspection: CE/RGI). For constraints checking, it means higher level maintenance activity [startDate, endDate] implies it has one lower level maintenance activity starting on startDate and one ending at endDate for each descendent. There is no need to check any constraint within [startDate, endDate] range.
- It should be allowed to fix some ODR or/and maintenance activities on a day.
- For each trainset, there is information about the accumulative distance or day since last maintenance activity for each type of maintenance activity and current location on the day before the first planning horizon. However if the accumulative days is zero, it means either this maintenance activity finish on the day before the planning horizon or it means it's in the middle of the maintenance. The start date of this maintenance activity may be less than or equal to that day. The end date may be in planning horizon.
- Each trainset must go through a list of maintenance activities. Each maintenance activities may use certain line resource in terms of hours (< one day), or days or shifts as duration. If the duration is less than or equal

to one shift, this maintenance can not cross two shifts, otherwise, it's free to move within a day. If the duration is in terms of days, it may indicate a further constraint to say it must start on certain day type.

We use constraint programming (CP) to carry out the generation of columns. Constraint programming provides primitives for declaring decision variables, stating constraints and solving the resulting problems. We use the CP library Ilog-Solver (Ilog 2001) for the subproblem.

## 2.3 The Cost Coefficients

The objective function of the model (SPP) is a linear combination with cost coefficients $c_P^k$ and its respective decision variables $x_P^k$. Without loss of generality we can omit the superscript and subscript of cost coefficient. We calculate the cost coefficient $c$ as following.

$$c = \sum_{i \in I} (F_i + V_i) \tag{4}$$

where $F_i$ is fixed cost of maintenance activity $i$, $V_i$ is variable cost of maintenance activity $i$, and $I$ is set of all type of maintenance activity lists. The variable cost $V_i$ of the above (4) is calculated as the following.

$$V_i = \sum_{j \in J_i} F_j \left( \frac{RT_j}{T_j^{\max}} W_j^T + \frac{RD_j}{D_j^{\max}} W_j^D \right) + \sum_{j \in J_i} F_j \left( \frac{OT_j}{T_j^{ext}} W_j^T + \frac{OD_j}{T_j^{ext}} W_j^D \right) \tag{5}$$

where $J_i$ is union of {a set of maintenance activities which are maintenance activity $i$'s lower level maintenance activities in the hierarchical relationship} and {maintenance activity $i$}. For example, if $i$ is 24 (Limited Inspection: LI), then $J_i$ is {24 (LI), 23 (SWW), 22 (CE/RGI)}. $F_j$ is fixed cost of maintenance activity $j$. $T_j^{\max}$ is length of time cycle (maximum time limit) of maintenance activity $j$. $D_j^{\max}$ is length of distance cycle (maximum distance limit) of maintenance activity $j$.

## 2.4 Finding an Integer Solution

As the original model is an integer programming problem, it is necessary that all decision variables take integer values. Thus, a branch and bound phase is required to obtain the integer optimal solution. The branching rule we use is motivated by a general rule for set partitioning problems developed by Ryan and Foster (1981). Their rule is based on the simple observation that given a fractional solution to the linear programming relaxation of a set partitioning problem there must exist two columns whose associated variables are factional such that they both contain coefficients of one in a common row $r$ and there exists another row $s$ where one column has a coefficient of one and the other has a coefficient of zero. This fact leads to a general branching rule where we require rows $r$ and $s$ to be covered by the same column on one branch and by different columns on the other. If ODRs $r$ and $s$ satisfy the conditions of the Ryan and Foster rule and they appear consecutively in at least one of the fractional pairings that contains them both, we can branch by requiring that they appear consecutively in the pairing that covers them at one node and by requiring that they cannot appear consecutively in any pairing in the solution at the other node. We refer to the ODR pair $r$ and $s$ as a follow-on. Given a fractional solutions to the master problem we can identify two ODRs $r$ and $s$ such that the subset $P_{rs}$ of pairings that contains $r$ and $s$ consecutively has the property $0 < \sum_{j \in P_{rs}} x_j < 1$ . In general, given a fractional solution to the master problem, there are many follow-on pairs that can be used to branch. Thus we need a priority rule to determine which pair of ODRs we will branch on given a fractional solution. Given a fractional solution to the linear programming relaxation of model (SPP), we compare candidate follow-ons $r$, $s$ by comparing the value of $\sum_{j \in P_{rs}} x_j$ for the different follow-on pairs. The higher the value of this sum, the better the follow-on $r$, $s$ looks with respect to the linear programming solution. If the value of the sum close to one, it seems quite likely that the follow-on pair is covered by the same column in a good feasible solution to the integer programming problem.

## 3. Computational Experience

In this section we will provide the results from our computational experience. The characteristics of our sample

**Table 1.** Characteristics of Sample Data Set

| | |
|---|---|
| Number of ODRs | 231 |
| Number of Train (including deadhead trains) | 1,335 |
| Number of Trainsets | 46 |
| Number of Maintenance Activity Types | 8 |

**Table 2.** Cycle Lengths of Maintenance Activities

| Maintenance Activity | Length of Time Cycle (days) | Length of Distance Cycle (km) |
|---|---|---|
| CE/RGI | 9 | 15,000 |
| SWW | 30 | 50,000 |
| LI | 90 | 150,000 |
| GI | 180 | 300,000 |
| FGI | 365 | 600,000 |
| NC | 4 | 7,000 |
| DC | 30 | 50,000 |
| MC | 365 | 600,000 |

data set are given in Table 1.

The cycle lengths of 8 maintenance activities are given in Table 2.

CE/RGI = Comfort Examination / Running Gear Inspection
SWW = Systematic Work on Wheel
LI = Limited Inspection
GI = General Inspection
FGI = Full General Inspection
NC = Normal Cleaning
DC = Detail Cleaning
MC = Master Cleaning

We tried several executions of the column generation algorithm which have different control parameter values. All the execution times are given in Table 3.

Subset size is the parameter to choose size of a subset of duty network. Duty graph max connection is the parameter to control how many arcs will be kept in duty graph for each service leg(ODR)'s all possible connections.

Connection max day gap is the parameter for heuristics to control what kind of connection is considered. If the original problem is feasible, which means all set partitioning rows are covered and all resource constraints are

**Table 3.** Results of Execution Times

| ID | Control parameters | | | | | Execution time (minutes: seconds) |
|---|---|---|---|---|---|---|
| | Subset size | Duty graph max connection | Connection max day gap | Initial time limit | Time limit | |
| 1 | 50 | 300 | 1 | 180 | 10 | 5:40 |
| 2 | 50 | 300 | 2 | 180 | 10 | 8:10 |
| 3 | 50 | 300 | 1 | 90 | 10 | 3:56 |
| 4 | 50 | 300 | 1 | 60 | 10 | 2:32 |
| 5 | 50 | 300 | 1 | 30 | 10 | 1:34 |
| 6 | 30 | 200 | 1 | 15 | 10 | 1:13 |
| 7 | 15 | 150 | 1 | 30 | 10 | 1:10 |
| 8 | 30 | 200 | 1 | 15 | 10 | 0:56 |
| 9 | 30 | 200 | 1 | 5 | 10 | 1:07 |
| 10 | 30 | 200 | 1 | 15 | 10 | 0:53 |
| 11 | 20 | 200 | 1 | 15 | 10 | 1:15 |
| 12 | 40 | 250 | 1 | 15 | 10 | 1:11 |

satisfied (by checking slacks), the maximum time (specified by Initial time limit) will be spent before doing branch & price. Max time (specified by Time limit) will be spent to generate columns for a subset network before calling the linear programming solver solveLp (normally every 20 columns generated). If it's difficult to look for columns within max time, it may only generate less-than-twenty columns.

A solution (maintenance routings of all trainsets) obtained is given in Figure 1 as a Gantt chart. This Gantt chart was generated by the SAS code provided in the appendix.The planning horizon of the solution is 7 days (June 02 ~ June 09).

The first one is a group of ODRs. The second one is a group of maintenance activities. Horizontal bar for ODR has a label indicating ODR ID such as K10014. Horizontal bar for maintenance activity has a label indicating the maintenance type.

## 4. Conclusion

We presented a model for the maintenance routing construction and the trainset allocation (maintenance routing problem) in this paper. The model solves the maintenance routing problem using column generation algorithm which

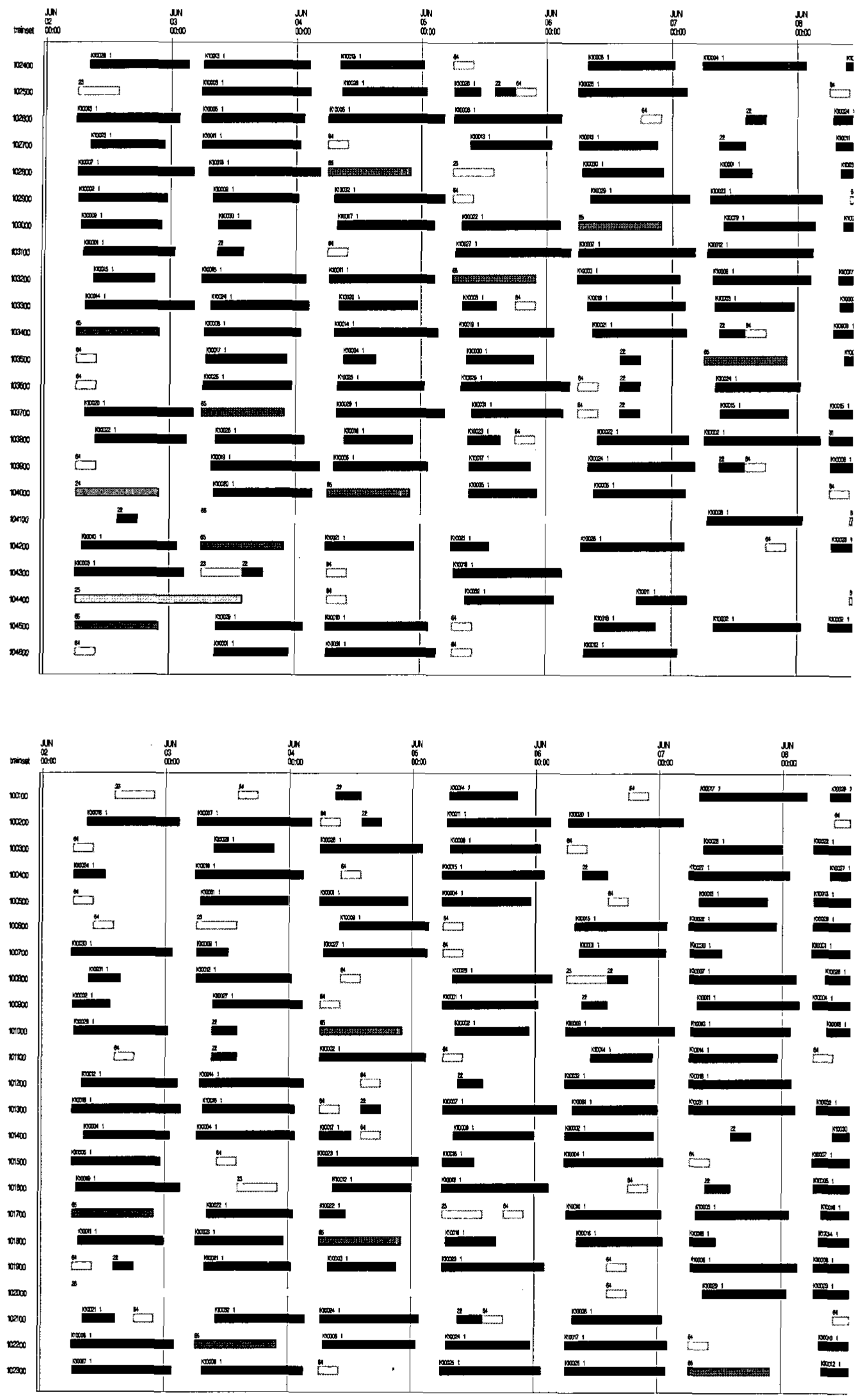
Fig. 1. Maintenance Routings for All Trainsets

was used to combine constraint programming and linear programming. Ilog-cplex was used to solve the linear programming model (master problem). Ilog-solver was used to solve the constraint programming model (subproblem). The model offers what-if capabilities that permit dispatchers to evaluate changes to schedules and to evaluate the economic effects of maintenance rules. An important practical issue for further work is improving the implementation of the model.

## 참 고 문 헌

1. Barnhart, C., E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance (1998), "Branch and Price: Column Generation for Huge Integer Programs," Operations Research Vol.46, pp.316-329.
2. Booler, J. M. (1980), "The Solution of Railway Locomotive Scheduling Problem," Journal of the Operational Research Society, Vol.31, pp.943-948.
3. Clampett, S. M., C. J. Timken, J. P. Chen, and C. P. Sebastian, "Maintenance Station Planning," AGIFORS proceedings, (Oct 20, 1986).
4. Ilog, ILOG Cplex 8.0 User's Mannual, (April 2001).
5. Ilog, ILOG Solver 5.1 User's Mannual, (April 2001).
6. Ryan, D. M. and B. A. Foster, "An Integer Programming Approach to Scheduling," in A. Wren (ed.), Computer Scheduling of Public Transport Urban Passenger Vehicle and Crew Scheduling, North-Holland, Amsterdam (1981) 269-280.
7. Wright, M. B. (1989), "Applying Stochastic Algorithms to a Locomotive Scheduling Problem," Journal of the Operational Research Society, Vol.40, pp.187-192.