Topological Constraints in Periodic Train Scheduling

M.A. Salido[1], M. Abril[2], F. Barber[2], L. Ingolotti[2], P. Tormos[3], A. Lova[3]
DCCIA[1], Universidad de Alicante, Spain
DSIC[2], DEIOAC[3], Universidad Politécnica de Valencia, Spain
{mabril, msalido, lingolotti, fbarber, }@dsic.upv.es
{ptormos, allova}@eio.upv.es

Abstract.

It is well known that many scheduling problems can be modeled as constraint optimization problems. The scheduling of train services can be considered as a problem subject to a number of constraints describing railway infrastructure, required train services and reasonable time-intervals for waiting and transits. Railway optimization problems are known to be hard problems and a good solution or the best solution is a rather difficult task. In this work, we propose a topological constraint optimization technique for solving periodic train scheduling, developed in collaboration with the National Network of Spanish Railways (RENFE). This topological technique transforms the railway optimization problem in subproblems such that a traffic pattern is generated for each subproblem. These traffic patterns will be periodically repeated to compose the entire running map. The results show that this technique improve the results obtained by well known tools as LINGO and ILOG Concert Technology (CPLEX).

1 Introduction

Over the last few years, railway traffic has increased considerably, which has created the need to optimize the use of railway infrastructures. This is, however, a hard and difficult task. Thanks to developments in computer science and advances in the fields of optimization and intelligent resource management, railway managers can optimize the use of available infrastructures and obtain useful conclusions about their topology.

The overall goal of a long-term collaboration between our group at the Polytechnic University of Valencia (UPV) and the National Network of Spanish Railways (RENFE) is to offer assistance to help in the planning of train scheduling, to obtain conclusions about the maximum capacity of the network, to identify bottlenecks, to determine the consequences of changes, to provide support in the resolution of incidents, to provide alternative planning and real traffic control, etc. Besides of mathematical processes, a high level of interaction with railway experts is required to be able to take advantage of their experience.

Different models and mathematical formulations for train scheduling have been created by researchers [10, 4, 5, 9, 7, 3, 6, 2], etc. Several European companies are also working on similar systems. These systems include complex stations, rescheduling due to incidents, rail network capacities, etc. These are complex problems for which work in network topology and heuristic-dependent models can offer adequate solutions.

In this paper, we propose a topological constraint optimization technique for solving periodic train scheduling. This technique has been inserted in our system [1] and it is committed to solve this problem in order to obtain as good and feasible running map as possible. The system is able to plot the obtained running map. A running map contains information regarding railway topology (stations, tracks, distances between stations, traffic control features, etc.) and the schedules of the trains that use this topology (arrival and departure times of trains at each station, frequency, stops, junctions, crossing, etc,) (Figure 1). In our system, the railway running map problem is formulated as a Constraint Optimization Problem (COP). Variables are frequencies, arrival and departure times of trains at stations. Constraints are composed by user requirements and the intrinsical constraints (railway infrastructures, rules for traffic coordination, etc.). These constraints are composed by the parameters defined using user interfaces and database accesses. The objective function is to minimize the journey time of all trains. The problem formulation is (traditionally) translated into a formal mathematical model to be solved for optimality by means of mixed integer programming techniques. In our framework, the formal mathematical model is partitioned in two different subproblems: integer programming problem composed by the constraints with integer variables and linearized problem in which there are now variables of type real remaining to be assigned. Therefore, the problem constraints are classified such that most restricted constraints are studied first [11]. This is based on the *first-fail* principle, which can be explained as

”To succeed, try first where you are more likely to fail”

The most restricted constraints are considered to be composed of integer variables. In this way, our system studies first the integer programming problem and then it solves the linearized problem. The integer programming problem will be partitioned in a set of subproblems such that the solution of each subproblem will generate a traffic pattern. The partition is carried out through the stations that take part in the running map. Each block of the partition is composed by contiguous stations, so that each traffic pattern represents the running map corresponding to each block of constraints. In Figure 1, a posible block of the partition may be composed by the first four stations: *Malaga Cent, Malaga Renfe, Los Prados* and *Aeropuerto*. Each traffic pattern will be periodically repeated to composed the entire running-map.

2 Problem Topology

A sample of a running map is shown in Figure 1, where several train crossings can be observed. On the left side of Figure 1, the names of the stations are presented and the vertical line represents the number of tracks between stations (one-way or two-way). The objective of the system is to obtain a correct and optimized running map taking into account: (i) the railway infrastructure topology, (ii) user requirements (parameters of trains to be scheduled), (iii) traffic rules, (iv) previously scheduled traffic on the same railway network, and (v) criteria for optimization.

A railway network is basically composed of stations and one-way or two-way tracks. A dependency can be:

- *Station*: Place for trains to park, stop or pass through. Each station is associated with a unique station identifier. There are two or more tracks in a station where crossings.

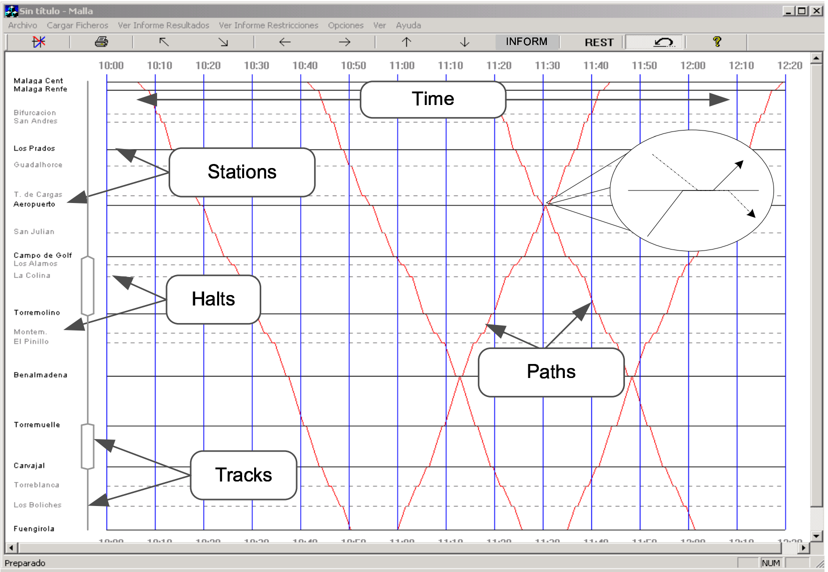


Figure 1: A sample of a running map

- *Halt*: Place for trains to stop, pass through, but not park. Each halt is associated with a unique halt identifier.
- *Junction*: Place where two different tracks fork. There is no stop time.

In Figure 1, horizontal dotted lines represent halts or junctions, while continuous lines represent stations. On a rail network, the user needs to schedule the paths of n trains going in one direction and m trains going in the opposite direction, trains of a given type and at a desired scheduling frequency.

The type of trains to be scheduled determines the time assigned for travel between two locations on the path. The path selected by the user for a train trip determines which stations are used and the stop time required at each station for commercial purposes. In order to perform crossing in a section with a one-way track, one of the trains should wait in a station. This is called a *technical stop*. One of the trains is detoured from the main track so that the other train can cross or continue. (Figure 2).

2.1 Railway Traffic Rules, topological and requirement constraints

A valid running map must satisfy and optimize the set of existing constraints in the periodic problem. Some of the main constraints to be considered are:

1. **Traffic rules** guarantee crossing operations. The main rules to take into account are:

   - *Crossing constraint*: Any two trains and going in opposite directions must not simultaneously use the same one-way track. The crossing of two trains can be performed only on two-way tracks and at stations, where one of the two trains has been detoured from the main track (Figure 2). Several crossings are shown in Figure 1.
   - *Expedition time constraint*. There exists a given time to put a detoured train back on the main track and exit from a station.

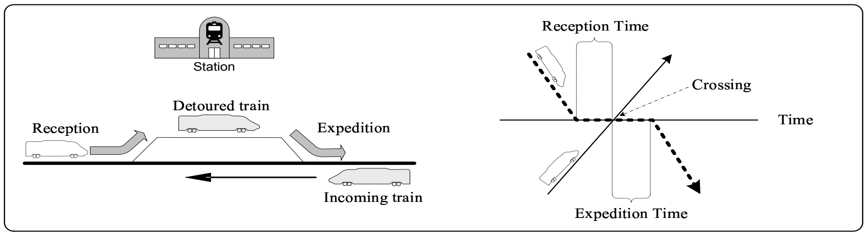


Figure 2: Constraints related to crossing in stations

- *Reception time constraint*. There exists a given time to detour a train from the main track so that crossing or overtaking can be performed.

2. **User Requirements**: The main constrains due to user requirements are:

   - *Type of train and Number of trains* going in each direction to be scheduled and *Travel time* between locations.
   - *Path of trains*: Locations used and *Stop time* for commercial purposed in each direction.
   - *Scheduling frequency*. The frequency requirements of the departure of trains in both directions. This constraint is very restrictive, because, when crossing is performed, trains must wait for a certain time interval at stations. This interval must be propagated to all trains going in the same direction in order to maintain the established scheduling frequency.

In accordance with user requirements, the system should obtain the best solutions available so that all constraints are satisfied. Several criteria can exist to qualify the optimality of solutions: minimize duration and/or number of technical stops, minimize the total time of train trips (span) of the total schedule, giving priority to certain trains, etc.

2.2 General System Architecture

The general outline of our system is presented in Figure 3. It shows several steps, some of which require the direct interaction with the human user to insert requirement parameters, parameterize the constraint solver for optimization, or modify a given schedule. First of all, the user should require the parameters of the railway network and the train type from the central database (Figure 3). This database stores the set of locations, lines, tracks, trains, etc. Normally, this information does not change, but authorized users may desire to change this information. With the data acquired from the database, the system generates the formal mathematical model. This model is composed by a large number of mixed-integer constraints. To translate the mixed-integer problem into a linear problem, a topological technique is carried out to assign value to each integer variable. This technique carries out a partition of the stations such that each block of stations represents a subproblem and a traffic pattern (solution) must be generated for each subproblem. This traffic pattern is generated based on the problem topology just as the number of stations, the train frequency, the type of stations, and mainly the distance among the stations. Once the traffic patterns are generated, the integer variables are instantiated and the linearized problem is straightforward solved returning the running map data. If the mathematical model is not feasible, the user must modify the parameters, mainly the most restrictive ones. If the running map is consistent, the graphic interface plots the scheduling. Afterwards, the user can graphically interact with the scheduling to modify the arrival or departure times. Each interaction is automatically checked by the constraint checker in order to guarantee the consistency of changes. The user can finally print out the scheduling, to obtain reports with the arrival and departure times of each train in each location, or graphically observe the complete scheduling topology.

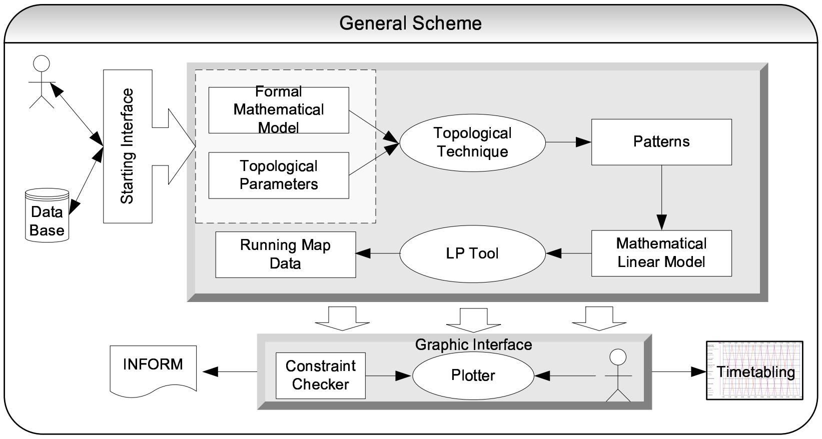


Figure 3: General scheme of our tool.

3 Topological Constraint Optimization Technique

The railway optimization problem is considered to be more complex than job-shop scheduling [8, 12]. Here, two trains, traveling in opposite directions use tracks between two locations for different durations, and these durations are causally dependent on how the scheduling itself is done (ie: order of tasks), due to the stopping, and starting time for trains in a non-required technical stop, expedition, reception times, etc. Some processes (detour from the main railway) may or may not be required for each train at each location. In our system, the problem is modeled as a COP, where finite domain variables represent frequency and arrival and departure times of trains of locations. Relations on these variables permit the management of all the constraints due to the user requirements, topological constraints, traffic rules, commercial stops, technical operations, maximum slacks, etc. Hundred of trains, of different types, in different directions, along paths of dozens of stations have to be coordinated. Thus, many variables, and many and very complex constraints arise. The problem turns into a mixed-integer programming problem, in which thousands of inequalities have to be satisfied and a high number of variables take only integer values. As is well known, this type of model is far more difficult to solve than linear programming models.

Our goal is focused on periodic train scheduling, where all the trains in the same direction are of the same type; they stop in the same stations; and no previously trains are scheduled. Therefore, our objective is to solve this problem previously assigning values to integer variables such that the mixed-integer programming problem is transformed into a linear programming problem. Then, the linearized problem is easily solved. In this way, the topological constraint optimization technique is committed to this goal.

The topological constraint optimization technique generates the traffic patterns based on several features as identification of bottlenecks, periodicity of running maps, number of stations, distance among stations, possible wide-paths for trains, etc.

3.1 Topological Technique

The main idea of this technique is to generate a traffic pattern for each set of stations such that the union of these contiguous traffic patterns determine the journey of each train. Figure 4 shows a possible set of stations (block).

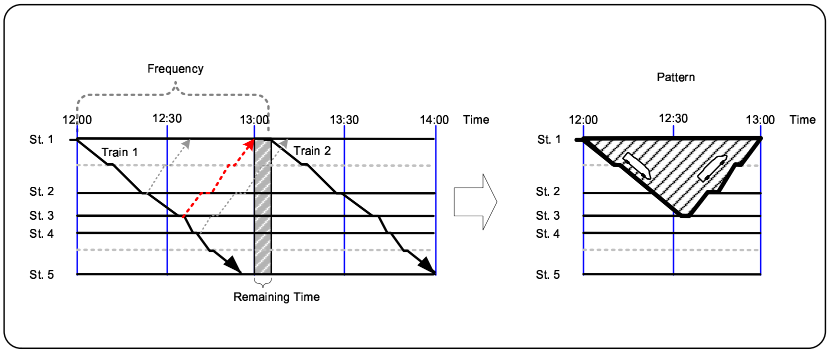


Figure 4: First traffic pattern generation.

The block of stations will be selected taking into account the speed of the trains, the distance among stations and the frequency inserted into the problem. Each traffic pattern covers the block of stations necessary for a train to go from the first station of the block to the last station of the block and return from the last station to the first one (round trip). This round trip must arrive to the first station (St.1) as close but before to the following train departure (Train 2) as possible. Thus, our objective is to minimize the remaining time between the frequency and the round trips. Each possible round trip will involve a different set of constraints. The round trip that minimize the remaining time will be selected as the *pattern*. This traffic pattern will be composed by a higher number of stations than the rest of possible round trips.

Once the first traffic pattern has been generated, we study the following pattern with the remaining stations. Figure 5 shows the generation of the second pattern using the same strategy.

Therefore, when the second traffic pattern is generated, the topological technique studies the following traffic pattern until there is no station left. In Figure 6, we can observe an example of running map with three complete traffic patterns and some stations without traffic pattern. However, it is usual that there are some stations left. These stations are not involved in any traffic pattern. We must take into account that the best traffic pattern in a block of stations implies to start the following block of stations in the last station of the previous block. We must check all traffic patterns together in order to obtain the journey. Moreover, the first combination of traffic patterns may not be the best solutions due to existence of some combinations of traffic patterns. This combination depends on the number of stations that are not involved in a traffic pattern. In this way, we explore all possible combinations in order to obtain the best set of traffic patterns.

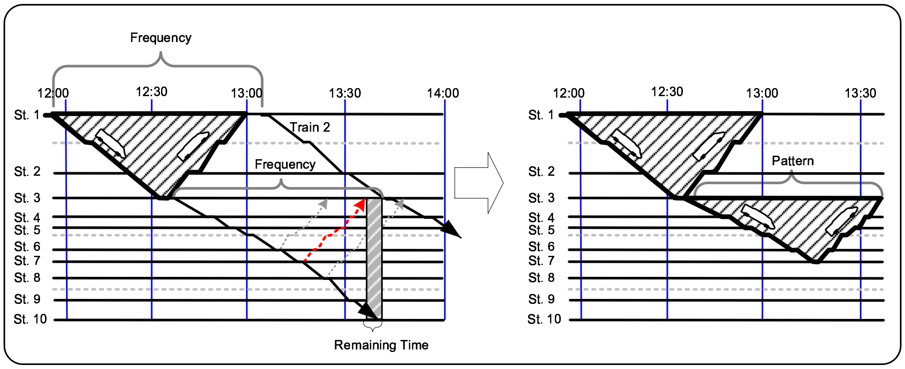


Figure 5: Second Pattern generation.

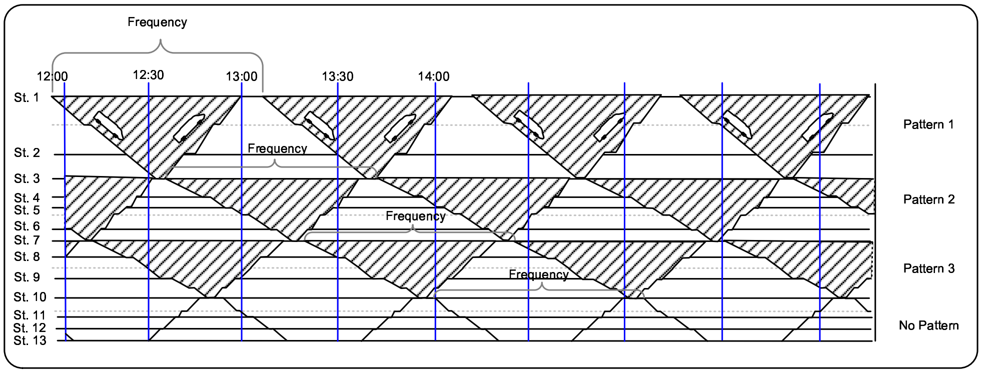


Figure 6: Periodic Pattern generation.

Figure 6 shows an example in which three stations are not involved in any traffic pattern. So, some combinations are possible and they are restricted to the set of stations involved in the first traffic pattern. Thus, these three stations can be sorted between the first and the last traffic pattern. In this way, the first traffic pattern may start at the second or third station and the last traffic pattern may finish in the penultimate or last but two station. However, due to efficient use of resources, or depending on the importance of the station, it is more appropriate the first traffic pattern (last traffic pattern) starts (finishes) at the first (last) station.

4 Evaluation

The application and performance of this system depends on several factors: Railway topology (locations, distances, tracks, etc.), number and type of trains (speeds, starting and stopping times, etc.), frequency ranges, initial departure interval times, etc.

In this section, we compare the performance of our topological technique with some well-known tools: LINGO as an Operational Research tool and ILOG Concert Technology (CPLEX) that combines techniques of constraint programming and mathematical programming. Both are appropriate tools for solving these types of problems. However, the system carried out important preprocessing heuristics [1] before executing these well-known tools in order to significantly reduce the size of these problems. Therefore, CPLEX and LINGO are combined with some heuristics, and they obtained the optimal solutions of their relaxed problems.

This empirical evaluation was carried out integrating both different types of problems: benchmark (real) problems and random problems. The computer used in our tests was a Pentium IV 2.8Mz with 512 Mb. of memory. Thus, we defined random instances over a real railway infrastructure that joins two important Spanish cities (La Coruña and Vigo). The journey between these two cities is currently divided by 40 dependencies between stations (23) and halts (17).

In our empirical evaluation, each set of random instances was defined by the 3-tuple $< n, s, f >$, where n was the number of trains in each direction, s the number of stations/halts and f the frequency. The problems were randomly generated by modifying these parameters. Thus, each of the tables shown sets two of the parameters and varies the other one in order to evaluate the algorithm performance when this parameter increases.

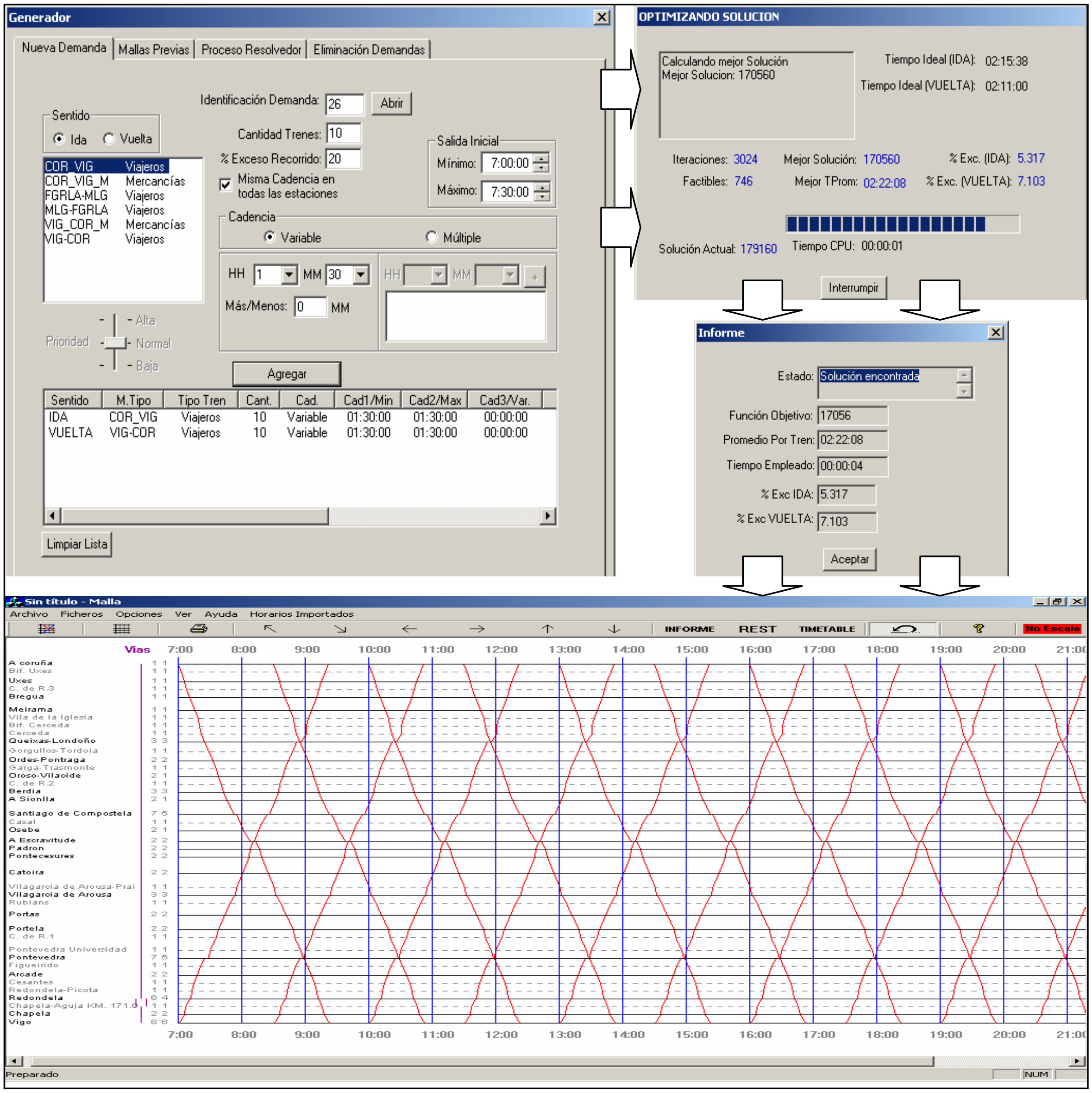
In Table 1, we present the running time in seconds and the journey time in problems where the number of trains was increased from 5 to 50 and the number of stations/halts and the frequency were set at 40 and 90, respectively: $< n, 40, 90 >$. The results shows that CPLEX obtained better running time and journey time than LINGO. However, it can be observed that the running time is lower using the topological technique than the other two COP tools. Furthermore, our technique always obtained the same journey time (lower than CPLEX and LINGO) due to the fact that it generates the corresponding traffic patterns and it is independent of the number of trains. Figure 7 shows the system interface executing our technique with the instance $< 10, 40, 90 >$. The first window shows the user parameters, the second window presents the best solution obtained in this moment, the third window presents data about the best solution found, and finally the last window show the obtained running map.

Table 1: Running time (sec.) and journey time in problems with different trains.

$< n, 40, 90 >$	**CPLEX+heuristics**		**LINGO+heuristics**		**TOPOLOGICAL**	
Trains	*running time*	journey time	*running time*	journey time	*running time*	journey time
5	5”	2:29:33	8”	2:30:54	3”	2:22:08
10	8”	2:26:04	17”	2:31:37	4”	2:22:08
15	13”	2:26:18	24”	2:31:51	5”	2:22:08
20	16”	2:26:25	35”	2:31:58	5”	2:22:08
50	55”	2:31:09	1302”	2:32:11	10”	2:22:08

Table 2 shows the running time in seconds and the journey time in problems where the number of stations was increased from 10 to 60 and the number of trains and the frequency were set at 10 and 90, respectively: $< 10, s, 90 >$. In this case, only stations were included to analyze the behavior of the techniques. It can be observed that the running time was lower using our technique in all instances. The journey time was also improved using our topological technique. It is important to realize the difference between the instance $< 10, 40, 90 >$ of the Table 1 and the instance $< 10, 40, 90 >$ of the Table 2. They represents the same instance, but in Table 2 we only used stations (no halts), so that the number of possible crossing between trains is much more larger. This item reduced the journey time from 2:22:08 to 2:20:22, but the number of combination increased the running time from 4” to 7”. Furthermore, CPLEX and LINGO maintained similar behaviors.

In Table 3, we present the running time in seconds and the journey time in problems where the frequency was increased from 60 to 140 and the number of trains and stations were set at 20 and 40, respectively: $< 20, 40, f >$. It can be observed that the frequency the topological



Generador

Nueva Demanda | Mallas Previas | Proceso Resolvedor | Eliminación Demandas

Identificación Demanda: 26 Abrir

Sentido: ● Ida ○ Vuelta

COR_VIG	Viajeros
COR_VIG_M	Mercancías
FGRLA-MLG	Viajeros
MLG-FGRLA	Viajeros
VIG_COR_M	Mercancías
VIG-COR	Viajeros

Cantidad Trenes: 10

% Exceso Recorrido: 20

☑ Misma Cadencia en todas las estaciones

Salida Inicial: Mínimo: 7:00:00 Máximo: 7:30:00

Cadencia: ● Variable ○ Múltiple

HH 1 MM 30 HH MM

Más/Menos: 0 MM

Prioridad: Alta, Normal, Baja

Agregar

Sentido	M.Tipo	Tipo Tren	Cant.	Cad.	Cad1/Min	Cad2/Max	Cad3/Var.
IDA	COR_VIG	Viajeros	10	Variable	01:30:00	01:30:00	00:00:00
VUELTA	VIG-COR	Viajeros	10	Variable	01:30:00	01:30:00	00:00:00

Limpiar Lista

OPTIMIZANDO SOLUCION

Calculando mejor Solución
Mejor Solucion: 170560

Tiempo Ideal (IDA): 02:15:38

Tiempo Ideal (VUELTA): 02:11:00

Iteraciones: 3024 Mejor Solución: 170560 % Exc. (IDA): 5.317

Factibles: 746 Mejor TProm: 02:22:08 % Exc. (VUELTA): 7.103

Solución Actual: 179160 Tiempo CPU: 00:00:01

Interrumpir

Informe

Estado: Solución encontrada

Función Objetivo: 17056

Promedio Por Tren: 02:22:08

Tiempo Empleado: 00:00:04

% Exc IDA: 5.317

% Exc VUELTA: 7.103

Aceptar

Sin título - Malla

Figure 7: System Interface.

Table 2: Running time (sec.) and journey time in problems with different number of stations.

$< 10, s, 90 >$	CPLEX+heuristics		LINGO+heuristics		TOPOLOGICAL	
Stations	*running time*	journey time	*running time*	journey time	*running time*	journey time
10	2”	0:58:36	4”	0:58:06	1”	0:57:36
20	3”	1:04:11	20”	1:04:11	2”	1:04:11
30	15”	1:45:08	42”	1:45:38	4”	1:45:08
40	56”	2:23:16	28”	2:24:36	7”	2:20:22
60	340”	3:44:28	326”	3:44:22	40”	3:32:15

technique improved the journey time when the frequency increased. As in previous results, the running time of the topological technique was lower than CPLEX and LINGO.

Table 3: Running time (sec.) and journey time in problems with different cadencies.

$< 20, 40, f >$	**CPLEX+heuristics**		**LINGO+heuristics**		**TOPOLOGICAL**	
Frequency	*running time*	journey time	*running time*	journey time	*running time*	journey time
60	$>$ 43200”	-	$>$ 43200”	-	36”	2:32:11
90	17”	2:26:25	32”	2:31:58	5”	2:22:08
100	18”	2:23:10	34”	2:22:55	3”	2:19:09
120	16”	2:16:17	27”	2:18:47	4”	2:16:00
140	17”	2:20:18	27”	2:16:19	4”	2:17:03

5 Conclusions

We have proposed a topological constraint optimization technique for solving periodic train scheduling in collaboration with the National Network of Spanish Railways (RENFE). This technique has been inserted into the system to solve more efficiently periodic timetables. This system, at a current stage of integration, supposes the application of methodologies of Artificial Intelligence in a problem of great interest and will assist railways managers in optimizing the use of railway infrastructures and will help them in the resolution of complex scheduling problems.

References

[1] Barber, F., Salido, M.A., Ingolotti, L., Abril, M., Lova, A., Tormos, P. *An Interactive Train Scheduling Tool for Solving and Plotting Running Maps*, Current Topics in Artificial Intelligence LNAI 3040, (2004), To appear.

[2] Bussiecky, M.R., Winter, T., Zimmermann, U.T. *Discrete optimization in public rail transport*, Mathematical Programming 79(3), (1997), 415-444.

[3] Bussiecky, M.R., Zimmermann, U.T. *Combinatorial Optimization Methods for Optimal Lines in Real-World Railway Systems*, Technical Report TR-95-03, (1996).

[4] Caprara, A., Fischetti, M., Toth, P. *Modeling and Solving the Train Timetabling Problem*, Research Report OR/00/9 DEIS, (2000).

[5] Caprara, A., Fischetti, M., Guida, P., Monaci, M., Sacco, G., Toth, P. *Solution of Real-World Train Timetabling Problems*, 34th Annual Hawaii International Conference on System Sciences **3** (2001).

[6] Chiu, C.K., Chou, C.M., Lee, J.H.M., Leung, H.F., and Leung, Y.W., *A Constraint-Based Interactive Train Rescheduling Tool*, Constraints **7** (2002), 167–198.

[7] Kaas, A.H., *Methods to Calculate Capacity of Railways*, Ph. Dissertation (1998).

[8] Kreuger, P.; Carlsson, M.; Olsson, J.; Sjoland, T.; Astrom, E. *The TUFF train scheduler: Two duration trip scheduling on multi directional track networks*. Workshop on Tools and Environments for Constraint Logic Programming, (1997).

[9] Lindner, T., *Train schedule optimization in public rail transport*, Ph. Dissertation, Technische Universitat Braunschweig, Germany, (2000).

[10] Oliveira, E., Smith, B.M., *A Job-Shop Scheduling Model for the Single-Track Railway Scheduling Problem*, Research Report 2000.21, University of Leeds, (2000).

[11] Salido, M.A., Barber, F., *A Constraint Ordering Heuristic for Scheduling Problem*, 1st Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA 2003), (2) 476-491, (2003).

[12] Zuidwijk, R.A., Kroon, L.G., *Integer Constraints for Train Series Connections*, Erasmus Research Institute of Management (ERIM), Discussion Paper. (2002).