Electronic Notes in Theoretical Computer Science 50 No. 1 (2001) – Proc. ATMOS 2001
URL: http://www.elsevier.nl/locate/entcs/volume50.html 15 pages

Heuristics for railway infrastructure saturation

Xavier Delorme [a,b,1], Joaquin Rodriguez [a,2] and Xavier Gandibleux [b,3]

[a] *ESTAS*
Institut National de Recherche sur les Transports et leur Sécurité
F-59650 Villeneuve d'Ascq, France

[b] *LAMIH – UMR CNRS 8530*
Université de Valenciennes
F-59313 Valenciennes cedex 9, France

Abstract

This research concerns the problem of the evaluation of the railway infrastructure capacity. It is an important question when railway authorities have to choose between different infrastructure investment projects. We developped independently two heuristic approaches to solve the infrastructure saturation problem. The first is based on a constraint programming model which is solved using a greedy heuristic. The second approach identifies the saturation problem as a unicost set packing problem and its resolution is ensured by an adaption of GRASP metaheuristic. Currently, both resolution techniques are not in competition. The goal is to grasp the resolution ability of the heuristics and to analyse the kind of solutions produced. The Pierrefitte-Gonesse junction has been used as experimental support. A software environment allows to simulate several timetables involving TGV, Inter City and Freight trains.

1 Introduction to the railway saturation problem

Basically, the capacity of a component of a rail system is defined as the maximum number of trains that can be operated on it within a certain unit of time u (e.g. an hour or a day). The theoretical expression of the capacity of a railway line in a given direction, noted C, can be defined as :

$$C = \frac{u}{h} \tag{1}$$

[1] Email: Xavier.Delorme@inrets.fr

[2] Email: Joaquin.Rodriguez@inrets.fr

[3] Email: Xavier.Gandibleux@univ-valenciennes.fr

where h is the minimum headway time between two successive trains. The minimum headway time depends on the signaling system installed on the line considered. Expressions which are more accurate can be used to include more features of the rail system (see [15]).

For a junction of lines, the previous analytical definition of the capacity does not apply. The capacity of a junction is not the sum of the capacity of the converging lines, therefore it is necessary to build models which are more complex [7]. The capacity of a junction can be defined as the solution for an optimization problem. The problem is to find the maximum number of trains among a predefined train set that can be operated on the junction, i.e. to find a saturation timetable. In this problem, it is assumed that trains do not stop during the run in the junction. To define more formally the railway saturation problem, we need to introduce the following notations.

Let $\mathcal{T}$ be the set of trains considered.

Let $\mathcal{R}$ be the set of routes used by trains running on the junction considered.

The function $fr : \mathcal{T} \rightarrow \mathcal{P}(\mathcal{R})$ gives for each train the feasible routes.

The function $fst : \mathcal{T} \rightarrow \mathcal{P}(\mathbb{N})$ gives for each train the feasible start time values.

The function $ra : \mathcal{T} \rightarrow \mathcal{R}$ defines the route assigned to a train.

The function $sta : \mathcal{T} \rightarrow \mathbb{N}$ defines the start time assigned to the train run on the junction.

Let $Inc \subseteq \mathcal{T} \times \mathcal{R} \times \mathbb{N} \times \mathcal{T} \times \mathcal{R} \times \mathbb{N}$ be the relation denoting which timetable assignements are conflicting.

Definition 1.1 *An instance of a railway saturation problem is a six-tuple* $(\mathcal{T}, \mathcal{R}, fr, fst, Inc, u)$*, the problem is to find a couple* (ra, sta) *so that:*

- $\forall t \in \mathcal{T}, ra(t) \in fr(t)$,
- $\forall t \in \mathcal{T}, sta(t) \in fst(t)$,
- $\forall (t, t') \in \mathcal{T} \times \mathcal{T}, (t, ra(t), sta(t), t', ra(t'), sta(t')) \notin Inc$,

and the objective function is to maximize the size of the set $\{t \in \mathcal{T}, sta(t) \leq u\}$.

Given a railway junction and the time interval u, an instance of the saturation problem is characterized by the $\mathcal{T}, fr$ and fst considered. Let us consider an example of the generation process of an instance problem, the instance I will be noted $(\mathcal{T}_I, \mathcal{R}, fr_I, fst_I, Inc_I, u)$.

Firstly, the set $\mathcal{T}_I$ can be constructed by using the equation 1 of the line capacity. Let $\mathcal{L}_I$ be a set of converging lines considered and the fonction $trl : \mathcal{L}_I \rightarrow \mathcal{P}(\mathcal{T})$ which provides the set of saturation trains running on each line. For each line, the saturation set must satisfy the following property :

$$|trl(i)| = C_i, \forall i \in \mathcal{L}_I \text{ where } C_i \text{ is calculated by the equation 1.}$$

The set $\mathcal{T}_I$ is then constructed from the sets $trl(i)$ by :

$$\mathcal{T}_I = \cup_{i \in \mathcal{L}_I} trl(i).$$

Secondly, the function fr_I has to be defined. For example, fr_I can give for each train $t \in trl(i)$ the set of all routes from the entry point to the exit point of the line i.

Finally, for the function fst_I the set of trains of a line is considered as a train sequence. Let $t_{i,j}$ be the j th train of the line i. The expression of the feasible start times is :

$$\forall i \in \mathcal{L}_{\mathcal{I}}, j = 1, \ldots, |trl(i)|, fst_I(t_{i,j}) = [(j-1) * h_i, H]. \quad (2)$$

where H is an arbitrarily large horizon value, h_i is the minimum headway time between two successive trains on a line i. The equation 3 of the section 2.1 gives an expression of this term depending on the specific features of the signaling system installed on the given line.

Any real case study of the railway saturation problem represents a large scale numerical instance. General optimization techniques may encounter difficulties to compute the optimal solution. The use of heuristics, aiming to find a suitable solution within a limited computing time, is pertinent in this context.

Next section presents the two models developed for railway junction saturation studies. The Pierrefitte-Gonesse junction and traffic scenarios elaborated are described in section 3. In accordance with the hypothesis of both models presented, numerical data are generated. Section 4 gives the main steps for the generation process. The resolution is heuristic. For the Constraint Programming model, a greedy algorithm using the ILOG libraries for the propagation mechanism has been elaborated. For the unicost set packing model, an adaptation of the GRASP metaheuristic has been designed. All details about the resolution methods are mentioned in section 5. The last section reports numerical experiments. A solution analysis is discussed and forthcoming investigations are underlined.

2 The formulation of models

2.1 Constraint Programming model

The aims of the Constraint Programming (CP) models were originally to solve feasibility problems : given a set $X_1, \ldots, X_n$ of variables, each associated with a domain $D_1, \ldots, D_n$ respectively, and a set of constraints C_1,...,C_n, i.e. a subsets of $D_1 \times \ldots \times D_n$, find an assignment of values to the variables while simultaneously satisfying the constraints. The CP models were extended to solve optimization problems : when a feasible solution is obtained, the value of the objective function is a new upper (resp. lower) bound of a variable, representing the objective function to minimize (resp. to maximize). This restriction is made by posting a new constraint on this variable.

In [14], we have presented a CP model of a real-time train scheduling problem. This formulation has been applied to the case study of the Pierrefitte-Gonesse junction. First, we will recall the main components of a signalling system, then the formulation of the CP model. Finally, we will present how this CP model has been transformed to tackle the train saturation problem defined in section 1.

2.1.1 Components of a signalling system

The main components of a signalling system are the *track circuits*, the *signals* and the *blocks*. A track circuit is an electrical circuit of which the rails of the track form a part. It detects without fail the presence of trains on a particular track section. To avoid train collision between following trains, the *signals* placed along the tracks provide the drivers with information about maximum authorized speed thanks to colored lights. The information of a signal applies on a line section named a *block*. A block may consist of one or more track circuits. For an automatic block signaling system featuring n_a colours used for light signals (also named n_a-aspect signals), the headway can be defined by :

$$h = \frac{(n_a l_b + l_t)}{v} \tag{3}$$

where l_b is the length of a block, l_t is the train length and v is the average speed.

2.1.2 Real-time train scheduling problem

The CP model presented in [14] focuses on expressing with *explicit terms* the influence of the signalling system on the run of the trains. This feature is important for coping correctly with problems within heavy traffic conditions. The run of a train through a junction is a sequence of elementary runs. Each elementary run is the run through a track circuit. An elementary run is considered as an *activity* and each track circuit as the *unary resource* required to process it. Using the notation of section 1, a run of a train $t \in \mathcal{T}$ is a *sequence of* n_t *activities*. In the CP model, $ra(t)$ is the variable of the route assignement of a train and $fr(t)$ is the domain associated with it. Each variable $ra(t)$ is linked to a set of track circuit assignement variables noted $tca_t(i)$, i being the index in the sequence of activities. The domains associated to $tca_t(i)$ are noted $ftc_t(i)$. These domains are deduced from $fr(t)$. A resource constraint links each activity i with all the alternative resources $ftc_t(i)$. As not all alternative routes can have the same number of track circuits, we have created a fake track circuit to ensure that our model is declarative. The fake track circuit is added to the track circuit sequence to obtain sequences of the same size for all alternative routes. Let $|r|$ be the notation which gives the number of track circuits for a route $r \in \mathcal{R}$. The value of n_t is defined by:

$$n_t = \max_{r \in fr(t)} |r|.$$

After the definition of the number of activities n_t, let us consider the definition of the capacity constraints of the resources. Let $st_t(i), ct_t(i), pt_t(i)$ be respectively the start time, completion time, and processing time variables of the activity associated to the elementary run of index i. The capacity constraint that restricts the use of each track circuit to only one activity at a time is :

$$\begin{aligned} &\forall t, t' \in \mathcal{T}, \forall i \in [1, n_t], \forall j \in [1, na_{t'}] \\ &\qquad tca_t(i) \neq tca_{t'}(j) \Rightarrow (ct_t(i) \leq (st_{t'}(j)) \vee ((ct_{t'}(j) \leq st_t(i)) \end{aligned} \tag{4}$$

i.e. unless two activities use different ressources, they cannot overlap.

We consider now the definition of the temporal constraints. Due to a clearing phase, the time windows of successive activities overlap each other, i.e. during that time the train occupies two contiguous track circuits (e.g. see the black rectangles of the Gantt chart in Figure 1). If we consider a block signalling system with 2 aspects, the start of each activity has to be synchronised with the start of the activity corresponding to the first track circuit of the current block. For the general case of a block system with n aspects, the synchronisation is established with the entrance in the first track circuit of the $n-2$ previous block (e.g. see dashed rectangles for $n = 3$ in Figure 1). Let $run_t(i), clr_t(i)$ be the variables for the minimum duration for these two phases, let $ftb_t(i)$ be the variable for the index of the first track circuit of the block. The start time of the running phase of an activity of index i is equal to $ct_t(i-1) - clr_t(i-1)$.

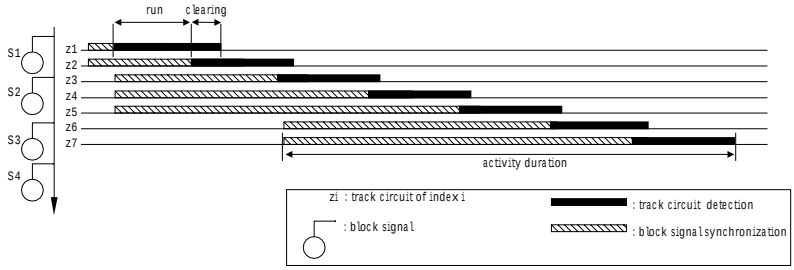

Fig. 1. Gantt chart of activities modelling a 3 aspect block signalling system

The temporal constraints are :

$$pt_t(i) \geq run_t(i) + clr_t(i) \tag{5}$$

$$ct_t(i-1) - st_t(i) \geq clr_t(i-1) \tag{6}$$

$$ct_t(i) - ct_t(i-1) \geq run_t(i) + clr_t(i) - clr_t(i-1) \tag{7}$$

$$\max_{j \in ftb_t(i)} (ct_t(j-1) - clr_t(j-1)) \geq st_t(i) \geq \min_{j \in ftb_t(i)} (ct_t(j-1) - clr_t(j-1)) \tag{8}$$

2.1.3 Saturation problem

The previous model was designed to solve real time train management problems. We will now present how this model has been transformed for the saturation problem. From the definition of section 1, the decision variable $sta(t)$ is replaced by variables $st_t(i = 1)$. In real time problems, trains can be delayed during the run. Conversaly, in a saturation problem, an extra constraint is added which ensures no delay to the movement of the trains through the junction :

$$ct_t(i) - ct_t(i-1) = run_t(i) + clr_t(i) - clr_t(i-1) \tag{9}$$

This constraint subsumes the previous constraint (7). The constraints 4 to 9 of the CP model enable to avoid a complete enumeration of the relation Inc.

2.2 Unicost Set Packing Problem model

This model is inspired by [16,17] which propose a Node Packing Problem formulation for the feasibility problem. The formulation we considered is a well-known

problem of combinatorial optimization, the Unicost Set Packing Problem (USPP) [11] and permits also to solve the feasibility problem.

2.2.1 Basic model

For this model, we need a function $at : \mathcal{T} \rightarrow \mathbb{N}$ that gives for each train its theoretical arrival time. Given a six-tuple $(\mathcal{T}, \mathcal{R}, fr, fst, Inc, u)$ and considering only one start time value for each train which is its theoretical arrival time in the node ($fst(t) = \{at(t)\}$), we define a binary variable $x_{t,r}$. This variable is equal to 1 if the train t uses the route r (i.e. $r = ra(t)$) and equal to 0 otherwise. These variable values are limited by two sets of constraints :

- a train can only use one route :

$$\sum_{r \in fr(t)} x_{t,r} \leq 1, \forall t \in \mathcal{T}$$

- the assignements of variable values that correspond to a conflicting timetable are impossible :

$$x_{t,r} + \sum_{r' \in fr(t'),((t,r),(t',r')) \in Inc} x_{t',r'} \leq 1, \forall (t,t') \in \mathcal{T}^2, r \in fr(t)$$

The objective of this problem is to maximize the number of variables $x_{t,r}$ set to 1. As mentionned, this model can also be used for the feasibility problem : a problem is feasible if this number is equal to the number of trains considered ($\sum_{t \in \mathcal{T}} \sum_{r \in fr(t)} x_{t,r} = |\mathcal{T}|$).

2.2.2 Completed model

This model can be completed if we need some start time values for at least one train. In this case, we consider a function $\Delta : \mathcal{T} \rightarrow \mathcal{P}(\mathbb{Z})$ which gives for each train the set of possible time-deviations δ. These time-deviations enable to move forward or delay from the theoretical arrival time of the train. So they define the set of feasible start time values ($fst(t) = \{at(t) + \delta, \forall \delta \in \Delta(t)\}$). So, binary variables are $x_{t,r,\delta}$ ($x_{t,r,\delta} = 1$ if $r = ra(t)$ and $at(t) + \delta = sta(t)$) and we obtain the following formulation (10) :

$$(10) \quad \left[\begin{array}{ll} Max\ z = \displaystyle\sum_{t \in T} \sum_{r \in fr(t)} \sum_{\delta \in \Delta(t)} x_{t,r,\delta} & \\ \displaystyle\sum_{r \in fr(t)} \sum_{\delta \in \Delta(t)} x_{t,r,\delta} \leq 1 & , \forall t \in \mathcal{T} \\ x_{t,r,\delta} + \displaystyle\sum_{\substack{r' \in fr(t'), \delta' \in \Delta(t'), \\ ((t,r,\delta),(t',r',\delta')) \in Inc}} x_{t',r',\delta'} \leq 1 & , \forall (t,t') \in \mathcal{T}^2, \\ & \quad r \in fr(t), \delta \in \Delta(t) \\ x_{t,r,\delta} \in \{0,1\} & , \forall t \in \mathcal{T}, r \in fr(t), \delta \in \Delta(t) \end{array} \right]$$

This formulation is more suitable to express our saturation problem. The instance S noted $(\mathcal{T}_S, \mathcal{R},\ fr_S, fst_S, Inc_S, u)$ in which $\mathcal{T}_S$ and fr_S are constructed respectively as $\mathcal{T}_I$ and fr_I defined in section 1 and fst_S is defined as follows :

$\forall i \in \mathcal{L}_S, j = 1, \ldots, |trl(i)|, fst_S(t_{i,j}) = \{at(t_{i,j}) + \delta, \forall \delta \in \Delta(t_{i,j})\} = [(j-1) * h_i, j * h_i[$

However, this instance is characterized by a huge number of variables and constraints. So, in practice we will not consider all the time-deviations in order to keep the problem within a reasonable size.

3 Junction and traffic analyse

3.1 Infrastructure considered

In this paper, we have considered the Pierrefitte-Gonesse node (Figure 2) which is located north of Paris. We noticed three main kinds of trains which travel through this node in both directions :

- TGV between Paris and the High Speed Line (HSL)
- Inter City trains between Paris and Chantilly
- Freight trains between Chantilly and the Grande Ceinture which cut-across the TGV routes

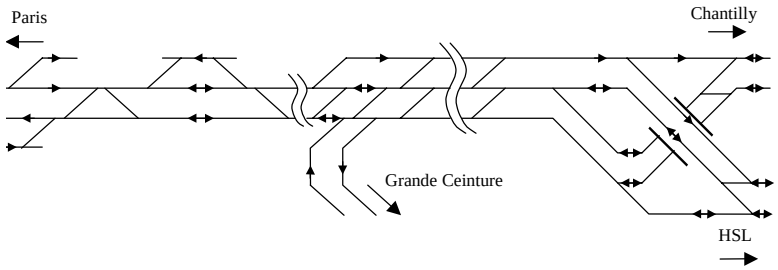

Fig. 2. Railway track map of Pierrefitte-Gonesse node

3.2 Tested scenario

Four relevant scenarios have been tested on this node :

- all kinds of train
- TGV and Inter City trains
- TGV and Freight trains
- Inter City and Freight trains

For the CP model, we have generated four instances, one for each scenario. As mentioned in section 1, an instance of a saturation problem is generated by specifying the three parameters $\mathcal{T}, fr, fst$. This generation is summerized in 3 steps :

- Step 1 : the capacity of each line of each scenario has been evaluated using the expression 1. The table 1 gives the result of this expression and the number of variables and constraints of the CP model instances.
- Step 2 : we have to define the function fr of the feasible routes. In previous experiments [13] of the CP model, we show that a complete search for solving instances with more than 12 trains, can not be carried out within reasonable time (i.e. less than one hour). We gave up the idea of carrying out a complete search therefore, we have restricted the set of feasible routes to one route, i.e. by setting the route variable to the "usual" route for each train category.
- Step 3 consists in defining the feasible start time values. To do this we used equation 2 of section 1.

For the USPP instances, the sets $\mathcal{T}$ are the same as those of the CP instances. The function fr considers all the routes from the entry point to the exit of the trains.

We used two definitions of the feasible start time function. Each definition depends on the expression of the parameter h_i i.e. the minimum headway time between two successive trains on a line i. The first value of h_i came from equation 3 of section 2.1. The second one noted h'_i considers values rounded to multiple values of a time-deviation granularity. For a time-deviation granularity of 30 seconds, the expression of h'_i is:

$$h'_i = \lfloor \frac{h_i}{30} \rfloor * 30$$

The expressions of the theoretical arrival time introduced in section 2.2 are :

$$at(t) = (j-1) * h_i \quad (resp.\ (j-1) * h'_i)$$

and the expressions of the time-deviations of a train j of a line i are :

$$\Delta(t_{i,j}) = \{30 * k\}, k \in [0, \ldots, \frac{h_i}{30}] \cap \mathbb{N} \quad (resp.\ \frac{h'_i}{30}\ )$$

Due to the two definitions of the minimal headway time, we generated 8 problem instances (Table 2). The instances N° 1-3-5-7 correspond to h'_i and the instances N° 2-4-6-8 correspond to h_i.

4 Numerical data generation

4.1 CP model

Figure 3 shows the process for generating data for the model. The model presented on section 2.1 takes input data from the SNCF railway simulator SYSIFE [10]. The simulator gives accurate data for the duration of run and clearing phases through track circuits. The simulation is done for each train category separatly through each possible route. A second input data set is the description of the infrastructure and the signaling system, this data set is shared by the simulation model and the CP model. Finally, a third input data set is the ordered set of trains considered

N°	$\mathcal{T}$			Numerical instances	
	TGV	IC	Freight	Variables	Constraints
1	81	76	49	75,307	77,142
2	81	76	0	64,422	66,117
3	81	0	49	44,185	45,543
4	0	76	49	42,099	43,524

Table 1
Instance characteristics for the CP model

N°	$\mathcal{T}$			Numerical instances		
	TGV	IC	Freight	Variables	Constraints	Density
1	100	100	50	3,720	53,489	0.26%
2	81	76	49	4,198	54,651	0.28%
3	100	100	0	2,880	33,767	0.36%
4	81	76	0	3,210	31,692	0.41%
5	100	0	50	2,160	17,354	0.43%
6	81	0	49	2,503	19,460	0.43%
7	0	100	50	2,400	21,794	0.40%
8	0	76	49	2,683	22,441	0.42%

Table 2
Instance characteristics for the USPP model

for the saturation problem instance whose generation has been described in section 3.2.

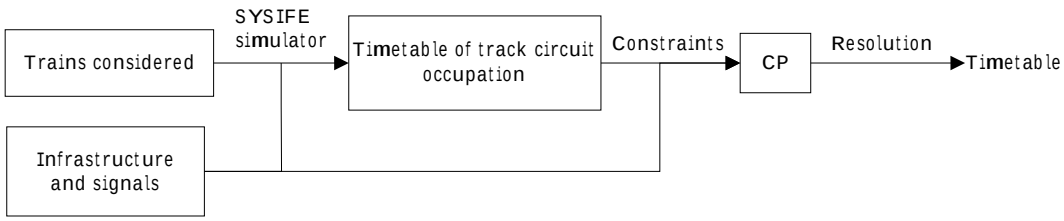

Fig. 3. The resolution process for CP

4.2 USPP model

We have obtained the use of ressources for each possible route for each train type considered, by using the simulation once only. Also, we determined the set Inc of incompatibilities for each scenario considered. These conflicts enable us to produce the constraints of the combinatorial problem to solve (Figure 4). Problem sizes are reported in Table 2, where column density correspond to the density of non-zero elements in coupling matrix between variables and constraints.

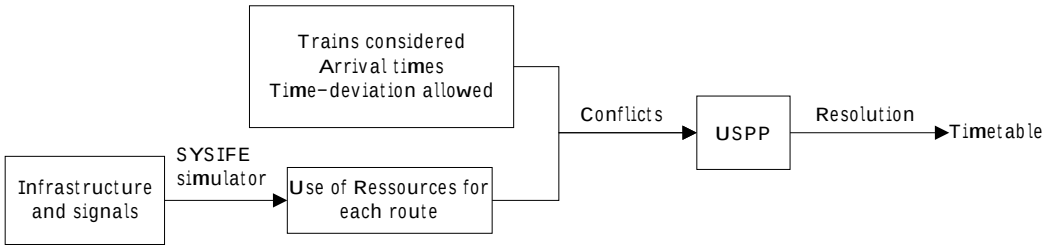

Fig. 4. The resolution process for USPP

5 Resolution methods

5.1 Constraint Programming

A CP formulation of a problem may be addressed by two categories of backtracking search. The first category is named "retrospective algorithms". It includes naive backtrack, backjumping [6]. The other category named "prospective algorithms" includes forward checking [8], which looks ahead to compute some form of local consistency among non instanciated variables. The CP has been extensively studied to develop various consistency algorithms, also named constraint propagation. A consistency algorithm makes it possible to reduce the domains of variables by removing values which are inconsistent with the constraints. For example, the Ilog Scheduler library [9] provides three mechanisms to propagate the resource utilisation constraint to adjust the time bounds of activities : timetable, disjunctive constraint and "edge finding".

As mentioned in section 3.2, in previous experiments [13] of the CP model, we showed that a complete search with a prospective algorithm for solving instances with more than 12 trains, can not be carried out within reasonable time. We gave up the idea of carrying out a complete search and we assumed that the route variables $ra(t)$ are set to the "usual" route (c.f. section 3.2). We developped the greedy algorithm 1 which uses the constraint propagation algorithms available in Ilog Solver/Scheduler libraries after each decision step on the $sta(t)$ variables. In this algorithm, we used the following notations :

- $propagate(Trains)$: a function which propagates the constraints posted for a set of trains $Trains$.
- $\prec$: an order relation so that two successive trains of a same converging line are separated by one train from all other lines .

The algorithm aims at scheduling all trains as early as possible. At each step of the loop, two criteria are used according a lexicographic order for choosing the train to schedule. The first one uses the earliest start time of the trains. If the first criterion is not sufficient to get only one train, the second criterion uses the order relation $\prec$.

$pendingTrains \leftarrow \mathcal{T}$
while ($pendingTrains \neq \emptyset$) **loop**
    $candidateTrains \leftarrow \{t \in pendingTrains$ with minimum earliest $sta(t)\}$
    $t = min_{\prec}(candidateTrains)$
    $sta(t) \leftarrow$ earliest $sta(t)$ value
    $pendingTrains \leftarrow pendingTrains \setminus \{t\}$
    $propagate(pendingTrains)$
endWhile

Algorithm 1. The greedy saturation-CP algorithm

5.2 Greedy Randomized Adaptative Search Procedure for USPP

Due to the important size of considered instances, we used an adaptation of the metaheuristic GRASP (Greedy Randomized Adaptative Search Procedure). This is a multistart two-phase metaheuristic for combinatorial optimization proposed by Feo and Resende [4]. First, a construction phase builds an initial solution with a greedy randomized procedure. This random character enables to obtain solutions in different areas of admissible solution space. Second, a local search phase improves these solutions. This process is repeated many times in order to compensate the random character of the greedy phase. Several new components extend the original GRASP method. They are presented and discussed in [12].

It is easy to customized this metaheuristic on any problems for which construction and local search algorithms are available. GRASP has been applied to a wide range of optimization problems. These include academic and industrial problems in scheduling, routing, logic, partitioning, location and layout, graph theory, assignment, manufacturing, transportation, telecommunications, electrical power systems, and VLSI design. An extensive anotated bibliography is available (see [5]).

The method produces good quality solutions for hard combinatorial optimization problems, particularly for the set covering and the set packing problems [2,3].

In the following, I denotes the set of variables, J the constraints and $t_{i,j}$ the coupling matrix between the variables ($i \in I$) and the constraints ($j \in J$). Our construction procedure (Algorithm 2) builds a solution from a trivial admissible solution ($x_i = 0, \forall i \in I$). Some variable values are changed (ie fixed to 1), keeping an admissible solution. The changes concern only one variable for one iteration. In order to increase the objective function, the variables which concern a minimum number of constraints and with a maximum value are prioritized, but the choice is random among the most interesting variables. Changes stop when we can not fix

a variable to 1 without the solution becoming non-admissible.

$x_i \leftarrow 0, \forall i \in I$
$Eval_i \leftarrow \sum_{j \in J} t_{i,j}, \forall i \in I$
$CL \leftarrow I$
while $(CL \neq \emptyset)$ **loop**
    $Limit \leftarrow (2 - \alpha) * \min_{i \in CL}(Eval_i)$
    $RCL \leftarrow \{i \in CL, Eval_i \leq Limit\}$
    $i^* \leftarrow RandomSelect(RCL)$
    $x_{i^*} \leftarrow 1$
    $CL \leftarrow CL \setminus \{i \in CL, \exists j \in J, t_{i,j} + t_{i^*,j} > 1\}$
endWhile

Algorithm 2. The construction phase algorithm

The neighbourhood used for our local search procedure is $k - p$ exchanges. A $k - p$ exchange consists in setting to 0 of k variables and to 1 of p others variables. Due to the combinatorial explosion of the number of exchange possibilities when k and p increase, we are obliged to limit them. So we have only tested $1 - 2$ exchanges. We have only accepted exchanges increasing the objective function. When an exchange is accepted, all exchange possibilities are tested again. Local search stops when there is no more exchange possible.

The parameter tuning is minimal, for our experiments we considered three different values for alpha (0.85 ; 0.9 ; 0.95) and we generated 60 solutions (20 solutions per alpha value).

6 Computational results

In this section, we present the computational results obtained with our two resolution methods (see Tables 3 and 4) for the four scenarios (see section 3.2). We remain that both resolution technics are not in competition. Thus there is no sense to do a comparison of CPU time. These results are obtained on UltraSparc with 143 MHz for CP and on a Pentium with 600 MHz for GRASP within reasonable times (between 1,000 seconds and 10,000 seconds). For information, results obtained by Cplex 6.0 [1] (LP and best IP value) on USPP instances are also indicated.

First of all, we can observe that the two algorithms produce "symetrical" quality solution performances for each scenario tested. The CP model highlights solutions with better performances on scenarios TGV/IC/FR and IC/FR and the USPP model shows better performances on scenarios TGV/IC and TGV/FR.

These results raise two preliminary assumptions. Firstly, to save capacity with the combination of Freight and TGV categories, it is necesary to consider alternative routes. Secondly, to combine Inter City trains and the other categories, the main role is given to the start variable in comparison with the route variable.

The scenario TGV/IC/FR supports the first assumption. The CP model has kept a balance between train categories, conversely the USPP model has discarded the Inter City trains.

Trains classes	TGV	IC	Freight	Total
TGV/IC/FR	35	46	12	93
TGV/IC	49	48	0	97
TGV/FR	81	0	0	81
IC/FR	0	75	19	94

Table 3
Computational results with CP

Train Classes	N°	Cplex		GRASP			
		LP	Best IP	TGV	IC	Freight	Total
TGV/IC/FR	1	231.9097	43	72	2	7	81
	2	199.9987	52	75	0	22	97
TGV/IC	3	184.6924	51	69	11	0	80
	4	151.6229	54	68	16	0	84
TGV/FR	5	145.9178	54	64	0	20	84
	6	130.0000	-	69	0	26	95
IC/FR	7	142.4778	46	0	58	14	72
	8	123.3292	48	0	65	17	82

Table 4
Computational results with GRASP

The scenarios TGV/IC and IC/FR support the second assumption. The best results have been obtained with the CP model. It could be explained by suitable choices on start time for Inter City and TGV trains. With these train category combinations, the search on route alternatives has a low impact on the number of trains. This can provide an explanation for the weak results of the USPP model.

The first assumption is also supported by the results of the scenario TGV/FR. The USPP model has the best number of trains with an important effort on the choice of routes. As the CP model does not provide the choice of routes, all the Freight trains are discarded. The set routes are incompatible, therefore the scheduling of the TGV postpones the earliest start time of the Freight trains to after the next possible scheduling of the TGV.

To summerize these experiments, the CP model is more efficient in finding good scheduling. On the other hand the USPP model is successful when the scenario needs to search good routes. The results obtained encourage us to take advantage of the complementary strengths of the two models into a hybrid method. The first

track is that the USPP model provides a ratio of good routes for initializing $\mathcal{T}$, the set of trains. The second track is that the CP model provides the good start time succession to the USPP model.

Acknowledgement

The authors wishes to thank the department "Direction de la Recherche" of SNCF for their comments and for supplying data used in the experiments.

References

[1] *Using the cplex callable library (manuel), version 4.0, cplex optimization, 1995.*

[2] Delorme, X., "Optimisation combinatoire et problèmes de capacité d'infrastructure ferroviaire," Mémoire de DEA, Université de Valenciennes et du Hainaut Cambrésis, Valenciennes, France (2000).

[3] Delorme, X., X. Gandibleux and J. Rodriguez, *Grasp for set packing problems*, Technical Report RT-01-704-FR, INRETS, France (2001).

[4] Féo, T. A. and M. Resende, *A probabilistic heuristic for a computationally difficult set covering problem*, Operations Research Letters **8** (1989), pp. 67–71.

[5] Festa, P. and M. Resende, *Grasp : an annoted bibliography*, in: P. Hansen and C. Ribeiro, editors, *Essays and surveys on metaheuristics*, Kluwer academic publishers, 2001 .

[6] Gaschnig, J., *A general backtracking algorithm that eliminates most redundant tests*, in: *International Joint Conference on Artificial Intelligence*, Cambridge MA, 1977, pp. 457–466.

[7] Hachemane, P., "Évaluation de la capacité de réseaux ferroviaires," Thèse 1632, École Polytechnique Fédérale de Lausanne (1997).

[8] Haralick, R. and G. Elliot, *Increasing tree search efficiency for constraint satisfaction problems*, Artificial Intelligence **14** (1980), pp. 263–313.

[9] Le Pape, C., *Three mechanisms for managing resource constraints in a library for constraint based scheduling*, in: *INRIA/IEEE Conference on Emerging Technologies and Factory Automation*, Paris-France, 1995.

[10] Moulin, R. and D. Gauyacq, *Simulation de systèmes ferroviaires - un projet décisif*, Revue d'Électricité et d'Électronique **7** (1997), pp. 39–46.

[11] Nemhauser, G. L. and L. A. Wolsey, "Integer and combinatorial optimization," Willey-Interscience, 1988, 763 p.

[12] Pitsoulis, L. and M. Resende, *Greedy randomized adaptive search procedures*, in: P. Pardalos and M. Resende, editors, *Handbook of Applied Optimization*, Oxford University Press, 2001 .

[13] Rodriguez, J., *Fluidification de noeuds ferroviaires (projet GRRT prospectif)*, Technical Report RR-00-724-FR, INRETS, France (2000).

[14] Rodriguez, J. and K. Lyes, *Constraint programming for real-time train circulation management problem in railway nodes*, in: *Computers in Railways VI*, Lisbonne-Portugal, 1998, pp. 597–606.

[15] U.I.C., *Fiche 405r*, Technical report, UIC (1978).

[16] Zwaneveld, P. J., L. G. Kroon, H. E. Romeijn, M. Salomon, S. Dauzère-Pérès, S. P. Van Hoesel and H. W. Ambergen, *Routing trains through railway stations : Model formulation and algorithms*, Transportation Science **30** (1996), pp. 181–194.

[17] Zwaneveld, P. J., L. G. Kroon and S. P. Van Hoesel, *Routing trains through railway a station based on a node packing model*, European Journal of Operational Research **128** (2001), pp. 14–33.