Electronic Notes in Theoretical Computer Science 66 No. 6 (2002)
URL: http://www.elsevier.nl/locate/entcs/volume66.html 11 pages

Solving the Freight Car Flow Problem to Optimality

Ricardo Fukasawa [a,1,2], Marcus Vinicius Poggi de Aragão [b,3], Oscar Porto [a,4] and Eduardo Uchoa [c,5]

[a] *Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), R. Marquês de S. Vicente, 225 , Rio de Janeiro, Brasil*

[b] *Departamento de Informática, PUC-Rio, Rio de Janeiro, Brasil*

[c] *Axioma Optimization. R. Jardim Botânico, 674/614, Rio de Janeiro, Brasil*

Abstract

A pervasive problem in freight railroad operations is to determine a feasible flow of cars to meet the required demands within a certain period of time. In this work we present a method to determine an optimal flow of loaded and empty cars in order to maximize profits, revenue or tonnage transported, given the schedule of the trains, together with their traction capacities. We propose an integer multicommodity flow model for the problem whose linear relaxation leads to very good upper bounds — at the cost of using a very large number of variables and constraints. In order to turn this model into a practical tool, we apply a preprocessing phase that may reduce its size by two or three orders of magnitude. The reduced model can then be solved by standard integer program packages with little, if any, branching effort. Computational results on real instances of the largest Latin American railroad freight company are reported. The product that resulted from this research is already in use at that company.

keywords: integer programming, network flows, railroad logistics

1 Introduction

Due to the high complexity of rail transportation network systems, a huge amount of joint decisions have to be taken periodically in order to achieve a proper and efficient operation. Since such decisions involve so many different

[1] Work supported by CNPQ
[2] Email: fukasawa@ele.puc-rio.br
[3] Email: poggi@inf.puc-rio.br
[4] Email: oscar@ele.puc-rio.br
[5] Email: euchoa@axiomaopt.com.br

aspects and data, the only reasonable approach is to break the operation problem into a number of subproblems of manageable size and try to solve then separately. Several studies of such subproblems, either in passenger or freight railroads, are available in the literature. We refer the reader to a comprehensive survey by Cordeau et al. [4].

In this paper, we address an operational freight railroad network problem, namely the *freight car flow problem*, which was presented to us by the operator of the largest freight railroad in Latin America. The problem consists in determining the full route of each car in the railroad network, as well as its load/unload sequence, within a time period of several days. Of course, empty or loaded cars can only move between locations when attached to a train. In the freight car flow problem, we assume that train schedules and train capacities are fixed in advance. The objective is, for a given period of time, to choose the demands to be met, totally or partially, in order to maximize (typically) total profit. The importance of this problem lies in the fact that if the car flow is decided "shortsightedly", not taking into account the forecast of demands for the following weeks, one may quickly fall into a very undesirable situation, in which there are no empty cars available or no capacities in the trains, to satisfy future demands.

We propose an integer multicommodity flow model for that problem. Such model has a very large number of variables and constraints, but its linear relaxation gives very good bounds. We devised a simple preprocessing procedure that turned out to be essential to reduce this model to a tractable size. This approach has already been turned into a product and is currently used to support the operations decisions at an important Brazilian logistics operator. The instances solved usually involve a 7-day time period, 150 railroad yards, and about 350 demands, 1700 train legs and 12,000 cars (divided into 25 distinct types). Such instances are usually solved to optimality in a few hours.

A similar model was already used by Holmberg et al. [5] to find an optimal flow of empty cars for the Swedish national railroad, by supposing that the flow of loaded cars is already determined and using the remaining capacities in the trains to move the empty cars. The drawbacks of that approach are: (i) it leaves the complex problem of finding a good flow of loaded cars to be solved elsewhere, and (ii) when a demand for empty cars can not be met, the assumed flow of loaded cars is not feasible anymore. Therefore, several rounds may be necessary to find a feasible flow of both empty and loaded cars. Of course, that solution is not likely to be optimal.

The outline of this paper is the following. Section 2 gives a detailed description of the problem. Section 3 presents our mathematical model, while Section 4 presents the preprocessing algorithms. Section 5 gives computational results. Final comments appear at the last section.

2 Problem description

The freight car flow problem seeks for a feasible flow of cars and a corresponding loading/unloading sequence in order to meet the demands of the clients as closely as possible. A feasible flow depends on the initial state of the network and must respect the train and yard capacities. Now we detail the problem by explaining some keywords:

Cars - The cars are divided into *types* according to its main characteristics. For instance, there are grain cars, fuel cars, container cars, etc. In many cases, a single commodity can be loaded into more than one type of car. The *initial state* of the network comprises the information of how many cars of each type are parked at each yard at the initial time, how many cars of each type are attached to trains already running at the initial time and whether such cars are empty or loaded with commodities from some demand. It is also possible to define *final states*, lower and upper bounds on the number of cars of each type that should be parked at each yard at the final time.

Yards - The yards are the locations of the railroad network where a car can be parked. Some yards are equipped with load/unload facilities. Some yards are only locations where an empty or loaded car can be left, waiting to be taken by another train. The *capacity* of a yard is the maximum number of cars that can be parked in it. For each yard there is a *handling time*, which is an upper bound on the time that takes to classify, maneuver, and possibly assemble cars into blocks, so they are ready to be attached to a train or loaded/unloaded.

Trains - The railroad has a fixed schedule of trains, defined as a set of journeys between successive yards (*train legs*) by a certain set of locomotives/crew. For example, a train may leave yard A at 7:00 of some day and arrive at yard B at 7:30, then leave to yard C at 7:45 and so on. Empty or loaded cars may be attached/detached to the trains at the yards. The stopping times (15 minutes in that example) are calculated to be enough for detaching some cars and attaching some other cars, which are supposed to be already handled and assembled into blocks. The *capacity* of a train is the maximum number of cars that can be attached to it.

Demands - A client asks for a *number of cars* of *types compatible with the commodity* that should be taken from a *set of origin yards* at each date of a *set of loading dates* and transported to a *set of destination yards*. This situation may define one demand or several demands. The defining characteristic of a demand is: loaded cars from the same demand are undistinguishable. For example, suppose that 40 cars are loaded with soybeans at yard A to be taken to yard B and another 40 cars are loaded with soybeans at yard C to be taken to yard D. If the soybean from A is really undistinguishable from the soybean from B, it is acceptable to define both orders as the same demand. In this case, the output solution may actually take some

cars from A to C and from B to D. Additional attributes of a demand are *loading/unloading times* and a *set of acceptable unloading dates*. Some demands must be necessarily met (by contractual reasons), but other demands may be only partially met, or even completely refused, for lack of resources (empty cars or train capacities) in the railroad network. In these cases, it is defined a monetary estimated *profit* per car actually transported from the origins to destinations. The usual objective function of the problem is to maximize such profits. Note that if the profit of taking a car from A to B is significantly different from the profit of taking a car from C to D, such orders should be defined as different demands, even if the involved commodities are undistinguishable.

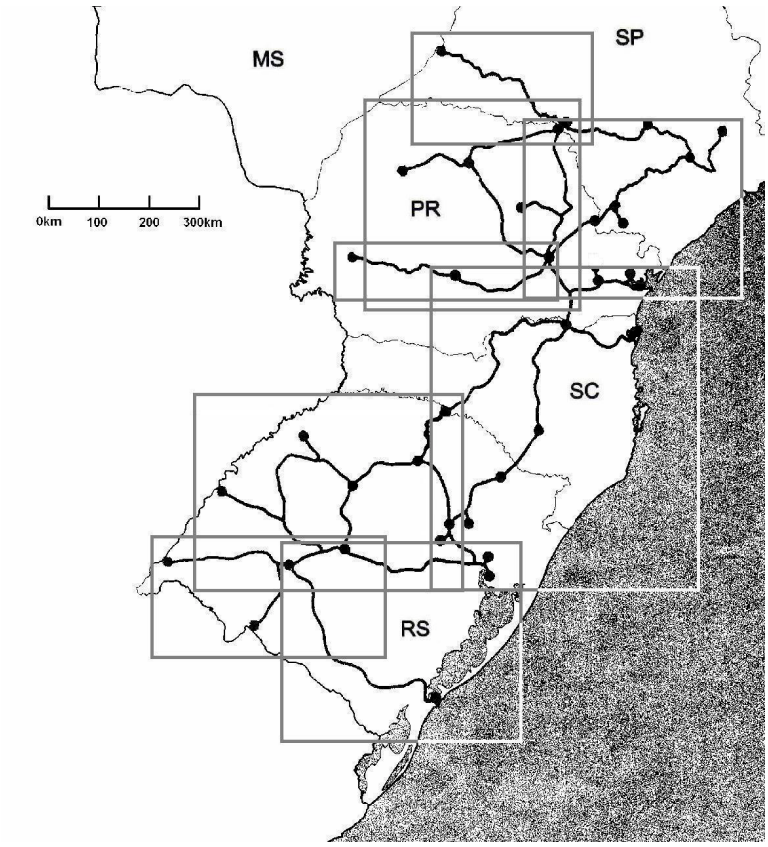

Fig. 1. The railroad network and its 8 operation zones (indicated by rectangles).

We give additional details of the particular problem we faced.

- The railroad operators desire as much regularity as possible in their operations. For this reason, a fixed weekly schedule of trains is decided quarterly, based on a forecast of the future demands. Of course, actual demands can be quite different from the forecast, so the flow car problem arises in order to meet as much as possible the actual demands without changing the train schedule.
- Demands that can not be fully met by the railroad are not backordered,

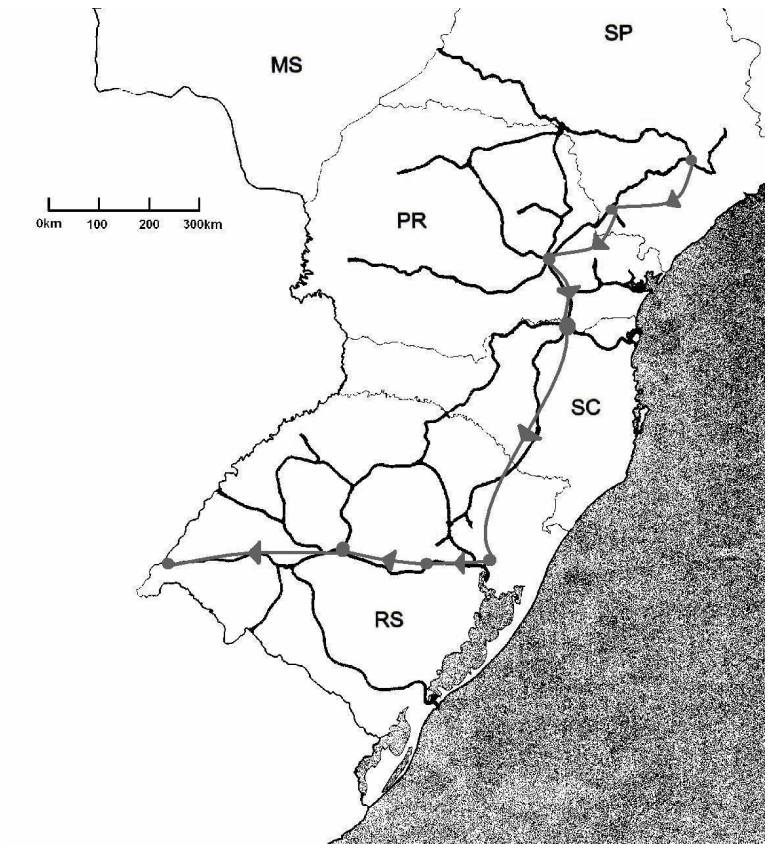

Fig. 2. Example of the route taken by a car to meet a demand between distant zones (each arrow represents a train).

because the operators have the option of transporting the remaining demand using its own truck fleet or hired trucks, at a higher cost. In those cases, profits are defined as the difference between truck and railway costs.

- The railroad network has a total length of 16,000 kilometers, spread among 4 Brazilian states: São Paulo (symbol SP), Paraná (PR), Santa Catarina (SC) and Rio Grande do Sul (RS). The network operators decided to divide it into 8 overlapping operational zones, they are shown as rectangles in the map of Figure 1. With the exception of very few special interzonal trains, the journeys of the scheduled trains are completely contained in a single zone. This policy eases a lot the problems of crew management and locomotive maintenance, but have the following effect on the car flow problem: demands across different zones can only be met by using at least 2 trains. Figure 2 gives an extreme example of this situation, where a car utilized 7 different trains to travel from origin to destination. Each arrow in the figure indicates a train, the circles are the origin, intermediary and destination yards.

3 Mathematical Model

Let D be the set of demands; Y the set of yards; K the set of car types; L the set of train legs; R the set of trains; and $T = \{1, 2, 3, \ldots, n\}$ the set numbering the successive relevant clock time instants. A clock time instant is called relevant if one of the following events happen: arrival or departure of a train leg; possible beginning or end of a loading/unloading or handling/attaching/detaching operation. Define $time(t)$ as the clock time instant having number t. Consider also that each train r has a set L_r of train legs $\{l_1^r, l_2^r, ..., l_{m_r}^r\}$, and that each leg l has origin $i(l)$ at time $t(l)$, and destination $j(l)$ at time $\tau(l)$.

Input data

- c_d: profit of demand d per car loaded.
- α_d and γ_d: loading and unloading times for each demand d.
- π_l and ψ_l: attaching and detaching times for train leg l. These times also include the handling times at the corresponding yards.
- $\kappa(d)$: set of car types compatible with demand d.
- ρ_{itd}: number of cars of types compatible with demand d requested at yard i and time t.
- η_{itd}: maximum number of cars loaded with demand d that can be unloaded at yard i and time t.
- CAP_l: capacity of train leg l.
- $YCAP_i$: capacity of yard i.

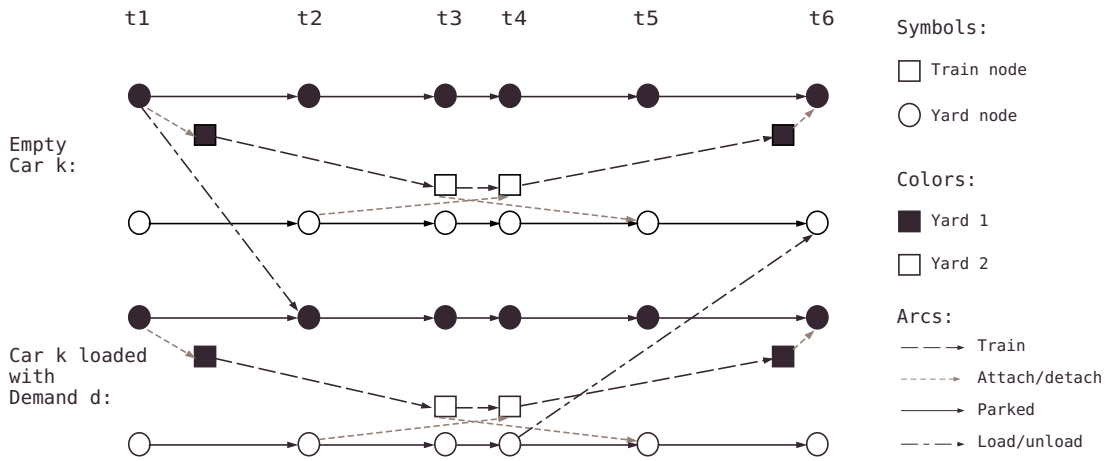

Fig. 3. Small example of car and demand subnetworks.

In our multicommodity flow formulation, each empty car type and each car type loaded with a certain demand defines a commodity. So we have an overall number of $|K| + \sum_d |\kappa(d)|$ commodities flowing on networks that

have yard nodes and train nodes. The arcs between yard nodes represent cars that are parked at that yard, and not attached to any train. Arcs between train nodes represent cars attached to a train, either stopped at a yard or moving in a train leg. Arcs between a yard and a train node represent the handling/attach/detach operations. Finally, that are the loading and unloading arcs that allow to transform empty cars of some type into cars of that type loaded with a certain demand and vice-versa.

We define more formally the directed graph $G = (V, A)$ described above (also see figure 3):

Graph entities

(i) Vertices:

- v_{itk}: yard vertices for the subnetwork of empty cars k.
- u_{itkd}: yard vertices for the subnetwork of cars of type k loaded with demand d.
- w_{kl} and w'_{kl}: train vertices for the subnetwork of empty cars k for the origin and destination (respectively) of train leg l.
- z_{kdl} and z'_{kdl}: train vertices for the subnetwork of cars k loaded with demand d for the origin and destination (respectively) of train leg l.

(ii) Arcs:

- $(v_{itk}, v_{i(t+1)k})$: Parked car arcs at yard i from $time(t)$ to $time(t+1)$ for empty car type k.
- $(u_{itkd}, u_{i(t+1)kd})$: Parked car arcs at yard i from $time(t)$ to $time(t+1)$ for car type k loaded with demand d.
- (w_{kl}, w'_{kl}): Car movement arcs for car k and train leg l.
- (z_{kdl}, z'_{kdl}): Car movement arcs for car k, loaded with demand d, and for train leg l.
- $(w'_{kl^r_q}, w_{kl^r_{q+1}})$: Cars of type k stopped between train legs l^r_q and l^r_{q+1} of train r.
- $(z'_{kdl^r_q}, z_{kdl^r_{q+1}})$: Cars of type k loaded with demand d stopped between train legs l^r_q and l^r_{q+1} of train r.
- $(v_{i(l)t_ik}, w_{kl})$: Attaching arcs at yard $i(l)$, from $time(t_i) = time(t(l)) - \pi_l$ to $time(t(l))$ for car type k and train leg l.
- $(u_{i(l)t_ikd}, z_{kdl})$: Attaching arcs at yard $i(l)$, from $time(t_i) = time(t(l)) - \pi_l$ to $time(t(l))$ for car type k loaded with demand d, for train leg l.
- $(w'_{kl}, v_{j(l)t_fk})$: Detaching arcs at yard $j(l)$ from $time(\tau(l))$ to $time(t_f) = time(\tau(l)) + \psi_l$ for car type k and train leg l.
- $(z'_{kdl}, u_{j(l)t_fkd})$:Detaching arcs at yard $j(l)$ from $time(\tau(l))$ to $time(t_f) = time(\tau(l)) + \psi_l$ for car type k loaded with demand d, for train leg l.
- (v_{itk}, u_{it_fkd}): Loading arcs at yard i from $time(t)$ to $time(t_f) = time(t) + \alpha_d$ of cars of type k to be used by demand d. These arcs only exist if $k \in \kappa(d)$.
- (u_{itkd}, v_{it_fk}): Unloading arcs at yard i from $time(t)$ to $time(t_f) = time(t) + \gamma_d$ of cars of type k that were used by demand d. These

arcs only exist if $k \in \kappa(d)$.

Let $A1_l$ be the set of all car movement arcs associated to train leg l; $A2_{it}$ the set of all parked car arcs for a certain pair (i,t); $A3_{itd}$ the set of all loading arcs at yard i and time t for demand d; and $A4_{itd}$ the set of all unloading arcs at yard i and time t for demand d.

The only decision variables we have in the model are x_a, which represent the number of cars that flow on arc a. That variable have different meanings depending on the type of arc.

The mathematical model is:

$$\text{Max} \sum_{i \in Y} \sum_{t \in T} \sum_{d \in D} \sum_{a \in A3_{itd}} c_d.x_a - \sum_{l \in L} \sum_{a \in A1_l} \epsilon.x_a$$

s.t.

$$\sum_{a \in \delta^+(v)} x_a = \sum_{a \in \delta^-(v)} x_a + b, \forall v \in V \tag{1}$$

$$\sum_{a \in A1_l} x_a \leq CAP_l, \forall l \in L \tag{2}$$

$$\sum_{a \in A2_{it}} x_a \leq YCAP_i, \forall i \in Y, t \in T \tag{3}$$

$$\sum_{a \in A3_{itd}} x_a \leq \rho_{itd}, \forall i \in Y, \forall t \in T, \forall d \in D \tag{4}$$

$$\sum_{a \in A4_{itd}} x_a \leq \eta_{itd}, \forall i \in Y, \forall t \in T, \forall d \in D \tag{5}$$

$$\text{All variables are integral} \tag{6}$$

Constraint (1) define the flow conservation constraints for all the vertices of all the subnetworks. The right-hand side b is only different from zero only on vertices corresponding to the initial states (or in some rare situations where cars enter or leave the network in the middle of the period). Trains and yards capacities are modeled by (2) and (3). Constraints (4) state that the loading arcs are bounded by the number of cars requested at that moment. Similarly, (5) bounds the unloading operations at each possible moment. Note that one may leave a degree of freedom in the model and allow unloading to be performed at different possible dates by setting parameters in a way such that the sum of the η_{itd} for a given d is greater than the sum of ρ_{itd} for the same d.

The usual objective in this model is to maximize the profits from transporting cars to meet demands. Since the total number of cars actually loaded and unloaded must be the same for each demand, any of these sets of variables could be used in the objective function. We also assign to movement arcs some residual costs to assure that no unnecessary movements will be made.

A similar model was used by Holmberg et al. [5] to find an optimal flow of empty cars on the remaining train capacities after a loaded car flow is given. In that case, there were only $|K|$ commodities and no load/unload arcs. Other

recent articles (Brucker et al.[2] and Cordeau et al. [3]) also describe the use of integer multicommodity flows to model car flow problems in passenger trains. In that kind of problems, there are no demands, the cars of each type (first class, second class, restaurant, etc) have to be moved in order to be assembled into the scheduled trains. The resulting networks are not as large as in our case, but all those authors reported good results due to the high quality of the bounds provided by the linear relaxation.

We finish by remarking that the output of our model is a feasible flow. As we want to know the route of each particular car in the network, we apply a standard flow decomposition procedure, as described in [1].

4 Preprocessing the model

The above model is defined over a multicommodity network composed by $|K| + \sum_d |\kappa(d)|$ subnetworks (around 550), each one composed by up to $|Y| \cdot |T| + |L|$ vertices and about the same number of arcs. In our test instances, set K is small ($|K| = 25$), set Y is medium ($|Y| = 150$), but $|T|$ can vary from around 200 to more than one thousand. The large size of $|T|$ reflects the fact that events like the arrival or departure of a train leg happen in many distinct times during the periods under consideration. As a result, the size of the model turned out to be really big, and we were not able even to allocate the memory necessary to load into a MIP solver. One possible remedy would be to reduce the size of T by constraining the time instants to be multiples of some larger unit, say 30 minutes. Of course, this would reduce by some degree the fitness of the model to the real world problem. We preferred to devise a preprocessing procedure to reduce the model size to a reasonable one without compromising its accuracy.

The first preprocessing procedure, called PRE-Degree, consists in eliminating intermediate vertices of degree two. Figure 4 illustrates this procedure. For all the instances tested, this procedure already reduces the number of arcs in the network to less than 10% of their original number. This reduction is significant, but this was not enough to allow the solution of the models by a MIP solver.

A slightly more sophisticated preprocessing procedure, called PRE-Path tries to eliminate arcs in the demands subnetworks. The idea is that arc (i, j) is certainly not used by the cars loaded with demand d if there is no path in that subnetwork from a vertex corresponding to an origin of demand d to i and another path from j to a vertex corresponding to a destination of demand d. This simple idea allows the removal of many arcs corresponding to train legs, therefore reducing the degree of many vertices. After that, the test PRE-Degree may be applied again to achieve further reductions. For all the instances tested, the combined use of procedures PRE-Degree and PRE-Path reduced the number of arcs in the network to less than 1% of their original number. The resulting models could then be loaded and solved by a MIP

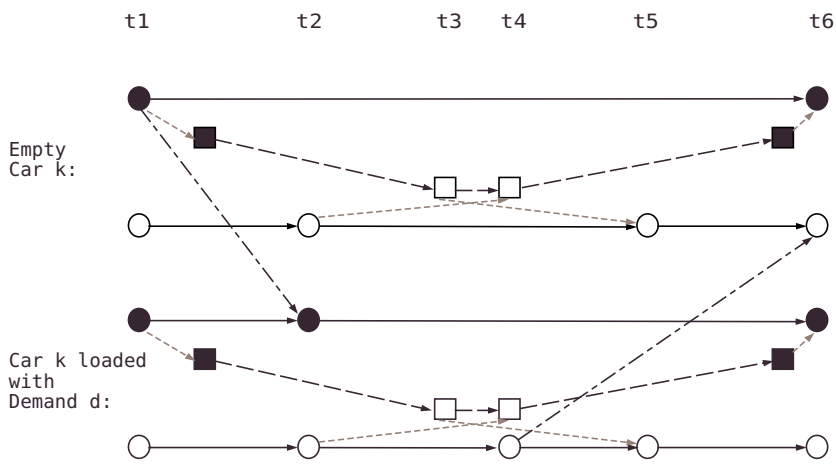

Fig. 4. Example of application of the procedure PRE-Degree to figure 3.

solver.

We only remark that modern MIP solvers already have very good preprocessing mechanisms, that probably would have performed those same reductions — if the original model could be fed into them.

5 Results

We present computational results from solving the model corresponding to 8 typical real instances in table 1. These runs were executed on a Pentium-III 800Mhz, with 768MB of RAM. The MIP solver utilized was the CPLEX 7.1. Some columns of Table 1 require some explanation. Column # is the instance number. Column **nC** is the total number of commodities of the problem ($|K| + \sum_d |\kappa(d)|$). Columns **rows** and **cols** are the number of rows and columns in the model after our preprocessing. Column **LPT** is the time in seconds required to solve the linear relaxation of the model, **BB** is the number of nodes in the branch-and-bound tree, **TT** is the total time in seconds to find an optimal integral solution and **Gap** is the percentual duality gap between the values of the optimal fractional and integral solutions. For all those 8 instances, there was no duality gap. Nevertheless, in 4 cases the solution obtained at the root node of the branch-and-bound tree was fractional, and some branching had to be performed in order to find an integral solution with exactly the same cost of that first fractional solution. This confirms the good quality of the bounds obtained by the multicommodity flow model.

6 Conclusions

We presented a model for the freight car flow problem that turned out to be quite effective in practice, allowing us to solve to optimality all the instances ever tested from the largest Latin America railroad operator company. A

#	$\|T\|$	$\|Y\|$	$\|D\|$	$\|L\|$	nC	*cols*	*rows*	LPT (s)	BB	TT (s)	Gap
1	397	159	397	1675	548	1644900	1213668	1168	2	1764	0.00
2	397	158	355	1675	530	1283438	949908	713	2	1336	0.00
3	397	154	382	1675	544	1255263	936404	429	1	435	0.00
4	417	159	373	1701	537	1224948	897270	611	1	623	0.00
5	212	159	397	1542	548	1542751	1163499	1306	1	1320	0.00
6	1045	159	397	1722	548	1817653	1266603	1345	1	1364	0.00
7	349	34	364	1511	521	1054931	724535	1149	3	1911	0.00
8	1075	159	405	2356	583	3592027	2650146	5012	12	17530	0.00

Table 1
Results for 8 instances.

simple but effective preprocessing scheme also played an important role in this achievement.

The resulting model and algorithm are incorporated into a complete software package, called OptVag, that also comprises a graphical user interface and the ability of receiving on-line information (such as the current position of each car in the network) from a database. Such package is currently used to support the operational decisions at that company.

Some improvements and extensions to the model are currently being developed, such a better measure of the train capacities in terms of lengths and weights, instead of number of cars, and an adaptation for a multimodal logistics problem also involving trucks.

References

[1] Ahuja, R., T. Magnanti and J. Orlin, "Network Flows: Theory, Algorithms, and applications," Prentice Hall, 1993.

[2] Brucker, P., J. Hurink and T. Rolfes, *Routing of railway carriages: A case study*, Technical Report 1498, University of Twente, Netherlands (1999).

[3] Cordeau, J. F., F. Soumis and J. Desrosiers, *Simultaneous assignment of locomotives and cars to passenger trains*, Operations Research **49** (2001), pp. 531–548.

[4] Cordeau, J. F., P. Toth and D. Vigo, *A survey of optimization models for train routing and scheduling*, Transportation Science **32** (1998), pp. 380–404.

[5] Holmberg, K., M. Joborn and J. Lundgren, *Improved empty freight car distribution*, Transportation Science **32** (1998), pp. 163–173.