Freight railcar routing problem arising in Russia

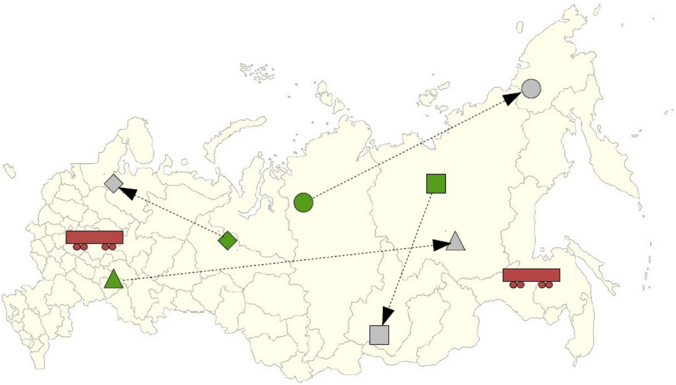
Ruslan Sadykov[1] Alexander A. Lazarev[2]
Vitaliy Shiryaev[3] Alexey Stratonnikov[3]

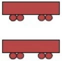
[1] INRIA Bordeaux, Talence, France
[2] Institute of Control Sciences, Moscow, Russia
[3] JSC Freight One Moscow, Russia

EURO 2013
Rome, Italy, July 2

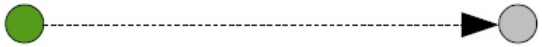
Contents

The freight car routing problem: overview





initial car distribution



transportation demands

Specificity of freight rail transportation in Russia

- The fleet of freight railcars is owned by independent freight companies
- Forming and scheduling or trains is done by the state company
  - It charges a cost for transferring cars and determines (estimated) travel times
  - Cost for the transfer of an empty car depends on the type of previously loaded product
- Distances are large, and average freight train speed is low ($\approx$ 300 km/day): discretization in periods of 1 day is reasonable

The freight car routing problem: input and output

Input

- Railroad network (stations)
- Initial locations of cars (sources)
- Transportation demands and associated profits
- Costs: transfer costs and standing (waiting) daily rates;

Output: operational plan

- A set of accepted demands and their execution dates
- Empty and loaded cars movements to meet the demands (car routing)

Objective

Maximize the total net profit

Data: overview

- T — planning horizon (set of time periods);
- I — set of stations;
- C — set of car types;
- K — set of product types;
- Q — set of demands;
- S — set of sources (initial car locations);
- M — empty transfer cost function;
- D — empty transfer duration function;

Demands data

For each order $q \in Q$

- $i_q^1, i_q^2 \in I$ — origin and destination stations;
- $k_q \in K$ — product type
- $C_q \subseteq C$ — set of car types, which can be used for this demand
- $n_q^{\max}(n_q^{\min})$ — maximum (minimum) number of cars, needed to fulfill (partially) the demand
- $r_q \in T$ — release time of demand
- $\Delta_q \in \mathbb{Z}_+$ — maximum delay for starting the transportation
- ρ_{qt} — profit from delivery of one car with the product, transportation of which started at period t, $t \in [r_q, r_q + \Delta_q]$
- $d_q \in \mathbb{Z}_+$ — transportation time of the demand
- $w_q^1(w_q^2)$ — daily standing rate charged for one car waiting before loading (after unloading) the product at origin (destination) station

Sources and car types data

For each source $s \in S$

- $\vec{i}_s \in I$ — station where cars are located
- $\vec{c}_s \in C$ — type of cars
- $\vec{r}_s \in T$ — period, starting from which cars can be used
- $\vec{w}_s$ — daily standing rate charged for cars
- $\vec{k}_s \in K$ — type of the latest delivered product
- $\vec{n}_s \in \mathbb{N}$ — number of cars in the source

For each car type $c \in C$

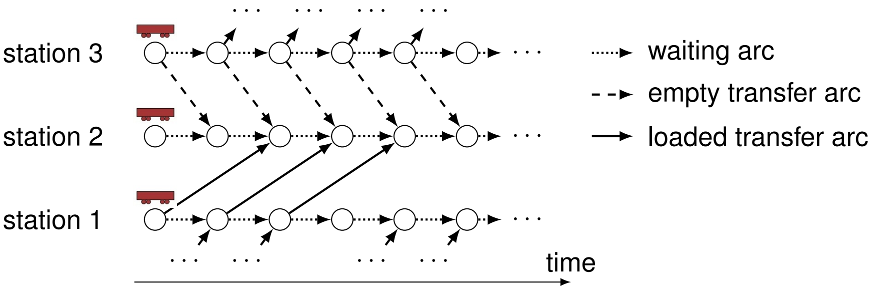
- Q_c — set of demands, which a car of type c can fulfill
- S_c — set of sources for car type c

Contents

Commodity graph

Commodity $c \in C$ represents the flow (movements) of cars of type c.

Graph $G_c = (V_c, A_c)$ for commodity $c \in C$:



Graph definition

- vertex v_{cit}^{wk} — stay of cars of type $c \in C$ at station $i \in I$ at daily waiting rate w at period $t \in T$, where $k \in K$ is the type of unloaded product. Flow balance is

$$b(v_{cit}^{wk}) = \begin{cases} \vec{n}_s, & \exists s \in S_c : \vec{i}_s = i, \vec{r}_s = t, \vec{w}_s = w, \vec{k}_s = k, \\ 0, & \text{otherwise.} \end{cases}$$

- waiting arc a_{cit}^{wk} — waiting of cars of type $c \in C$ from period $t \in T$ to $t + 1$ at station $i \in I$ at daily rate w, $k \in K$ is the type of previously loaded product. Cost $g(a)$ is w.
- empty transfer arc $a_{cijt}^{w'w''k}$ — transfer of empty cars of type $c \in C$ waiting at station $i \in I$ at daily rate w' to station $j \in I$ where they will wait at daily rate w'', such that the type of latest unloaded product is $k \in K$, and transfer starts at period $t \in T$. Cost is $M(c, i, j, k)$.
- loaded transfer arc a_{cqt} — transportation of demand $q \in Q$ by cars of type $c \in C$ starting at period $t \in T \cap [r_q, r_q + \Delta_q]$. Cost is $-\rho_{qt}$.

Multi-commodity flow formulation

Variables

- $x_a \in \mathbb{Z}_+$ — flow size along arc $a \in A_c$, $c \in C$
- $y_q \in \{0, 1\}$ — demand $q \in Q$ is accepted or not

$$\min \sum_{c \in C} \sum_{a \in A_c} g(a) x_a$$

$$\sum_{c \in C_q} \sum_{a \in A_{cq}} x_a \leq n_q^{\max} y_q \quad \forall q \in Q$$

$$\sum_{c \in C_q} \sum_{a \in A_{cq}} x_a \geq n_q^{\min} y_q \quad \forall q \in Q$$

$$\sum_{a \in \delta^-(v)} x_a - \sum_{a \in \delta^+(v)} x_a = b(v) \quad \forall c \in C, v \in V_c$$

$$0 \leq x_a \quad \forall c \in C, a \in V_c$$

$$0 \leq y_q \leq 1 \quad \forall q \in Q$$

We concentrate on solving its LP-relaxation

Path reformulation

- P_s — set of paths (car routes) from source $s \in S$

Variables

- $\lambda_s \in \mathbb{Z}_+$ — flow size along path $p \in P_s$, $s \in S$

$$
\begin{aligned}
\min \sum_{c \in C} \sum_{s \in S_c} \sum_{p \in P_s} g_p^{path} \lambda_p \\
\sum_{c \in C_q} \sum_{s \in S_c} \sum_{p \in P_s:\ q \in Q_p^{path}} \lambda_a \le n_q^{\max} y_q \quad & \forall q \in Q \\
\sum_{c \in C_q} \sum_{s \in S_c} \sum_{p \in P_s:\ q \in Q_p^{path}} \lambda_a \ge n_q^{\min} y_q \quad & \forall q \in Q \\
\sum_{p \in P_s} \lambda_p = \vec{n}_s \quad & \forall c \in C, s \in S_c \\
\lambda_p \in \mathbb{Z}_+ \quad & \forall c \in C, s \in S_c, p \in P_s \\
y_q \in \{0, 1\} \quad & \forall q \in Q
\end{aligned}
$$

Column generation for path reformulation

- Pricing problem decomposes into shortest path problems for each source
  - slow: number of sources are thousands
- To accelerate, for each commodity $c \in C$, we search for a shortest path in-tree to the terminal vertex from all sources in S_c
  - drawback: some demands are severely “overcovered”, bad convergence
- We developed iterative procedure which removes covered demands and cars assigned to them, and the repeats search for a shortest path in-tree

Iterative pricing procedure for commodity $c \in C$

foreach *demand* $q \in Q_c$ **do** $uncvCars_q \leftarrow n_q^{\max}$;
foreach *source* $s \in S_c$ **do** $rmCars_s \leftarrow \vec{n}_s$;
$iter \leftarrow 0$;
repeat
  Find an in-tree to the terminal from sources $s \in S_c$, $rmCars_s > 0$;
  Sort paths p in this tree by non-decreasing of their reduced cost;
  foreach *path p in this order* **do**
    if $\bar{g}_p < 0$ **and** $uncvCars_q > 0,\ \forall q \in Q_p^{path}$, **then**
      Add variable λ_p to the restricted master;
      $s \leftarrow$ the source of p;
      $rmCars_s \leftarrow rmCars_s - \min\{rmCars_s, uncvCars_q\}$;
      $uncvCars_q \leftarrow uncovCars_q - \min\{rmCars_s, uncvCars_q\}$;
  $iter \leftarrow iter + 1$;
until $uncvCars_q > 0,\ \forall q \in Q_c$, **or** $rmCars_s > 0,\ \forall s \in S_c$, **or** $iter = \texttt{nbPricIter}$;

Flow enumeration reformulation

- F_c — set of fixed flows for commodity $c \in C$

Variables

- $\omega_f \in \{0, 1\}$ — commodity c is routed accordity to flow $f \in F_c$ or not

$$
\begin{aligned}
\min \sum_{c \in C} \sum_{f \in F_s} g_f^{flow} \omega_f & \\
\sum_{c \in C_q} \sum_{f \in F_c} \sum_{a \in A_{cq}} f_a \omega_f \le n_q^{\max} y_q & \quad \forall q \in Q \\
\sum_{c \in C_q} \sum_{f \in F_c} \sum_{a \in A_{cq}} f_a \omega_f \ge n_q^{\min} y_q & \quad \forall q \in Q \\
\sum_{f \in F_c} \omega_f = 1 & \quad \forall c \in C \\
\omega_p \in \{0, 1\} & \quad \forall c \in C, f \in F_c \\
y_q \in \{0, 1\} & \quad \forall q \in Q
\end{aligned}
$$

Approach CGEF

- Pricing problem decomposes into minimum cost flow problem for each commodity
  - slow: very bad convergence
- "Column generation for extended formulations" (CGEF) approach: we disaggregate the pricing problem solution into arc flow variables, which are added to the master.
- The master then becomes the multi-commodity flow formulation with restricter number of arc flow variables, i.e. "improving" variables are generated dynamically

Proposition

If an arc flow variable x has a negative reduced cost, there exists a pricing problem solution in which $x > 0$.
(consequence of the theorem in [S. and Vanderbeck, 13])

Contents

Tested approaches

- DIRECT: solution of the multi-commodity flow formulation by the *Clp* LP solver
  - Problem specific solver source code modifications
  - Problem specific preprocessing is applied (not public)
  - Tested inside the company

- COLGEN: solution of the path reformulation by column generation (*BaPCod* library and *Cplex* LP solver)
  - Initialization of the master by “doing nothing” routes
  - Stabilization by dual prices smoothing
  - Restricted master clean-up

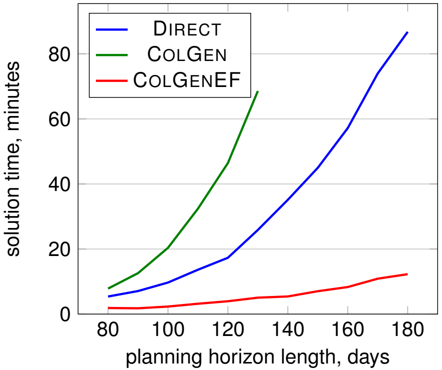
- COLGENEF: “dynamic” solution of multi-commodity flow formulation by the CGEF approach (*BaPCod* library, *Lemon* min-cost flow solver and *Cplex* LP solver)
  - Initialization of the master by all waiting arcs
  - Only trivial preprocessing is applied

First test set of real-life instances

Instance name	x3	x3double	5k0711q
Number of stations	371	371	1’900
Number of demands	1’684	3’368	7’424
Number of car types	17	17	1
Number of cars	1’013	1’013	15’008
Number of sources	791	791	11’215
Time horizon, days	37	74	35
Number of vertices, thousands	62	152	22
Number of arcs, thousands	794	2’846	1’843
Solution time for DIRECT	20s	1h34m	55s
Solution time for COLGEN	22s	7m53s	8m59s
Solution time for COLGENEF	3m55s	>2h	43s

Real-life instances with larger planning horizon

1'025 stations, up to 6'800 demands, 11 car types, 12'651 cars, and 8'232 sources.
Up to ≈ 300 thousands nodes and 10 millions arcs.



Horizon	Direct	ColGenEF
80	5m24s	1m52s
90	7m05s	1m47s
100	9m42s	2m19s
110	13m38s	3m11s
120	17m19s	3m57s
130	25m52s	5m03s
140	35m08s	5m25s
150	44m58s	7m02s
160	57m11s	8m19s
170	1h13m58s	10m53s
180	1h26m46s	12m16s

Convergence of ColGenEF in less than 15 iterations.
About 3% of arc flow variables at the last iteration.

Conclusions

- Three approaches tested for a freight car routing problem on real-life instances
- Approach ColGen is the best for instances with small number of sources
- Problem-specific preprocessing is important: good results for Direct
- Approach ColGenEF is the best for large instances
- Combination of ColGenEF and problem-specific preprocessing would allow to increase discretization and improve solutions quality

Perspectives

Some practical considerations are not taken into account:

- Progressive standing daily rates
- Special stations for long-time stay (with lower rates)
- Compatibility between two consecutive types of loaded products.
- Penalties for refused demands
- Groups of cars are transferred faster and for lower unitary costs.