Problem Solving Competition 2010

Virot Chiraphadhanakul, Cristian Figueroa

Operations Research Center, Massachusetts Institute of Technology, Cambridge MA 02139

{virot,cfiguero}@mit.edu

1 Executive Summary

To our knowledge, a problem with this particular structure has not been solved before. Although the problem resembles a dynamic lot-sizing problem, it contains many hard constraints such as joint truck cost, locomotive capacity, and limited number of stops. We model the problem as a mixed-integer program and propose several solution methods, including lagrangian relaxation and adding valid cuts. Our best solution has an optimality gap smaller than 0.25%.

2 Problem Formulation

Using the notation in Table 1, we formulate the following mixed-integer program. The decisions to be made are the number of trucks contracted at each yard and the fuel plan for each locomotive.

$$\text{(P)} \quad \text{Minimize} \quad \sum_{i\in I}\sum_{j\in J}\sum_{k\in K_j}(C_i^f\alpha_{i,k}^j)w_k^j + C^s\sum_{j\in J}\sum_{k\in K_j}x_k^j + C^t\sum_{i\in I}z_i \tag{2.1}$$

$$\text{subject to} \quad y_j + \sum_{k=1}^{m-1}(w_k^j - R\cdot D_k^j) + w_m^j \le LC \qquad \forall j\in J, m = 1,...,|K_j| \tag{2.2}$$

$$y_j + \sum_{k=1}^{m}(w_k^j - R\cdot D_k^j) \ge 0 \qquad \forall j\in J, m = 1,...,|K_j|-1 \tag{2.3}$$

$$\sum_{k\in K_j}(w_k^j - R\cdot D_k^j) \ge 0 \qquad \forall j\in J \tag{2.4}$$

$$w_k^j \le LC\cdot x_k^j \qquad \forall j\in J, k\in K_j \tag{2.5}$$

$$\sum_{k\in L}x_k^j \le 2 \qquad \forall j\in J, L\in L_j \tag{2.6}$$

$$\sum_{j\in J}\sum_{k\in K_j}(\alpha_{i,k}^j\cdot\beta_{k,t}^j)w_k^j \le TC\cdot z_i \qquad \forall i\in I, t\in T \tag{2.7}$$

$$w_k^j \geq 0 \qquad \forall j \in J, k \in K_j \tag{2.8}$$

$$x_k^j \in \{0,1\} \qquad \forall j \in J, k \in K_j \tag{2.9}$$

$$y_j \geq 0 \qquad \forall j \in J \tag{2.10}$$

$$z_i \in \mathbb{Z}^+ \qquad \forall i \in I \tag{2.11}$$

The objective is to minimize the sum of all the costs– fueling cost, stopping cost and truck contracting cost. Constraints (2.2) ensure that the fuel level of each locomotive never exceeds the locomotive tank capacity. Constraints (2.3) ensure that each locomotive never runs out of fuel along the way. Additionally, constraints (2.4) enforces that the remaining fuel of each locomotive after completing its cycle is larger than the initial fuel required. Constraints (2.5) ensure that the amount of fuel dispensed to a locomotive can be positive only when it stops there. The number of intermediate stops any train could make is limited through constraints (2.6). Constraints (2.7) guarantee that the number of contracted trucks at each yard is sufficient for the fuel plan. Finally, equations (2.8)-(2.11) specify the characteristics of each decision variable.

Although the problem resembles a dynamic lot-sizing problem, it contains many hard constraints such as joint truck cost, locomotive capacity, and limited number of stops. Therefore, solving this problem will require a large computational time, especially when the problem size is large.

Table 1: Notation

Sets	I	set of yards, indexed by i.
	J	set of locomotive routes (or lomotives), indexed by j.
	K_j	sequence of stops in locomotive route j, indexed by k.
	L_j	set of sequences of intermediate stops in locomotive route j, indexed by L.
	T	days of operations $= \{1, ..., 14\}$, indexed by t.
Constants	R	fuel consumption rate.
	D_k^j	distance from k-th stop to $(k+1)$-th stop (or 1st stop, if $k = \|K_j\|$) of locomotive route j.
	TC	truck capacity.
	LC	locomotive's tank capacity.
	C^t	truck contracting cost during the horizon (\$ per truck).
	C_i^f	fuel cost at yard i (\$ per gallon).
	C^s	fixed refueling cost (\$ per stop).
	$\alpha_{i,k}^j$	equal to 1 if yard i is the k-th stop of locomotive route j; 0 otherwise.
	$\beta_{k,t}^j$	equal to 1 if the k-th stop of locomotive route j is visited on day t; 0 otherwise.
Decision Variables	w_k^j	amount of fuel dispensed at the k-th stop of locomotive route j.
	x_k^j	equals to 1 if locomotive j stops for refueling at its k-th stop.
	y_j	initial fuel level of locomotive j.
	z_i	number of trucks contracted at yard i.

3 Solution Method

Our initial approach was to implement the MIP and solve it using a commercial solver, CPLEX. While we could solve the small example to optimality in a matter of seconds, the given instance was not solved to optimality after hours, although the best solution already has an optimality gap of less than 1%. This happened because the LP relaxation of the formulation does not provide a tight bound, and thus there is no certificate that the best solution found is actually optimal. Therefore, we tried several approaches that can potentially acheive smaller optimality gap in a short time.

3.1 Adding Valid Cuts

One way to improve the bound obtained from an LP relaxation is to add valid cuts– constraints that can be added to the formulation without affecting the feasible set of the original problem. The first set of constraints are:

$$\sum_{k=1}^{l} x_k^j \geq \frac{\sum_{k=1}^{l} R \cdot D_k^j - y_j}{LC} \qquad \forall j \in J, l = 1, ..., |K_j| - 1 \qquad (3.1)$$

$$\sum_{k=|K_j|-l+1}^{|K_j|} x_k^j \geq \frac{\left(\sum_{k=|K_j|-l+1}^{|K_j|} R \cdot D_k^j + y_j\right) - \left(4500 - R \cdot D_{|K_j|-l}^j\right)}{LC} \qquad \forall j \in J, l = 1, ..., |K_j| - 1 \qquad (3.2)$$

$$\sum_{k=1}^{|K_j|} x_k^j \geq \frac{\sum_{k=1}^{l} R \cdot D_k^j}{LC} \qquad (3.3)$$

Consider constraints of the form (3.1). Given an initial fuel level y_j, the total fuel required for locomotive j to get from the first stop to l-th stop is $\sum_{k=1}^{l} R \cdot D_k^j - y_j$, and thus the locomotive needs to make at least $\frac{\sum_{k=1}^{l} R \cdot D_k^j - y_j}{LC}$ stops, assuming that it always fills up the tank at any stop. On the other hand, given that a locomotive should end its cycle with the initial fuel, we can apply a similar idea backward and obtain (3.2), assuming that a fuel level at the $(|K_j| - l)$-th stop is maximum. Lastly, the last equation obtained when we consider either forward or backward direction for the entire cycle.

Additionally, we add $x_k^j \leq \sum_{i \in I} \alpha_{i,k}^j z_i$ for all $j \in J$ and $k \in K_j$. This will ensure that a locomotive will not make a stop at that particular yard for refueling when $z_i = 0$. Lastly, we add $z_i \leq \left\lceil \frac{LC \cdot \bar{Z}_i}{TC} \right\rceil$ for all $i \in I$, where $\bar{Z}_i$ is the maximum number of locomotives visiting yard i on any particular day. This set of constraints will be

used extensively in lagrangian relaxation to bound the feasible range of z_i.

3.2 Lagrangean Relaxation

One common way to solve large-scale optimization is to use lagrangian relaxation, especially when the problem becomes much easier to solve or decomposable after relaxing some constraint. In this case, by relaxing the contraint (2.7), the resulting formulation can be solved separately for a fuel plan of each locomotive and the number of contracted trucks.

To obtain an optimal lagragian relaxation solution, we first set the value of z_i according to its cost coefficient, which depends on the current lagragrian multiplier, i.e., if the cost is positive, we set z_i to zero, otherwise we make it equal to the bound introduced in the previous section. In order to solve for a fuel plan of each locomotive, we re-formulate the problem as a shortest cycle problem. This is possible due to the fact that there exists an optimal solution such that for any two consecutive fueling stops k' and k visited by a locomotive, the locomotive either leaves k' with a full tank or arrives at k with an empty tank.

Although the idea seems somewhat similar to Nourbakhsh and Ouyang (2010) [2], we propose a novel approach to solve this particular problem as follows. Firstly, because an intial fuel level is a decision variable in this case, many nodes in the network could be a starting point, and we thus need to solve all-pair shortest path problem for a shortest cycle. Floyd-Warshall's algorithm is used in our implementation. Secondly, we introduce additional nodes to the network to enable us to completely keep track of the number of stops a train has been made, and hence ensure that a train does not make more than two stops. Lastly, because the running time of Floyd-Warshall's algorithm is $O(n^3)$ where n is a number of node, we reduce the number of nodes further, using the fact that in an optimal solution, for any two consecutive fueling stops k' and k visited by a locomotive, the locomotive never leaves k' with a full tank if a fuel price at stop k' is larger than k. The detail of the problem re-formulation is provided in the appendix.

Using the lagragian relaxation solution (LR), we construct two feasible solutions. In the first solution, we fix x_j^k to zero in the original MIP if it is zero in (LR) and solve using CPLEX solver. Note that a locomotive is now allowed to stop only at the stops included in (LR). This restricted problem can be solved to optimality very fast because most of the x_j^k's and w_j^k's that are fixed to zero can be removed from the original problem. Additionally,

we can also omit the constraints (2.6) because the set of x_j^k's that remain in the formulation is gauranteed to satisfy the constraints. For the second solution, we solve the shortest cycle problems again using only yards with positive z_i. We then obtain a feasible solution from CLPEX in a similar manner as before.

Lastly, we use the subgradient method and the step size formula provided in Held et al. (1974) [1] to update lagrangian multipliers.

4 Implementation and Results

We solved the given instance using three different methods: (i) using CPLEX to solve the original MIP, (ii) using CPLEX to solve the original MIP with additional valid cuts, and (iii) using lagrangian relaxation approach detailed in the previous section. The time limit for each run is 10 minutes. All the methods are implemented in Java 1.6 and CPLEX 11.2 and run on a 2 GHz CPU and 2 GB memory PC. The computational results are as follows:

Methods	Best feasible solution	Best lowerbound	Optimality Gap
(i)	$ 11,449,604.94	$ 11,316,900	1.16%
(ii)	$ 11,402,334.22	$ 11,374,400	0.25%
(iii)	$ 11,512,278.75	$ 11,198,033	2.80%

In addition to the results shown, we solved the problem using different methods up to one hour. The best solution was obtained by using CPLEX to solve the original MIP with additional valid cuts. The cost of the solution is $11,401,272.58. The detail of the solution is in the submitted Excel file.

Even though the lagrangean relaxation did not perform as well as the other methods, we still expect that the lagrangean relaxation approach should outperform CPLEX in larger problem. Additionally, from the observation that given a set of stops that each locomotive is allowed to refuel, CPLEX can solve to problem to the optimality very fast. It might be worth applying a genetic algorithm on x_k^j's.

A Appendix

The lagragian relaxation is given by

$$
\begin{aligned}
(\mathsf{L}) \qquad \text{Minimize} \quad & \sum_{i\in I}\sum_{j\in J}\sum_{k\in K_j}(C_i^f\alpha_{i,k}^j)w_k^j + C^s\sum_{j\in J}\sum_{k\in K_j}x_k^j + C^t\sum_{i\in I}z_i \\
& - \sum_{i\in I}\sum_{t\in T}u_{i,t}\left[TC\cdot z_i - \sum_{j\in J}\sum_{k\in K_j}(\alpha_{i,k}^j\cdot\beta_{k,t}^j)w_k^j\right] \\
= \quad & \sum_{i\in I}\sum_{j\in J}\sum_{k\in K_j}\left[C_i^f + \sum_{t\in T}(\beta_{k,t}^j u_{i,t})\right]\alpha_{i,k}^j w_k^j + C^s\sum_{j\in J}\sum_{k\in K_j}x_k^j + \sum_{i\in I}(C^t - TC\sum_{t\in T}u_{i,t})z_i \\
\text{subject to} \quad & \text{(2.2)-(2.6) and (2.8)-(2.11)}
\end{aligned}
$$

and the subproblem for each locomotive is as follows:

$$
(\mathsf{S}_j) \qquad \text{Minimize} \quad \sum_{i\in I}\sum_{j\in J}\sum_{k\in K_j}\left[C_i^f + \sum_{t\in T}(\beta_{k,t}^j u_{i,t})\right]\alpha_{i,k}^j w_k^j + C^s\sum_{j\in J}\sum_{k\in K_j}x_k^j \tag{A.1}
$$

$$
\text{subject to} \quad y_j + \sum_{k=1}^{m-1}(w_k^j - R\cdot D_k^j) + w_m^j \le LC \qquad \forall m = 1,...,|K_j| \tag{A.2}
$$

$$
y_j + \sum_{k=1}^{m}(w_k^j - R\cdot D_k^j) \ge 0 \qquad \forall m = 1,...,|K_j|-1 \tag{A.3}
$$

$$
\sum_{k=1}^{|K_j|}(w_k^j - R\cdot D_k^j) \ge 0 \tag{A.4}
$$

$$
w_k^j \le LC\cdot x_k^j \qquad \forall k\in K_j \tag{A.5}
$$

$$
\sum_{k\in L}x_k^j \le 2 \qquad \forall L\in L_j \tag{A.6}
$$

$$
w_k^j \ge 0 \qquad \forall k\in K_j \tag{A.7}
$$

$$
x_k^j \in \{0,1\} \qquad \forall k\in K_j \tag{A.8}
$$

$$
y_j \ge 0 \tag{A.9}
$$

To solve this particular problem, we first claim that there exists an optimal solution such that for any two consecutive fueling stops k' and k visited by a locomotive, the locomotive either leaves k' with a full tank or arrives at k with an empty tank.

Let Γ_k^j be a set of stops k' of locomotive j for which the amount of fuel required for traveling from k' to

k does not exceed the locomotive tank capacity. Mathematically, we have $\Gamma_k^j = \{k' \mid \sum_{m=0}^{\overline{k-k'-1}} R \cdot D^j_{\overline{k'+m}} \leq LC \quad \text{where } \overline{x} = x \bmod |K_j|\}$.

According to earlier claim, in an optimal solution, unless locomotive j arrives at stop $k \in K_j$ with an empty tank, it must depart from the previous refueling stop $k' \in \Gamma_k^j$ with a full tank. Therefore, there are only $1 + |\Gamma_k^j|$ possible optimal fuel levels when locomotive j arrives at k.

We can transform subproblem (S_j) into a network formulation as follows. For each stop k in locomotive route j, there are $1+|\Gamma_k^j|$ nodes, (k', k) where $k' \in \{0\} \cup \Gamma_k^j$, corresponding to different fuel levels at stop k. Additionally, if k is an intermediate stop (except the intermediate stop immediately after an origin), we add another node, $(0', k)$, representing a zero fuel level at stop k, but a train cannot make any more stop. This will allow us to completely keep track of the number of stops a train has been made, and hence ensure that a train does not make more than two stops. Let N denote the set of all nodes.

Each arc from node (k_1', k_1) to node (k_2', k_2) in this network formulation represents a fuel plan where k_1 and k_2 are two consecutive refueling stops without any other stops in between. Now we consider incoming arcs to each node (k_2', k_2). If k_2 is an intermediate stop of a train, then the previous two nodes together cannot be intermediate stops of the same train; otherwise a train will make more than two intermediate stops. Specifically, a previous node (k_1', k_1) must not be contained in the set $\Delta^j_{k_2} = \{(k', k) \in N \mid k \in \Gamma^j_{k_2} \cap L(k_2), k' \in L(k_2) \cup \{0'\}\}$, where $L(k_2)$ is a set of intermediate stops of the same train prior to k_2. The incoming arcs to node (k_2', k_2) are created as follows:

- If k_2' is 0 and k_2 is an origin stop, then the locomotive must depart, with the minimal amount of fuel, from any stop $k_1 \in \Gamma^j_{k_2}$. Mathematically, the set of incoming arcs is given by $\{((k_1', k_1), (0, k_2)) \mid (k_1', k_1) \in N, k_1 \in \Gamma^j_{k_2}\}$
- If k_2' is 0 and k_2 is an intermediate stop, then the locomotive must depart, with the minimal amount of fuel, from any stop $k_1 \in \Gamma^j_{k_2}$, and k_1 must not be an intermediate stop of the same train $L(k_2)$. Mathematically, the set of incoming arcs is given by $\{((k_1', k_1), (0, k_2)) \mid (k_1', k_1) \in N, k_1 \in \Gamma^j_{k_2} \setminus L(k_2)\}$
- If k_2' is 0', which implies that k_2 is an intermediate stop, the locomotive must depart, with the minimal amount of fuel, from any stop $k_1 \in \Gamma^j_{k_2}$. Additionally, k_1 must be an intermediate stop of the same

train $L(k_2)$, and (k_1', k_1) is not in $\Delta^j_{k_2}$. In other words, k_2 is the second intermediate stop of this train. Mathematically, the set of incoming arcs is given by $\{((k_1', k_1), (0', k_2)) \mid (k_1', k_1) \in N \setminus \Delta^j_{k_2}, k_1 \in \Gamma^j_{k_2} \cap L(k_2)\}$

- If $k_2' \in \Gamma^j_{k_2}$ and k_2 is an origin stop, then the locomotive must depart, with the maximal amount of fuel (i.e., locomotive tank capacity), from stop k_2'. Mathematically, the set of incoming arcs is given by $\{((k_1', k_1), (k_2', k_2)) \mid (k_1', k_1) \in N, k_1 = k_2'\}$

- If $k_2' \in \Gamma^j_{k_2}$ and k_2 is an intermediate stop, then the locomotive must depart, with the maximal amount of fuel, from stop k_2'. Also, (k_1', k_1) must not be in $\Delta^j_{k_2}$. Mathematically, the set of incoming arcs is given by $\{((k_1', k_1), (k_2', k_2)) \mid (k_1', k_1) \in N \setminus \Delta^j_{k_2}, k_1 = k_2'\}$

The cost associated with each arc from node (k_1', k_1) to node (k_2', k_2) comprises a fixed refueling cost for one additional stop at stop k_2 and a fuel purchasing cost at stop k_1. The amount of fuel purchased depends on the fuel levels represented by the origin and destination nodes. Importantly, fuel costs are adjusted according to the lagrangian multipliers. In particular, for given $j \in J$ and $k \in K_j$, the fuel cost at yard i becomes $C^f_i + \sum_{t \in T}(\beta^j_{k,t} u_{i,t})$ per gallon.

References

[1] Michael Held, Philip Wolfe, and Harlan P. Crowder. Validation of subgradient optimization. *Mathematical Programming*, 6:62–88, 1974. 10.1007/BF01580223.

[2] Seyed Mohammad Nourbakhsh and Yanfeng Ouyang. Optimal fueling strategies for locomotive fleets in railroad networks. *Transportation Research Part B: Methodological*, 44(8-9):1104 – 1114, 2010.