Abstract

A “less than truckload” (LTL) network organises the transport of small shipping volumes by truck between given depots. To be cost-efficient it is necessary to bundle and unbundle goods on their way, using depots as so-called hubs. Our aim is to develop a strategic plan which is cost-optimal for given average shipping volumes. We consider transshipment and transport costs; to give a realistic estimate of the economies of scale, we charge each truck on a specific route equally, whether it is full or (nearly) empty.

Real-sized problems become too hard for standard solvers so that we develop a combination of heuristic strategies (which can, in the end, be combined with solvers like CPLEX). We consider the problem in two flavours: MAPIT requires to transport unsplit goods from one depot to another, using at most two intermediate depots as hubs. IO-MAPIT furthermore considers the circulation of trucks.

Keywords: Heuristics, IP, Hub Location, LTL Logistics, Economies of Scale
1 Introduction

LTL networks form the main leg of many transport networks; they organise the carriage of (usually small) shipping volumes between given depots, often over large distances. To increase capacity utilization, the goods are usually not transported directly, but bundled and unbundled at specially equipped depots which we then call hubs. Maintaining these hubs generates costs (depending on the transshipment capacity which is provided) which have to be balanced with the transportation costs. A general discussion of hub location problems can, e.g., be found in [1].

In the following discussion we consider a set $D$ of $n$ depots distributed all over Europe with given average shipping volumes between them. All depots have the capability to act as hubs; we have to pay for the installed transshipment capacity and the actual turnover costs by a given factor $t$. We have two further practical requirements:

- Each shipping volume may be turned over at most twice (but may also be sent directly if this is cheaper).
- Each shipping volume may only use one route (so it cannot be split and transported on different routes between the same origin and destination).

Since the cost of using a truck hardly depends on its load, we measure the transportation costs by the number of trucks on each route. We consider only one type of truck. Our aim is to find a routing of all shipping volumes which is cost-optimal. It is not meant to be a vehicle routing plan on a day-to-day basis; it should provide the information for strategic decisions as the planning of hub capacities, truck numbers and truck circulation.

Mathematically, the decisions in the problem can be described by $n^4$ binary variables $\text{Route}(i, k, l, j)$ which indicate how the shipping volumes $w(i, j)$ are routed over two depots $k$ and $l$ ($k = l$ or $k = l = j$ represent one-stop or no-stop routes). We introduce integer variables $\text{Truck}(k, l)$ which describe the number of trucks on the connection $k \to l$, determined by the chosen Routes.

There are also other models which measure vehicles by integer variables and have only approximately $n^3$ variables (see e.g. [2,3]). These flow-based models allow splitting of the shipping volumes $w(i, j)$ and routing over an arbitrary number of hubs. Although these assumptions stem from rail and

---

1 This research is part of the DFG project CL 318/14-1 “Lenkung des Güterflusses in durch Gateways gekoppelten Logistik-Service-Netzwerken mittels quadratischer Optimierung”
2 Email: meier@itl.tu-dortmund.de
3 Email: Uwe.Clausen@iml.fraunhofer.de
air traffic and are not realistic for LTL, these 3-index-models can provide relaxations of our model; hence their solutions are lower bounds.

In contrast to classical models, our model is undoubtedly more realistic than just “discounting” the hub-hub connections by a fixed factor. But:

• For realistic \( n \) (in the range from 25 to 100) we get a huge number of integer variables, so that standard MIP solvers give up quickly.
• The LP relaxation is very poor. Actually, without any further restrictions, the LP solution is just given by transporting every \( w(i, j) \) directly and charging it with the appropriate fraction of a truck size. Hence it is difficult to get a useful lower bound. Some preliminary results to improve this situation are presented in 2.1.

Our approach is as follows. Starting from the “direct transportation” solution, we apply three different heuristic improvement strategies after each other (see section 3):

• Firstly, a tree-based method which quickly gives a medium-good solution
• Secondly, a “destroy and locally repair” strategy
• Thirdly, a strict descend to a local optimum

This approach produces a good solution to start an optimization in CPLEX; furthermore, it gives us information about the “quality” of the possible hubs. We use this information to restrict the number of possible Routes to those expected to be “good”. We never delete Routes belonging to our initial solution so that it is still valid in the restricted model. This restricted problem can then be given to CPLEX (see 3.3).

We will also shortly glance at the problem of truck circulation. In realistic scenarios every truck should eventually come back to its home depot. While it is easy to state this property in the mathematical description \((\text{incomingtrucks}(i) = \text{outgoingtrucks}(i) \text{ for } i \in D)\), it is difficult to include this behavior in the heuristics.

2 The models MAPIT and IO-MAPIT

In this section we define our model, which we call MAPIT: The “Multi-Allocation Problem with Integer Trucks”. For the set \( D \) of depots, we define (with \( i, j, k, l \) always from \( D \)):

• \( w(i, j) \) is the shipping volume (the truck capacity is set to 1).
• \( c(i, j) \) is the cost for one truck to get from \( i \) to \( j \). We assume the triangle
inequality for \(c(i, j)\) since every truck should always choose the cost-optimal connection between two given points.

Furthermore, we charge the transshipment costs with a factor \(t\). The most important variable is \(\text{Route}(i, k, l, j)\) which is binary and indicates the route for the shipping volume from \(i\) to \(j\) \((k\) and \(l\) may be equal and/or equal to \(j\) to indicate transport over less than two hubs). The \(\text{Routes}\) are used to calculate the variables \(\text{Transport}(i, j)\) for the actual transport volume and the integer variables \(\text{Truck}(i, j)\) for the number of trucks to be used. Then the model looks like this:

Minimize \[\sum_{i,j} \text{Truck}(i, j) \cdot c(i, j) + \sum_{i,j} (\text{Transport}(i, j) - w(i, j)) \cdot t\] (1)

\[
\text{Transport}(i, j) = \sum_{k,l} \text{Route}(i, j, k, l) \cdot w(i, l) + \sum_{k,l} \text{Route}(k, l, i, j) \cdot w(k, j) + \sum_{k,l} \text{Route}(k, i, j, l) \cdot w(k, l)
\] (2)

\[
\text{Truck}(i, j) \geq \text{Transport}(i, j)
\] (3)

\[
\sum_{k,l} \text{Route}(i, k, l, j) = 1
\] (4)

\[
\text{Transport}(i, j) \in \mathbb{R}_{\geq 0}, \quad \text{Truck}(i, j) \in \mathbb{Z}_{\geq 0}, \quad \text{Route}(i, k, l, j) \in \{0, 1\}
\] (5)

The model IO-MAPIT has one additional constraint:

\[
\sum_k \text{Truck}(k, l) = \sum_k \text{Truck}(l, k) \quad \text{for all } l
\] (6)

The objective function (1) consists of transportation costs (first part) and transshipment costs (second part). In constraint (2), we determine the total transport volume on each edge, consisting of three sums for the three (possible) parts of a \(\text{Route}\). The inequality (3) determines the number of trucks (from the transport volume). Finally, (4) indicates that only one \(\text{Route}\) is used. The additional requirement (6) keeps the number of trucks at each depot constant, meaning that we could handle the \(\text{Routes}\) with round-trips for each truck.

We want to remind the reader that although we did not mention the transshipment capacities in the model and did not introduce special “hub variables”, the costs for building and maintaining transshipment capacities at
each depot are included in the cost parameter $t$ assuming a simplified linear cost structure. More complex cost structures make the MIP model harder but could be easily incorporated into the heuristics explained in section 3.

2.1 LP relaxation, lower bounds and possible improvements

The optimal LP solution of MAPIT is given by direct transport for every commodity with the objective value $\sum_{i,j \in D} c(i,j) \cdot w(i,j)$. As most of our shipping volumes are smaller than one tenth of a truck, rounding the LP solution increases the objective function by a factor of ten to twenty. How can we improve this?

For every subset $I \subset D$ we can compute the minimal amount that must be shipped from $I$ to $D \setminus I$ using the given values $w(i,j)$. We then generate a lower bound for the number of trucks between these sets by rounding up the minimal shipping volume. Especially, if we divide the set of depots into 1 and $(n-1)$ depots or 2 and $(n-2)$ depots, these inequalities give the LP relaxation additional strength. Nevertheless, the LP relaxation is still weak.

A more fruitful approach could be derived from the 3-index-model (mentioned in the introduction) allowing splitting and arbitrary rerouting. If we further relax the scenario by replacing the truck cost step function by a piecewise linear function, we can drastically improve the branch-and-cut solvability. We aim to give numerically evidence for this in future publications.

3 The heuristics

Our initial heuristic consists of three steps which are detailed below. The first two use parallel (sometimes interacting) threads; we prefer between 4 and 8 threads since less would increase the risk of running into local minima and more would lead to significantly longer running times (as modern computers have about 4 to 8 CPU kernels).

3.1 The three steps

Tree heuristic

The tree heuristic is based on the additional assumption that the transport of all $w(i,j)$ for a fixed destination $j$ forms a “sending tree”. This means that shipping volumes for $j$ that meet at one hub will never be split again. Computationally, this has the advantage that we can represent Route by a data structure consisting of $n$ trees; hanging a subtree to a different node is an $O(1)$ operation but enables us to reroute a lot of traffic.
We use an improvement strategy: We start from an arbitrary solution (like the rounded LP solution), and search for edges where empty space is transported over large distances. These “bad edges” are checked in every of the $n$ trees: If we could save costs by avoiding the given edge we reroute the respective subtree. The cost estimates are mildly randomized to avoid local minima. The tree heuristic quickly produces reasonable solutions (e.g. less than 30% gap for $n = 60$) and provides the basis for the next step.

**Local-Global heuristic**

We call a single route $i \rightarrow k \rightarrow l \rightarrow j$ for a shipping volume $w(i,j)$ locally optimal, if changing just this route cannot decrease the overall costs. Furthermore, in our given solution we calculate individual costs for each shipping volume $w(i,j)$, meaning the costs we would save by not transporting $w(i,j)$.  

The general idea of the Local-Global heuristic is now given by repeating the following two steps:

(i) Remove shipping volumes with high individual costs from the solution; furthermore, remove some arbitrarily chosen shipping volumes.

(ii) Find new routes for the removed shipping volumes (one after the other); the routes are chosen to be locally optimal (in the sense defined above). Since our routes have at most three edges, we can find such a route in $O(n^2)$, but a clever saving algorithm for already computed values can drastically speed up the procedure in the average case.

We do not only save the best solutions but maintain a solution pool. Unsuccessful threads choose from this pool to keep up with the better threads.

**Local improvement**

In the last step we undertake minor improvements until our solution is locally optimal, i.e. changing only one route cannot decrease the overall costs. Coming from the strongly optimized solution of the local-global heuristic, this is usually fast. From the complexity point of view, this is identical to part 2 of the local-global heuristic applied to just one route in each step.

**3.2 IO-MAPIT**

Our heuristic approaches do not include the circulation of trucks. The easiest way to add this aspect is to take the heuristic solution of MAPIT and balance it with the help of a min-cost flow. In our test data, we got reasonable results with this approach but in general this may not be the case.
Instead of introducing “artificial shipping volumes”, which was difficult, we prefer to let CPLEX do this correction afterwards (see also section 4).

3.3 Giving the solution to CPLEX

If we take our heuristic result as initial solution for CPLEX we cannot expect CPLEX to improve it (for high \( n \)) since the gap is still high. Hence, we delete a lot of Routes from the model which are probably not helpful (our equipment can handle about 250,000 Routes, meaning that we need further restrictions if \( n > \sqrt{250,000} \approx 22 \)). We apply two strategies:

- Avoid Routes which are “long” compared to the direct connection.
- Ignore Routes which include turnover at hubs with low quality: By this we mean hubs at which nearly no transshipment happens and which are therefore probably far away from the main connections.

Together with our initial solution we give CPLEX the chance to apply its branch-and-cut strategies effectively. The numerical results are discussed below, also for IO-MAPIT.

4 Experiments

We use a real world data set for 60 depots, from which we derive scenarios \( I_n \) with \( n = 15, n = 30 \) and \( n = 60 \). The full data set \( (I_{60}) \) is the most interesting case since CPLEX completely fails on it and therefore heuristics become necessary. At first we test good parameter choice by running four different configurations over relatively short time periods (20 minutes). To incorporate statistical spread we made 10 tests each and give the average objective value together with the maximal deviation:

<table>
<thead>
<tr>
<th></th>
<th>Best found for ( I_{60} )</th>
<th>Short tree heuristic</th>
<th>Long tree heuristic</th>
</tr>
</thead>
<tbody>
<tr>
<td>High thread interaction</td>
<td>211,167 (±1.6%)</td>
<td>211,038 (±2.0%)</td>
<td></td>
</tr>
<tr>
<td>Low thread interaction</td>
<td>210,751 (±2.4%)</td>
<td>210,293 (±0.8%)</td>
<td></td>
</tr>
</tbody>
</table>

We see that the method is stable and only mildly depends on the choice of the parameters. For our long time experiments we combine 6 hours of heuristic with 6 hours of CPLEX as described in section 3.3. For \( I_{60} \), we show the development of the solution in figure 1. For the table we use the configuration described above. The “Pure CPLEX” indicates the result we get by applying CPLEX directly to our mathematical model (12h runtime). For IO-MAPIT
we took the same heuristic, corrected the result by a min-cost flow and gave it to CPLEX. Unfortunately, it is difficult to compare the results to those from the literature: The “usual” models consider linear (rather than truck-based) transportation costs with a discount factor for hub-hub-connections. Some papers from the Nineties (like [2]) have comparable models, but the computational results are outdated. The recent work [3] has a comparable model, but considers only problem sizes up to 100,000 decision variables (like our “Instance 15”), for which a gap of about 20% is encountered.

5 Conclusion

The experiments show that the heuristics have good strength for short and long running times. Our aim is to incorporate them into more complex models including time slices. Furthermore we need to raise the lower bounds to get a better theoretical understanding of the problem.

References

