Partial Benders decomposition for logistics network design

Simon Bélières¹, Mike Hewitt², Nicolas Jozefowiez³, Frédéric Semet⁴, Tom Van Woensel⁵

¹ CNRS, LAAS, 7 Avenue du Colonel Roche, 31077 Toulouse Cedex 4, France
² Quinlan School of Business, Loyola University, 16 E. Pearson Ave., IL, Chicago 60611, USA
³ LCOMS EA 7306, Université de Lorraine, Metz 57000, France
⁴ Université de Lille, CNRS, Centrale Lille, Inria, UMR 9189 - CRISTAL, F-59000 Lille, France
⁵ Eindhoven University of Technology, School of Industrial Engineering, P.O. Box 513, 5600 MB
  Eindhoven, The Netherlands

Key-words: Logistics, Service Network Design, Benders

1 Introduction

We model and solve a transportation problem faced by consolidation carriers in supply chains. A supply chain is composed of suppliers, hubs, and customers that repeatedly requests products. Thus the consolidation carrier task is to determine a cost-effective solution to move products from the suppliers to the customers. This is expressed by a transportation plan over a time horizon that describes extensively the transport of goods. To minimize costs, the transportation plan must consolidate as many shipments as possible and coordinate goods paths in order to share resources. As each path is characterized by physical locations and starting/arrival times, consolidation requires to synchronize both geographic and temporal components.

Computation of transportation plans has long been assisted by solving network design problems, in particular the Service Network Design Problem (SNDP) [5]. In SNDP, commodities must be routed through a network from sources to sinks, by means of services route enabling transportation. Therefore, decisions must be taken concerning both services selection and traffic distribution. A lot of effort has been devoted to the SNDP, which is recognized as an highly complex problem. Unfortunately, exact methods still have difficulties with realistic-sized instances [2]. And in that sense, one major barrier for solving real-world sized network design problems is the number of commodities. As this parameter defines partly both the model size and the consolidation opportunities, it has a significant impact on computational tractability. To such an extent that only heuristic algorithms have proved to be effective on instances with a large number of commodities [4].

We consider an extension of the SNDP, the Logistics Network Design Problem (LNDP). The difference lies in the demand management. In the SNDP, one intent to flow goods with fixed origins and destinations (i.e. parcel transport). In the LNDP goods destinations are also known in advance, but the demands can be served by multiple potential origins. Therefore the solver must determine which origin is the most profitable to serve each demand. This is specific to supply chains, in which different suppliers can answer the same request. We then develop an effective partial Benders decomposition [1] that performs despite the increase of products. To improve convergence, the master problem is strengthened with information induced from aggregated products. We accelerate the Benders scheme with valid inequalities and heuristics solutions, and demonstrate the algorithm efficiency with an extensive computational study.

To summarize, the main contributions of the paper are (1) the introduction, to the best of our knowledge, of a new Logistics Network Design Problem (2) the development of a Benders scheme for efficiently solving this problem, and (3) demonstrating that aggregation of commodities/products can be an efficient opportunity to solve large-scale network design problems.
The remainder of the paper is organized as follows. In Section 2, we introduce the formulation of the Logistics Network Design. In Section 3, we present the Benders scheme and detail its acceleration techniques. In Section 4, we present and interpret the results of an extensive computational study of the algorithm performance. Finally, in Section 5, we finish with conclusions and a discussion of future work.

2 Logistics Network Design formulation

Let $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ be a network modelling the supply chain. Actors such as suppliers, warehouses and customers are represented by the node set $\mathcal{N}$. The directed arc set $\mathcal{A}$ models the network connectivity. Associated with each arc $a = (i, j) \in \mathcal{A}$ is a travel time $t_{ij} \in \mathbb{N}^*$, a per-unit-of-flow cost $c_{ij} \in \mathbb{R}^{+*}$ and a fixed cost $f_{ij} \in \mathbb{R}^{+*}$.

The supply chain consists in three layers. Suppliers $s \in \mathcal{S}$ produce and ship products $p \in \mathcal{P}$, therefore they only have outcoming arcs. Those products are requested by customers $c \in \mathcal{C}$ that exclusively have incoming arcs. At the interface between suppliers and customers, warehouses $w \in \mathcal{W}$ can store and consolidate freights. Figure 1 depicts a supply chain example.

Each supplier $s \in \mathcal{S}$ manufactures a subset of products $\mathcal{P}^s \subseteq \mathcal{P}$. If $p \in \mathcal{P}^s$, the supplier $s$ is qualified to satisfy any demand of $p$ in the supply chain. A supplier manufacturing a given product can ship any quantity of it, at any time. Given a time horizon $\mathcal{T}$ and a priori fixed customers demands $d^c_{it}$, our objective is to design the minimal-cost transportation plan that satisfies all customers demands. The design of a transportation plan consists in setting flows of goods such that every customers demands are satisfied, with enough vehicles on service routes to enable transportation.

To deal with the time aspect, it is possible to extend the flat network to a time-expanded network $\mathcal{G}_T = (\mathcal{N}_T, \mathcal{H}_T \cup \mathcal{A}_T)$. The graph $\mathcal{G}_T$ is built as follows. Each physical node $n_i \in \mathcal{N}$ is duplicated $|\mathcal{T}|$ times. Therefore the set $\mathcal{N}_T$ is formed of a couple $(i, t)$ for each $n_i \in \mathcal{N}$ and $t \in \mathcal{T}$. An holding arc in $\mathcal{H}_T$ represents the storage at a warehouse. For each $n_j \in \mathcal{W}$ and each $t \in [1, |\mathcal{T}|-1]$, there is a time-expanded arc $((i, t), (i, t+1))$ in $\mathcal{H}_T$ with a per-unit-of-flow cost $c_{ij}$ equal to the storage cost of the warehouse $n_j$. The transportation time-expanded arcs of $\mathcal{A}_T$ are built from the arcs in $\mathcal{A}$ as follows. For each $(n_i, n_j) \in \mathcal{A}$ and each time $t \in \mathcal{T}$ such that $t + t_{ij} < |\mathcal{T}|$, we build a time-expanded arc $((i, t), (j, t + t_{ij}))$ in $\mathcal{A}_T$ permits to transit goods from $n_i$ to $n_j$, leaving at time $t$ and arriving at time $t + t_{ij}$.

We define LNDP($\mathcal{G}_T$) to be our Logistics Network Design Problem defined over a time-expanded network $\mathcal{G}_T$. The design integer variables $y^p_{ij}$ quantify the number of trucks on transportation arcs. Trucks have a single capacity $c$. The continuous variables $x^p_{ij}$ denote the flow of product $p$ along transportation/holding arcs. Given an arc $((i, t), (j, t'))$ from a supplier $i$, there is no variable $x^p_{ij}$ if $i$ does not manufacture $p$, i.e. if $p \notin \mathcal{P}^i$. LNDP($\mathcal{G}_T$) seeks to minimize:

$$z(x, y) = \sum_{A_T} f_{ij} y^p_{ij} + \sum_{A_T} \sum_{p \in \mathcal{P}} c_{ij} x^p_{ij} + \sum_{A_T} \sum_{p \in \mathcal{P}} c_{ij} x^p_{ij}$$

Under the following constraints:
\[
\sum_{A_T \cup H_T} x_{ij}^{ptt} - \sum_{A_T \cup H_T} y_{ij}^{ptt} = 0, \quad \forall (j, t) \in W_T, \forall p \in P, \forall (i, t) \in N_T \tag{2}
\]
\[
\sum_{A_T \cup H_T} x_{ij}^{ptt} \geq d_{jt}^p, \quad \forall (j, t) \in C_T, \forall p \in P, \forall (i, t) \in N_T \tag{3}
\]
\[
\sum_{p \in P} x_{ij}^{ptt} \leq cy_{ij}^{ptt}, \quad \forall ((i, t), (j, t')) \in A_T \tag{4}
\]
\[
x_{ij}^{ptt} \in \mathbb{R}^+, \quad \forall ((i, t), (j, t')) \in A_T \cup H_T, \forall p \in P \tag{5}
\]
\[
y_{ij}^{ptt} \in \mathbb{N}^+, \quad \forall ((i, t), (j, t')) \in A_T \tag{6}
\]

LNDP($G_T$) seeks to minimize the sum of fixed costs on transportation arcs (first term), variable costs on transportation arcs (second term), and variable costs on holding arcs (third term), i.e., holding costs. The first two constraints ensure the flow feasibility. Constraint (2) enforces the flow conservation on each warehouse. Constraint (3) imposes the respect of each customers demands. Constraint (4) ensures that enough trucks are dispatched to transport products. Constraints (5) and (6) define the variable domains.

3 An enhanced Benders strategy

In this section, we detail an algorithmic strategy for solving the LNDP (1)-(6). Our method is an accelerated Benders strategy.

3.1 Partial Benders decomposition

The Benders algorithm is designed to tackle problems with complicating variables. It consists in decomposing the problem into a master problem and a subproblem that are solved iteratively. The master problem takes first-stage decisions by assigning values to a subset of variables. Given these decisions, the subproblem computes values of the remaining variables in order to obtain the best possible solution to the complete problem. At each iteration, a Benders cut from the dual subproblem is added to the master problem, ensuring the algorithm convergence.

In a classical Benders decomposition, the master problem and subproblem are obtained by projecting the complete problem respectively onto the subspace of complicating and non-complicating variables. When applied to the LNDP, this straightforward approach results in a master problem that assigns an integer value to every truck variable. Therefore, vector $y$ expresses the number of trucks allocated on each transportation arc. The master problem objective function minimizes trucks fixed costs. The only constraints are Benders cuts added dynamically. Given first-stage decisions, the linear subproblem seeks to satisfy customers demands by computing values for variables $x_{ij}^{ptt}$. As in the LNDP, flow feasibility is ensured by constraints (2) and (3). Moreover on each transportation arc, the quantity of products transiting cannot exceed the allocated capacity $y_{ij}^{ptt}$. The subproblem function minimizes flows linear costs.

It has been recognized that this straightforward approach is generally weak, essentially because the master problem and subproblem are unbundled. Therefore it is unlikely that the master problem, only driven by unstructured Benders constraints, can compute high-quality first-stage decisions in the first iterations. This results in a very slow algorithm convergence.

Crainic et al. introduced a more sophisticated decomposition that improves Benders strategies performance. The Partial Benders Decomposition strengthens the master problem adding explicit information from the subproblem, by retaining or creating scenarios. This allows
to articulate the master problem first-stage decisions, and accelerate the algorithm convergence. To enhance our straightforward decomposition, we construct an artificial information from the subproblem, and add it to the master problem. In that sense, we consider a 'super-product' \( \chi \) that merges all the products \( p \in P \). Therefore for every customer \((c, t) \in C_T\), the demand of 'super-product' is obtained by aggregating the demand of all products: \( D^\chi_{ct} = \sum_{p \in P} d^p_{ct} \).

Also for each flow variable \( x^{\chi tt'}_{ij} \) in the LNDP, we now consider a corresponding super-product variable \( x^{\chi tt'}_{ij} \). Figures 2 and 3 illustrate an example, respectively before and after merging the products. Note that aggregating induces a loss information, as we cannot restrict suppliers to ship only products they manufacture. In the aggregated version, each supplier does manufacture the super-product.

![Figure 2 – Original problem](image)

![Figure 3 – Aggregated problem](image)

To improve quality of the first-stage decisions, we supplement the master problem with a super-product flow component. Continuous variables \( x^{\chi tt'}_{ij} \) denotes the flow of super-product \( \chi \) along transportation/holding arcs. Our enhanced master problem allocates trucks on transportation arcs such that customers demands of super-products are satisfied. It is formulated as the following :

\[
\min \sum_{A_T} f_{ij} y_{ij}^{tt'} + z
\]

\[
\sum_{A_T \cup H_T} x^{\chi tt'}_{ij} - \sum_{A_T \cup H_T} x^{\chi tt'}_{ij} = 0, \quad \forall (j, t) \in W_T, \forall (i, t) \in N_T
\]

\[
\sum_{A_T \cup H_T} x^{\chi tt'}_{ij} \geq D^\chi_{ct}, \quad \forall (j, t) \in C_T, \forall (i, t) \in N_T
\]

\[
\sum_{p \in P} x^{\chi tt'}_{ij} \leq c y_{ij}^{tt'}, \quad \forall((i, t), (j, t')) \in A_T
\]

\[
0 \geq \rho (b - F_y), \quad \forall \rho \in \Omega
\]

\[
z \geq \pi (b - F_y), \quad \forall \pi \in \Gamma
\]

\[
x^{\chi tt'}_{ij} \in \mathbb{R}^+, \quad \forall((i, t), (j, t')) \in A_T \cup H_T
\]

\[
y_{ij}^{tt'} \in \mathbb{N}^+, \quad \forall((i, t), (j, t')) \in A_T
\]

\[
z \in \mathbb{R}^+
\]


Our enhanced master problem is a relaxation of the LNDP, and therefore is operable in a Benders decomposition. To improve our Benders strategy we also implement acceleration techniques described in the following subsections.
3.2 Polyhedral approach

When considering a super-product, we cannot restrict suppliers to ship only products they manufacture. This loss of information is illustrated in figures 2 and 3. In the original problem, the only way to satisfy products demands of customer \( c \) is to ship one unit of \( p_1 \), \( p_2 \) and \( p_3 \) respectively from \( s_1 \), \( s_2 \) and \( s_3 \). Therefore at least one truck must be allocated on each arc to enable shipments. In the aggregated problem each supplier can ship the super-product. If trucks capacity is greater or equal to three, the optimal solution to satisfy \( c \) demand of super-product is to ship three units of \( \chi \) from a single supplier. In that case, only a single truck is allocated. Since more trucks are required to satisfy products demands of customer \( c \), that first-stage decision leads to an unfeasible subproblem.

The loss of information induced from aggregating products can lead the master problem to have an optimal solution which induces an infeasible subproblem. Thus, to try and prevent that from happening, we supplement the master problem with three valid inequalities that render infeasible such solutions to the master problem. In this paper, we only describe our first valid inequality.

![Figure 4 – Valid Inequality 1 : Example](image)

**Valid Inequality 1** We illustrate this valid inequality with a static network, but it has a natural analog in a time-expanded network. In figure 4, the first image illustrates the original problem. \( p_1 \) and \( p_2 \) are manufactured respectively by suppliers \( S_1 \) and \( S_2 \). Customer \( C \) requires one unit of each product. There are transportation arcs \((S_1, C)\) and \((S_2, C)\), but the variable and fixed costs associated with \((S_1, C)\) are less than with \((S_2, C)\). Second image shows the optimal solution of the enhanced master problem, shipping two units of super-product from \( S_1 \) to satisfy \( C \) demand. The associated truck allocation leads to an unfeasible subproblem as the capacity from \( S_2 \) to \( C \) is null, which prevents to ship \( p_2 \) to \( C \).

We develop a valid inequality to cut such first-stage decisions from the enhanced master problem. For each product \( p \in P \) we supplement the network with a super-source \( SS_p \). For each supplier \((S, t)\) that manufactures \( p \), we build an arc \(((SS_p),(S, t))\) with null transit time, linear cost and fixed cost. The idea is to build an artificial layer of nodes to articulate the flow of super-product. For each product, we compute the total demand by summing demands of every customers over time : \( D_p = \sum_{(c, t) \in C_T} d_{ct} \). We then enforce each super-source \( SS_p \) to ship an amount of super-product superior or equal to \( D_p \), and impose a flow conservation on suppliers. Formally, we add the following constraints to the master problem :

\[
\sum_{A_T} x^{0t}_{SS_p} \geq D_p, \quad \forall p \in P, \forall (j, t) \in N_T
\]

\[
\sum_{A_T} x^{j't'} \chi_{ij} - \sum_{A_T} x^{j't'} \chi_{ij'} = 0, \quad \forall (j, t) \in S_T, \forall (i, t) \in N_T
\]
Total demands of $p_1$ and $p_2$ being of one unit for each product, we enforce both $SS_1$ and $SS_2$ to ship at least one unit of super-product. Therefore $S_1$ and $S_2$ receive a unit of super-product respectively from $SS_1$ and $SS_2$ (cuts illustrated in third image). Because we imposed flow conservation on suppliers, $S_1$ and $S_2$ both must ship an unit of super-product. The new enhanced problem optimal solution (see fourth image) allocates one truck one both $(S_1, c)$ and $(S_2, c)$, which results into a feasible subproblem. Therefore, cuts (16) and (17) prune low quality first-stage decisions and accelerate convergence.

### 3.3 Heuristic solutions

In a Benders strategy, primal solutions arise only from subproblem solutions. Therefore the incumbent can improve only when the subproblem is feasible. To improve the algorithm convergence, we generate heuristic primal solutions from well-chosen unfeasible subproblems. A subproblem is unfeasible if the number of trucks allocated $y$ is not sufficient to flow products.

Whenever a subproblem is unfeasible, we solve the same problem with slack variables $s$ on demand constrains $[(3)]$, and get a solution $(\hat{x}, \hat{s})$. Due to the slack variables prohibitive costs, this program seeks to serve as many demands as possible via flow variables $x$. Demands served by slack variables $s$ indicate that $y$ lacks of capacity. We compute the percentage of customer demands that cannot be met with the allocation $y : r = \frac{\sum_{(t,T) \in T} \sum_{p \in P} s_{pt}}{\sum_{(t,T) \in T} \sum_{p \in P} p_{pt}}$. This measure is our indicator of how “close” the allocation of vehicle capacities, $y$, is to inducing a feasible subproblem. If it is lower than a threshold value $r$, we repair $y$ and get a heuristic solution.

To do so, we use an iterative process starting from solution $(\hat{x}, \hat{y})$. We rank in decreasing order the demands that are served by slack variables. For each demand, we route the product with a linear program using slope-scaling to approximate trucks fixed cost, and update solution $(\hat{x}, \hat{y})$. In the end, we obtain an primal solution for the original problem. If its objective is lower than the incumbent, it replaces it.

### 4 Computational study

The algorithm was tested on a series of benchmark from a random generator. The latter is directly inspired from the context of our industrial collaborator. To build an instance, we position nodes $n \in N$ in a squared area of size $100 \times 100$. To ensure the feasibility, we set arcs such that it exists a path from each supplier to each customer. Additional arcs are added according to a radius $\alpha$. Specifically, transportation arcs are added to $A$ for pairs of locations that are less than $\alpha$ units apart. Travel times and truck fixed costs are proportional to arc lengths. Variable costs for loading and unloading onto the trucks are independent of the arcs or the products.

For our experiments, we generate instances with a number of days $D = \{30\}$, $|P| = \{100, 200, 300, 400, 500\}$, $|N| = \{50\}$, time intervals $\Delta = \{12h, 8h, 6h\}$, and $\alpha = \{10, 30\}$. There are 30 possible combinations and for each combination 5 randomly instances are generated. Therefore, we have 150 instances. In Figure 5 we report the complete program growth in terms of variables and constraints, as the number of products increases. Growth is normalised on instances with 100 products, which include an average of 718 156 variables and 111 720 constraints. We see that growth is substantial and quasi-linear, with a factor 5 increase for instances with $|P| = 500$.

To assess the efficiency of the proposed algorithm, we tested several methods on the instances detailed above. SPBD is the Partial Benders decomposition-based scheme, without valid inequalities and heuristic. SPBD3 embeds valid inequalities in the decomposition. SPBD3H is the complete strategy that also generates heuristic solutions.

As benchmarks we executed two other methods. The first method, CBD is the Classic Benders Decomposition, wherein none of the enhancements proposed in this paper are used. The second, CPLEX, is the CPLEX implementation of branch-and-cut. We initiate every
method with a heuristic solution \((x_h, y_h)\) obtained rounding-up every truck variables from the optimal solution of the LNDP linear relaxation. All algorithms were coded in C++ and executed on an Intel Xeon E5-2695 processor with 16 GB of memory under Linux 16.04. Linear and integer programs were solved using Cplex 12.7. All algorithms were executed with a stopping criteria of a proven optimality gap of 1% of less and a maximum run-time of 1.5 hours. For SPBD3H, The threshold parameter, \(\bar{r}\), was set to 20% after tuning.

In Figure 6 we benchmark SPBD3H against CBD and CPLEX by comparing optimality gaps at termination for each method. Results are averaged over instances with the same number of products \(|P|\). We observe that SPBD3H yields better gaps at termination, on average, than our two benchmarks for every set of instances. We also observe that the performance of CPLEX degrades as the number of products increases. On the other hand, SPBD3H remains effective for the largest instances.

Then, we study the impact of the super-product-based master problem, and the valid inequalities, on the lower bound produced at termination. To do so, we report in Table 1 the average lower bound reported by SPBD, SPBD3 and each of our two benchmarks at termination.

**TAB. 1 – Average lower bound reported at termination**

<table>
<thead>
<tr>
<th>Method</th>
<th>CBD</th>
<th>CPLEX</th>
<th>SPBD</th>
<th>SPBD3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1031.30</td>
<td>120316.48</td>
<td>187064.79</td>
<td>291930.02</td>
</tr>
</tbody>
</table>

We see that CBD yields a very weak lower bound. SPBD produces a stronger lower bound, one that is 26.39% greater, on average, in value than the bound produced by the best benchmark (CPLEX). Thus, we conclude that the Benders decomposition scheme based on the super-product master problem is superior to the benchmark methods with respect to the lower bound produced at termination. However, amongst these methods, the best results are obtained with SPBD3, indicating that all three valid inequalities, together, are the most effective.

We next analyze the impact the proposed heuristic has on the ability of SPBD3H to produce high-quality primal solutions. To do so, we measure for an instance and a method the improvement in the primal solution over that of the initial heuristic solution, \((x_h, y_h)\), by computing the primal gap:

\[
\text{primal-gap}_{\text{Method}} = \frac{z(x_h, y_h) - \text{UB}_{\text{Method}}}{z(x_h, y_h)} \times 100
\]

Here, \(\text{UB}_{\text{Method}}\) represents the objective function value of the best primal solution found by the method \textit{Method} during its execution. We benchmark SPBD3H against CPLEX and SPBD3. In Figure 7 we compare the upper bound performance of CPLEX, SPBD3 and SPBD3H. Results are averaged over instances with the same number of products \(|P|\). On the left image is displayed the percentage of instances wherein primal-gap_{UB}^{\text{CPLEX}} > 0..
The right image shows the average value of primal-gap\textsuperscript{\textit{CPLEX}}\textsubscript{\textit{UB}}, averaged over instances wherein \( \text{primal-gap}_{\text{CPLEX}} > 0 \).

![Figure 7 – Upper bounds comparison](image)

We see that both \textit{CPLEX} and \textit{SPBD123} struggle to produce an improved primal solution, particularly for instances with more than 200 products. \textit{SPBD123H}, however, is often able to produce an improved primal solution. That said, the relative magnitude of the improvement is fairly small.

5 Conclusion

We have presented a Benders strategy based on products aggregation, to solve the Logistics Network Design Problem. Computational study has demonstrated that the current implementation of our method is quite effective. In particular, our Benders strategy shows resilience when the number of products increases. As the number of product/commodities is one major barrier for solving real-world sized network design problems, we suppose and hope that underlying concepts of our approach will be adapted to other problematics.

To solve the LNDP, a column generation approach could be implemented through efforts. A straightforward strategy, where a column represents a path flow a given request, is unlikely to provide good results for several reasons. One of the main reasons is that the number of request is substantial. Also a column is not able to express the cost savings introduced by the consolidation. Other not straightforward decompositions could lead to other interesting models. It is a possible perspective to solve the LNDP.

Références


