The Transshipment Problem

Optimization Methods in Management Science
Master in Management
HEC Lausanne

Dr. Rodrigue Oeuvray

Fall 2019 Semester

The Transshipment Problem

- Formulation
- Basic solution and tree solution
- The simplex algorithm applied to this problem
- Totally unimodular matrix

Formulation

We consider a network $R = (V, E, b, c)$ where

- $G = (V, E)$ is a **simple** directed and **connected** graph
- $b : V \to \mathbb{R}$ is a **weighting of the vertices** of G representing the supply (a negative value) or the demand (a positive value) at each vertex. The weighting at a vertex can also be null
- $c : E \to \mathbb{R}$ is a **weighting of the arcs** of G representing the unit cost of using them

We assume that the graph is connected to be sure that there exists a chain (not a path !) between any vertex

Formulation (Cont'd)

Problem

We would like to find a **flow** $x : E \to \mathbb{R}_+$ of quantities to ship along the arcs in order to satisfy the supply and demand at each vertex with a minimal cost

The fact the graph is connected does not guarantee that there is a feasible solution to this problem !

Formulation (Cont'd)

- Let $x : E \to \mathbb{R}_+$ be a flow, total shipping costs are $z = \sum_{(i,j)\in E} c_{ij}x_{ij}$
- The equilibrium is satisfied if the difference bewteen entering and exiting quantities at each vertex is equal to the supply or the demand at that vertex

$$\sum_{j \in Pred(i)} x_{ji} \quad - \quad \sum_{j \in Succ(i)} x_{ij} \; = \; b_i \qquad \forall\, i \in V$$

- If $b_i < 0$, this a **source**; if $b_i = 0$, then this is a **transshipment** vertex. Finally, if $b_i > 0$, this is a **sink**
- A necessary condition to have an equilibrium between demand and supply is $\sum_{i\in V} b_i = 0$. From now on, we will assume that this assumption is always satisfied

Transhipment Problem: LP Formulation

The problem consisting in determining a transshipment planing satisfying supply and demand at each vertex of R with a minimal total cost can be formulated as a LP:

$$
\begin{array}{llrcll}
\text{Min} & z = & \displaystyle\sum_{(i,j)\in E} c_{ij} x_{ij} & & & \\
\text{s.t.} & & \displaystyle\sum_{j\in Pred(i)} x_{ji} - \sum_{j\in Succ(i)} x_{ij} & = & b_i & \forall\, i \in V \\
 & & x_{ij} & \geq & 0 & \forall\, (i,j) \in E
\end{array}
$$

Dual Problem

$$
\begin{array}{llcccll}
\text{Max} & w = & \sum_{i \in V} b_i y_i & & & & \\
\text{s.t.} & & y_j & - & y_i & \leq & c_{ij} & \forall\, (i,j) \in E \\
 & & & & y_i & \in & \mathbb{R} & \forall\, i \in V
\end{array}
$$

Economic interpretation of the dual problem:

- A production company (company A) hires a logistics company (company B) to handle the transportation business
- Company B buys all the products from the different factories and sells them back to the warehouses
- y_i is the unit price received/paid at vertex i to satisfy its demand/supply
- The objective of company B is to maximize its profit
- $y_j - y_i \leq c_{ij}$ means that company B needs to be competitive with the costs c_{ij} that would incur to company A if it would have to organize the logistics

Reminder: the demand is positive

Formulation in Matrix Form

In matrix form, the transshipment problem and its dual can be written as:

$$
\begin{array}{lrcl}
& \multicolumn{3}{c}{\text{(PLP)}} \\
\text{Min} \quad z = & \boldsymbol{cx} & & \\
\text{s.t.} & \boldsymbol{Ax} & = & \boldsymbol{b} \\
& \boldsymbol{x} & \geq & \boldsymbol{0}
\end{array}
\qquad\qquad
\begin{array}{lrcl}
& \multicolumn{3}{c}{\text{(PLD)}} \\
\text{Max} \quad w = & \boldsymbol{yb} & & \\
\text{s.t.} & \boldsymbol{yA} & \leq & \boldsymbol{c} \\
& \boldsymbol{y} & \in & \mathbb{R}^n
\end{array}
$$

where $\boldsymbol{A}$ is the incidence matrix

Basic Solution and Tree-Solution

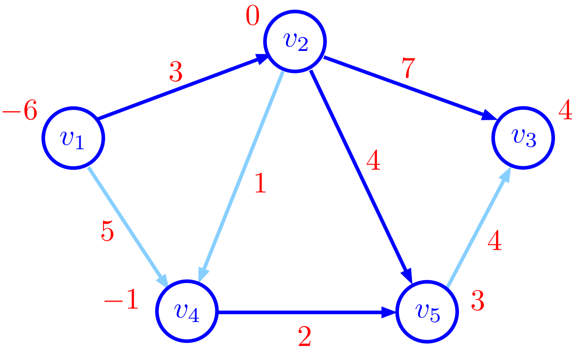
- For a connected graph containing n vertices, the rank of its incidence matrix is $n - 1$
- For a connected network with n vertices, choosing a basis of the space of the columns of $\boldsymbol{A}$ is equivalent to selecting $n - 1$ columns of $\boldsymbol{A}$ or equivalently $n - 1$ variables x_{ij}
- In this network, these variables form a **spanning tree**. A spanning tree of a graph G is a partial graph that is a tree which includes all of the vertices of G

Basic Solution and Tree-Solution (Cont'd)

- In the transshipment problem, the **basic** solutions of the LP and the **spanning trees** of the network are in bijection
- In the network corresponding to a transshipment problem, a spanning tree T is called a **tree-solution**. Only the arcs of T are used for the shipping
- The concepts of feasibility, non-feasibility, optimality, degeneracy, unboundedness are also valid in this network

Example of a Tree-Solution

The spanning tree $E_T = \{(v_1, v_2), (v_2, v_3), (v_2, v_5), (v_4, v_5)\}$ is a **tree-solution**



It corresponds to choosing $x_{12}, x_{23}, x_{25}, x_{45}$ as a basis of the matrix $\boldsymbol{A}$

The Transshipment Simplex Algorithm (Phase II)

Input: a **connected** network $R = (V, E, b, c)$, $|V| = n$, $|E| = m$ and a **feasible** tree-solution $T = (V, E_T)$.

Output: a **flow** $x : E \to \mathbb{R}_+$ of minimal total cost or the proof that this flow does not exist

(1) Computations of the primal $\boldsymbol{x}$ and dual $\boldsymbol{y}$ solutions associated to T

(2) Search for an entering arc:

If it does not exist: STOP. Actual solutions are optimal

(3) Search for an exiting arc:

If it does not exist : STOP. The network has a circuit with a negative cost and the problem has no finite optimum

(4) Update of the tree-solution and back to point (1)

Primal Solution Associated With $T = (V, E_T)$

Input: a connected network $R = (V, E, b, c)$ and a tree-solution $T = (V, E_T)$

Output: the flow $x : E \to \mathbb{R}_+$ associated with T

(1) While $|E_T| > 1$ do

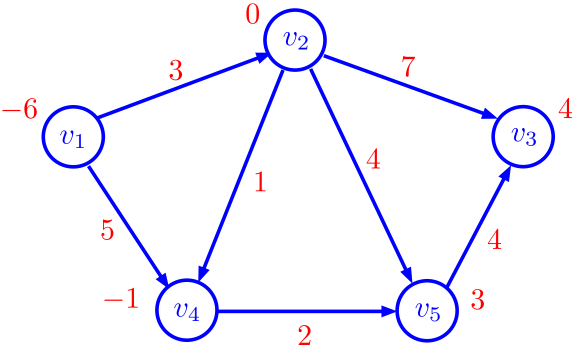
- (1.1) Find a vertex j of degree 1 [1] in $T = (V, E_T)$. Let $e \in E_T$ be the only arc incident with j and i its other endpoint
- (1.2) If $e = (i, j)$, set $x_{ij} = b_j$. Otherwise $e = (j, i)$ and set $x_{ji} = -b_j$
- (1.3) Set $b_i = b_i + b_j$
- (1.4) Remove e from E_T : $E_T = E_T \setminus \{e\}$

(2) It remains only one arc in E_T, let's say (i, j), set $x_{ij} = b_j$

[1] In a tree, a leaf is a vertex of degree 1

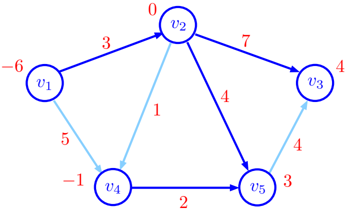
Example

Let's consider the following transshipment problem:



Example (Cont'd)

Let $E_T = \{(v_1, v_2), (v_2, v_3), (v_2, v_5), (v_4, v_5)\}$ be a tree-solution



- We select v_4. As $b_4 =$ -1, then $x_{45} = 1$ and we update $b_5 = 3 - 1 = 2$
- We select v_3. As $b_3 = 4$, then $x_{23} = 4$ and $b_2 = 0 + 4$
- We select v_5. As $b_5 = 2$, then $x_{25} = 2$ and $b_2 = 4 + 2 = 6$
- We select v_2. As $b_2 = 6$, then $x_{12} = 6$

The **basic** solution corresponding to this tree is $x_{45} = 1, x_{23} = 4, x_{25} = 2$ and $x_{12} = 6$ which is feasible and its cost is:

$$z = 1 \times 2 + 4 \times 7 + 2 \times 4 + 6 \times 3 = 56$$

Computation of the Dual Solution

- The incident matrix $\boldsymbol{A}$ of a connected network being of size $n \times m$ but of rank $n - 1$, we can remove arbitrarly a constraint of the problem
- As dual variables are associated with constraints, then it means that **one can fix one dual variable to zero**

Dual Solution Associated With $T = (V, E_T)$

Input: a connected network $R = (V, E, b, c)$ and a tree-solution $T = (V, E_T)$

Output: dual prices $y_i : V \to \mathbb{R}$ associated with T

(1) Choose arbitrarily $i \in V$ and set $y_i = 0$
Set $W = V \setminus \{i\}$

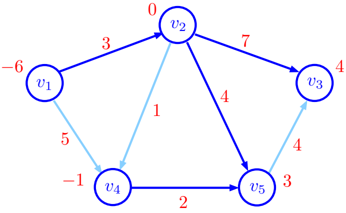
(2) While $W \neq \emptyset$ do

(2.1) Find an arc $e \in E_T$ with endpoints i and j with $i \in V \setminus W$ and $j \in W$
(2.2) If $e = (i, j)$ set $y_j = y_i + c_{ij}$. Otherwise ($e = (j, i)$) set $y_j = y_i - c_{ji}$
(2.3) Remove j from W: $W = W \setminus \{j\}$

Remark: with step 2.1, we have the guarantee that y_i has already been set

Example (Cont'd)

We consider the following tree $E_T = \{(v_1, v_2), (v_2, v_3), (v_2, v_5), (v_4, v_5)\}$



- We select v_1. Then $y_1 = 0$
- We select (v_1, v_2). Then $y_2 = 0 + 3 = 3$
- We select (v_2, v_3). Then $y_3 = 3 + 7 = 10$
- We select (v_2, v_5). Then $y_5 = 3 + 4 = 7$
- We select (v_4, v_5). Then $y_4 = 7 - 2 = 5$

The dual basic solution corresponding to this tree is $y_1 = 0, y_2 = 3,$ $y_3 = 10, y_4 = 5, y_5 = 7$ and has a cost of:

$$w = 0 \times (-6) + 3 \times 0 + 10 \times 4 + 7 \times 3 + 5 \times (-1) = 56$$

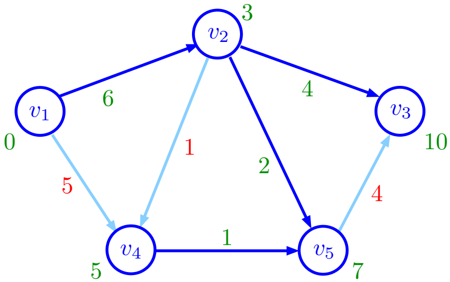
Search for an Entering Arc

We assume that we have a feasible tree-solution $T = (V, E_T)$ and its primal $\boldsymbol{x}$ and dual $\boldsymbol{y}$ solutions

- if the primal and dual solutions are feasible, then there are both **optimal** (weak duality)
- By construction, the dual constraints are satisfied (with equality) for the arcs of E_T. So we have to test if the dual constraints $y_j - y_i \leq c_{ij}$ associated with the **non**-**basic** arcs (i, j) are satisfied or not
- As soon as a constraint is **violated**, the corresponding arc **enters** the basis
- If all the constraints are satisfied, the actual primal and dual solutions are **optimal**

Example (Cont'd)

Let's continue with the previous example:



- Numbers beside the vertices are the dual variables. The green numbers beside the arcs are the primal variables. The red ones are the unit costs
- For the non-basic arc (v_1, v_4), we have $c_{14} = 5$ and $y_4 - y_1 = 5 - 0 = 5$. The constraint $y_4 - y_1 \leq c_{14}$ is **satisfied**
- For the non-basic arc (v_2, v_4), we have $c_{24} = 1$ and $y_4 - y_2 = 5 - 3 = 2$. The constraint $y_4 - y_2 \leq c_{24}$ is **violated**. The arc (v_2, v_4) enters the basis

Search for an Exiting Arc

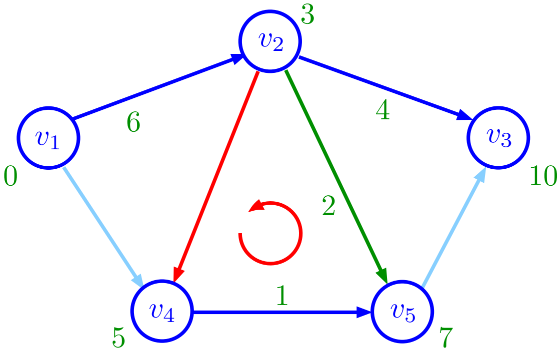
- If we add the entering arc $e = (i, j)$ to the current tree-solution, then it forms a **cycle** C
- Using the orientation of (i, j), we divide the arcs of C in two **disjoint** sets C^+ and C^-, where C^+ is composed of the arcs of C with the **same** orientation as (i, j) and C^-, the arcs with the opposite orientation
- If we change the flow along the cycle C by shipping an additional quantity Δ over the arcs of C^+ and $-\Delta$ over the arcs of C^-, then we do not change the totals at the vertices of C

Search for an Exiting Arc (Cont'd)

- If $C^- = \emptyset$, STOP: the circuit C has a **negative** cost (since $y_j - y_i \leq c_{ij}$) and the problem has **no finite optimum**
- Otherwise, compute $\Delta = \min\{x_{kl} \mid (k,l) \in C^-\}$ and s the arc corresponding to Δ. Then s exits the basis

Example (Cont'd)

- The entering arc is $e = (v_2, v_4)$



- The cycle C is formed by (v_2, v_4), (v_4, v_5) and (v_2, v_5).
  $C^+ = \{(v_2, v_4), (v_4, v_5)\}$ and $C^- = \{(v_2, v_5)\}$
- Consequently $\Delta = \min\{x_{kl} \mid (k, l) \in C^-\} = x_{25} = 2$ and $s = (v_2, v_5)$

Updating the Primal and Dual Solutions

- The new tree-solution is given by $E_T = E_T \cup \{e\} \setminus \{s\}$
- Only the quantities on the arcs of C change

$$x_{ij} = \begin{cases} x_{ij} + \Delta & \text{si } (i,j) \in C^+ \\ x_{ij} - \Delta & \text{si } (i,j) \in C^- \\ x_{ij} & \text{si } (i,j) \notin C \end{cases}$$

- The dual basic solution is recomputed

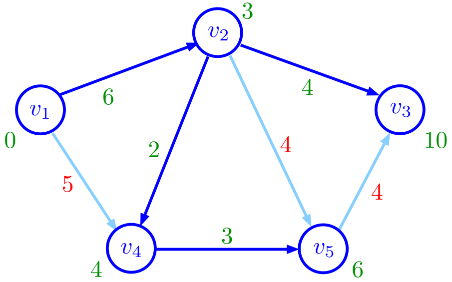
Example (Cont'd)

The **new** tree-solution is $E_T = E_T \cup \{(v_2, v_4)\} \setminus \{(v_2, v_5)\}$

$$x_{24} = x_{24} + \Delta = 0 + 2 = 2$$

$$x_{45} = x_{45} + \Delta = 1 + 2 = 3$$

$$x_{25} = x_{25} - \Delta = 2 - 2 = 0$$



Two **changes** for the new dual solution: 1) $y_4 = 4$, and 2) $y_5 = 6$

Degeneracy

- If the primal basic solution is **degenerated**, i.e. if it exists at least one arc in E_T with a null quantity, the modification Δ of the flow may be null and there is a risk of cycling
- To avoid this situation, we can use a network version of **Bland's rule**:
  - Test the non-basic arcs in the **lexicographical order** and the first arc whose dual constraint is violated enters the basis
  - If quantity Δ (which is null in the case of degeneracy) is shipped along several arcs of C^-, the smallest arc in the **lexicographical order** exits the basis

Computation of an Initial Feasible Tree-Solution

- When a **feasible** tree-solution is unknown or hard to determine, we need to use the **Phase I** of this algorithm
- Similarly to Phase I of the simplex algorithm, we define an **auxiliary** problem having:
  - always a feasible solution,
  - always a finite optimum,
  - a finite optimium with a zero value if and only if the initial problem has at least one feasible solution
- Moreover, it is easy to find a **feasible** tree-solution of the auxiliary problem and its optimal solution, if it has a **zero** value, provides a feasible tree-solution for the **initial** problem

Construction of the Auxiliary Problem

Input: a connected network $R = (V, E, b, c)$

Output: a network $R' = (V, E', b, c')$ and a feasible tree-solution $T' = (V, E'_T)$ for R'

(1) Set $c'_{ij} = 0$ for all $(i, j) \in E$.

(2) Choose a source, let's say k

(3) Connect each source i ($\neq k$) to k with an artificial arc (i, k) of weight $c'_{ik} = 1$ (if it already exists in R, do not add it)

(4) Connect k to each sank j with an artificial arc (k, j) of weight $c'_{kj} = 1$ (if it already exists in R, do not add it)

(5) Set $E'_T = \{(i, k) \mid i \text{ is a source}\} \cup \{(k, j) \mid j \text{ is a sink}\}$ and, if necessary, add some arcs to E'_T until we get a spanning tree

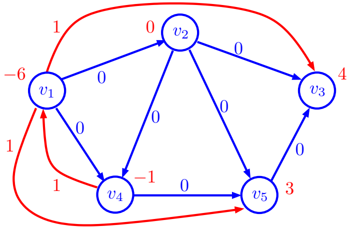
Construction of the Auxiliary Problem (Cont'd)

- **Interpretation**: one builds an artificial tree in which all the sources and sanks are connected to the main source k. By doing so, all the quantities from any source move in transit via k and are rooted to sanks

- The new objective function z' in R' consists in minimizing the total cost with the new weighting c'_{ij} of the arcs defined in R'

Example (Cont'd)

Here is the **auxiliary network** for our example:



- The initial tree-solution T is formed by the arcs of weight 1 and by one of the incident arcs to v_2, let's say (v_1, v_2)
- So E_T is given by $\{(v_1, v_3), (v_1, v_2), (v_4, v_2), (v_1, v_5)\}$
- The initial basic solution associated to R' is given by $x_{14} = 4, x_{12} = 0$, $x_{41} = 1, x_{15} = 3$ for a total cost of $z' = 8$
- This solution is degenerated

The Transshipment Simplex Algorithm: Phase I

Input: a **connected** network R

Output: a **feasible** tree-solution in R or a **certificate** that **no feasible** solution exists for the problem defined by R

(1) Construct the auxiliary network R'

(2) Solve the auxiliary problem with phase II of the transshipment simplex algorithm

- If $z' = 0$ at the optimum, remove the **artificial** arcs from R' to get a **feasible** tree-solution in R
- If $z' > 0$ at the optimum, then there is **no feasible** solution to the problem defined by R

Why Do We Always Get an Integer Solution ?

Question

We assume that the vector $\boldsymbol{b}$ of supply and demand is integer. Even though $\boldsymbol{b}$ is integer, there is, a priori, no reason why the optimal solution of this problem should be integer. So, why is it **always** the case ?

Why Do We Always Get an Integer Solution ? (Cont'd)

Let's have a look at the algorithm to compute a primal basic solution :

(1) While $|E_T| > 1$ do

  (1.1) Find a vertex j of degree 1 in $T = (V, E_T)$. Let $e \in E_T$ be the only arc incident with j and i its other endpoint

  (1.2) If $e = (i, j)$, set $x_{ij} = b_j$. Otherwise $e = (j, i)$ and set $x_{ji} = -b_j$

  (1.3) Set $b_i = b_i + b_j$

  (1.4) Remove e from E_T : $E_T = E_T \setminus \{e\}$

(2) It remains only one arc in E_T, let's say (i, j), set $x_{ij} = b_j$

Why Do We Always Get an Integer Solution ? (Cont'd)

During the algorithm :

- b_i is updated with the following formula : $b_i = b_i + b_j$
- As b_i and b_j are integer, then $b_i + b_j$ is also integer
- The variables x_{ij} are set either to b_j or -b_j
- In both cases, there are integer values

We conclude that the optimal solution built by this algorithm is **necessarily integer** !