The Network Simplex Algorithm

Presented by:

- Fares Bekhouche
- Fakherdinne seddiki
- Asma Gafour
- Mouna Benbettiche

Submitted to: Dr. Baha Alzalg

The University of Jordan
Department of Mathematics

Integer and Combinatorial Optimization (0301973)
Spring 2018

Introduction.

The network simplex method is an adaptation of the bounded variable primal simplex algorithm. The basis is represented as a rooted spanning tree of the underlying network, in which variables are represented by arcs, an entering variable is selected by some pricing strategy, based on the dual multipliers, and forms a cycle with the arcs of the tree. The leaving variable is the arc of the cycle with the least augmenting flow. The substitution of entering for leaving arc, and the reconstruction of the tree is called a pivot. When no non-basic arc remains eligible to enter, the optimal solution has been reached.

Presentation Plan I

1. Comments about the simplex algorithm
2. The network the simplex algorithm
3. Conclusion

1. Comments about the simplex algorithm
2. The network the simplex algorithm
3. Conclusion

Comments about the simplex algorithm

Each step is called a pivot.

- Pivots are carried out using linear algebra
- Pivots for network flow problems can be carried out directly by changing flows in arcs.

Typically, the simplex method finds the optimal solution after a small number of pivots.

The simplex algorithm is VERY efficient in practice.

1 Comments about the simplex algorithm

2 The network the simplex algorithm

3 Conclusion

Back to networks

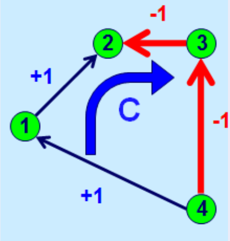
We will work with the arcs of the original network (not the residual network)
We will soon describe extreme flows.
Connection between spanning tree flows and extreme flows.

Networks : sending flow around a cycle

- Assume that cycles are oriented in a direction.
- The forward arcs of the cycle are the arcs in the same orientation.
- The backward arcs are in the opposite direction.
- A flow of 1 unit around C refers to a flow of 1 unit in the forward arcs and a flow of -1 units in the arcs in the backward arcs

Networks : sending flow around a cycle

Exemple



Arcs $(1,2)$ and $(4,1)$ are forward arcs.
Arcs $(3,2)$ and $(4,3)$ are backward arcs.

What is an extreme point solution of a network flow problem ?

Let x be a feasible flow for a minimum cost flow problem. An arc (i, j) is called free if $0 < x_{ij} < u_{ij}$. One can increase or decrease the flow in a free arc by a small amount and still satisfy bound constraints.

Theorem. A feasible flow x is an extreme point solution if and only if there is no (undirected) cycle of free arcs.

Basic feasible solutions

A basis structure consists of a spanning tree T, a set L of arcs, and a set U of arcs, such that $T \cup L \cup U = A$.
For each $(i, j) \in L, x_{ij} = 0$.
For each $(i, j) \in U, x_{ij} = u_{ij}$.
The arc flows in T are selected so that each node satisfies its supply/demand constraint.
The basis structure is feasible if the arc flows also satisfy the upper and lower bounds. (Not all basis structures are feasible.) The feasible flow is called basic.

Optimality Conditions for Spanning Tree Solutions :

The following are conditions under which x is an optimal solution for the minimum cost flow problem and y is optimal for the dual problem :

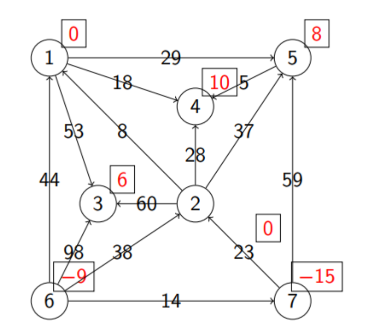
1. The basic flow x is feasible
2. p is the vector of simplex multipliers.
3. For each non-tree arc (i, j)

✓ if $c_{ij}^y > 0$, then $xij = 0$,
✓ if $c_{ij}^y < 0$, then $x_{ij} = u_{ij}$,
with

$$c_{ij}^y = c_{ij} - y_i + y_j.$$

Example of transport problems

Example of transport problems



Example of transport problems

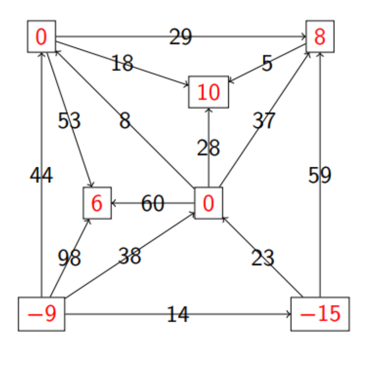
A node i such that $b_i < 0$ is a source.
A node i such that $b_i > 0$ is a well.
We have $\sum_i b_i = 0$.

Matrix formulation of the problem

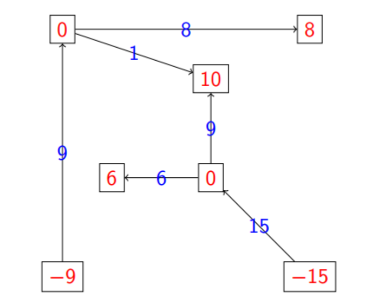
Minimize $\sum c_{ij}x_{ij}$ under the constraint $Ax = b,\ x \geq 0$, with A of size $n \times a$,
Where n is the number of nodes and has number of arcs and

$$
\begin{array}{ll}
A[k, ij] = -1 & \textit{if ij is an arc},\ k = i, \\
A[k, ij] = 1 & \textit{if ij is an arc},\ k = j, \\
A[k, ij] = 0 & \text{o.w.}
\end{array}
$$

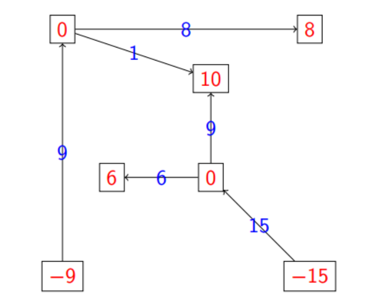
Example of transport problems



Example of transport problems



Example of transport problems



The first solution :

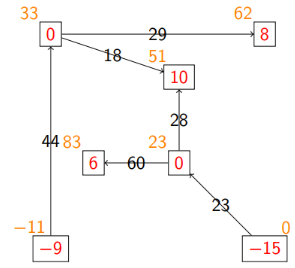
$$x_{14}^* = 1, x_1^*58 =, x_{23}^* = 6, x_{24}^* = 9, x_{61}^* = 9, x_{72}^* = 15, \ with\ z = 1603.$$

Example of transport problems

the network simplex Algorithm

• We start from a tree T covering which is a solution (when it exists one). We say we have a tree solution.
• We define a price vector $y = (y_1, \cdots, y_n)$ at a constant close to by

$$y_i + c_{ij} = y_j \text{ for all arc } ij \text{ of } T.$$



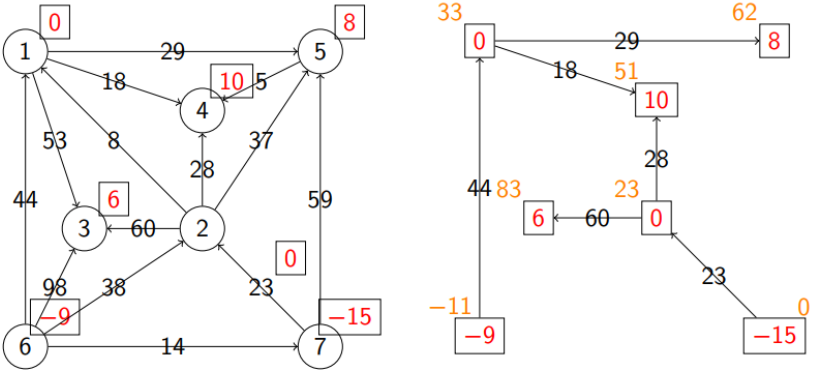
Example of transport problems

- We search for an incoming arc $e = uv$, $e \notin T$ such that

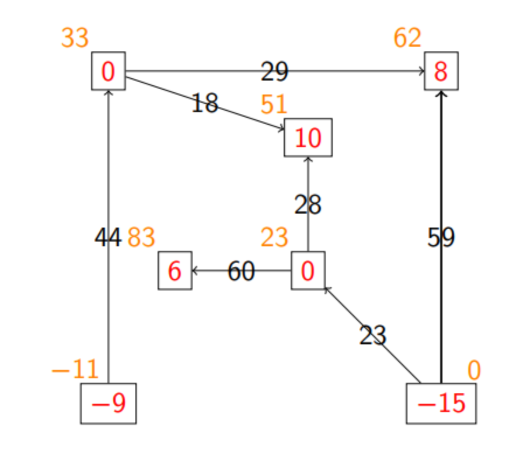
$$y_u + c_{uv} < y_v.$$

- $T + e$ creates a cycle. We search for an arc coming out f on the cycle. It is an opposite arc to e on the cycle and maximum flow.
- $T \leftarrow T + e - f$.
- Update T.
- Updating x, updating prices y.

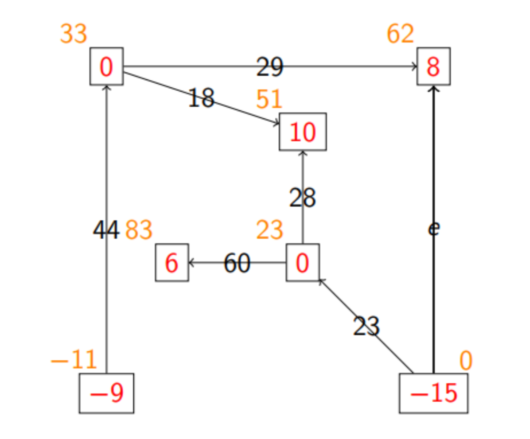
Example of transport problems



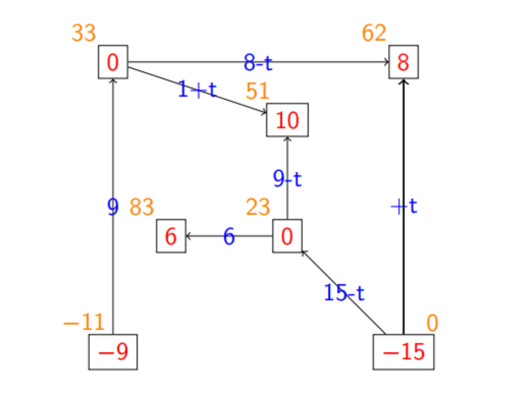
Example of transport problems



Example of transport problems

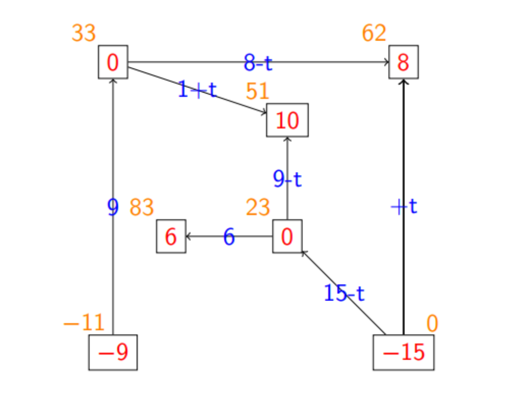


Example of transport problems



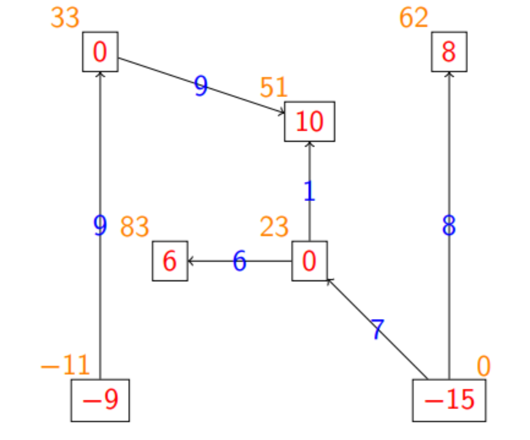
Example of transport problems

The arc (15) comes out.



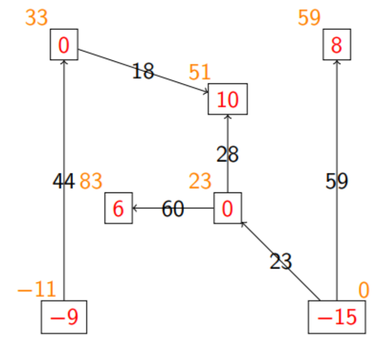
Example of transport problems

The arc (15) comes out.



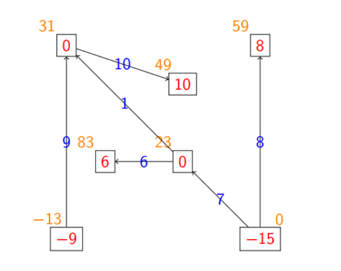
Example of transport problems

Price update.



Example of transport problems

New iteration



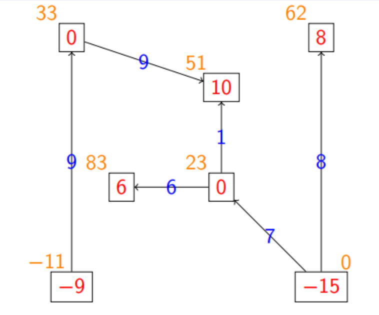
The third solution

$$x^*_{14} = 10, x^*_{23} = 6, x^*_{21} = 1, x^*_{61} = 9, x^*_{72} = 7, x^*_{75} = 8, with z = 1577.$$

Finally, the optimal solution is the second solution, we choose the min of them.

Example of transport problems

Finely the optimal solution is the second solution, we choose the min of them



The second solution

$$x^*_{14} = 9, x^*_{23} = 6, x^*_{24} = 1, x^*_{61} = 9, x^*_{72} = 7, x^*_{75} = 8, with\ z = 1183.$$

1. Comments about the simplex algorithm
2. The network the simplex algorithm
3. Conclusion

Conclusion

Conclusion

- Network simplex is extremely fast in practice.
- Relying on network data structures, rather than matrix algebra, causes the speedups. It leads to simple rules for selecting the entering and exiting variables.
- A good pivot rule can dramatically reduce running time in practice.

Thanks for your attention !