Notes for Operations Research & More

Lan Peng, PhD Student

Department of Industrial and Systems Engineering
University at Buffalo, SUNY
lanpeng@buffalo.edu

November 25, 2019

November 25, 2019

Contents

Chapter 1

Minimum Weight Flow Problem

1.1 Transshipment Problem

Transshipment Problem (D, b, w) is a linear program of the form

$$\begin{aligned} \min \quad & wx && (1.1)\\ \text{s.t.} \quad & Nx = b && (1.2)\\ & x \geq 0 && (1.3) \end{aligned}$$

Where N is a vertex-arc incident matrix. For a feasible solution to LP to exist, the sum of all bs must be zero. Since the summation of rows of N is zero. The interpretation of the LP is as follows.
The variables are defined on the edges of the digraph and that x_e denote the amount of flow of some commodity from the tail of e to the head of e
Each constraints

$$\sum_{h(e)=i} x_e - \sum_{t(e)=i} x_e = b_i \tag{1.4}$$

represents consequential of flow of all edges into k vertex that have a demand of $b_i > 0$, or a supply of $b_i < 0$. If $b_i = 0$ we call that vertex a transshipment vertex.

1.2 Network Simplex Method

Lemma 1.1. *Let C_1 and C_2 be distinct cycles in a graph G and let $e \in C_1 \cup C_2$. Then $(C_1 \cup C_2) \setminus e$ contains a cycle.*

Proof. Case 1: $C_1 \cap C_2 = \emptyset$. Trivia.
Case 2: $C_1 \cap C_2 \neq \emptyset$. Let $e \in C_2$ and $f = uv \in C_1 \setminus C_2$. Starting at v traverse C_1 in the direction away from u until the first vertex of C_2, say x. Denote the (v, x)-path as P. Starting at u traverse C_1 in the direction away from v until the first vertex of C_2, say y. Denote the (u, y)-path as Q. C_2 is a cycle, there are two (x, y)-path in C_2. Denote the (x, y)-path without e as R. Then $vPxRyQ^{-1}uf$ is a cycle. □

Theorem 1.2. *Let T be a spanning tree of G. And let $e \in E \setminus T$ then $T + e$ contains a unique cycle C and for any edge $f \in C$, $T + e - f$ is a spanning tree of G*

Let (D, b, w) be a transshipment problem. A feasible solutions x is a **feasible tree solution** if there is a spanning tree T such that $||x|| = \{e \in A, x_e \neq 0\} \subseteq T$.
The strategy of network simplex algorithm is to generate negative cycles, if negative cycle exists, it means the solution can be improved.
For any tree T of D and for $e \in A \setminus T$, it follows from above theorem that $T+e$ contains a unique cycle. Denote that cycle $C(T, e)$ and orient it in the direction of e, define

$$\begin{aligned} w(T, e) = & \sum \{w_e : e \text{ forward in } C(T, e)\}\\ & - \sum \{w_e : e \text{ reverse in } C(T, e)\} \end{aligned} \tag{1.5}$$

We think of $w(T, e)$ as the weight of $C(T, e)$.
The following is the Network Simplex Method Algorithm:

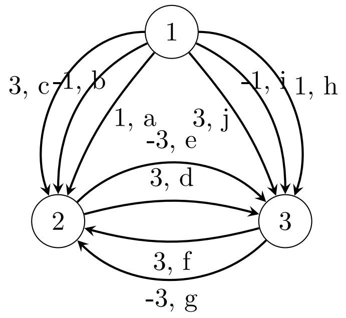
Algorithm 1 Network Simplex Method Algorithm

Ensure: An optimal solution or the conclusion that (D, b, w) is unbounded
Require: A transshipment problem (D, b, w) and a feasible tree solution x containing to a spanning tree T

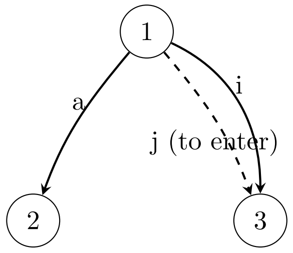
```
while $\exists e \in A \setminus T, w(T, e) < 0$ do
    let $e \in A \setminus T$ be such that $w(T, e) < 0$.
    if $C(T, e)$ has no reverse arcs then
        Return unboundedness
    else
        Set $\theta = \min\{x_f : f$ reverse in $C(T, e)\}$ and set $f = \{f \in C(T, e) : f$ reverse in $C(T, e), x_f = \theta\}$
        if $f$ forward in $C(T, e)$ then
            $x_f \leftarrow x_f + \theta$
        else
            $x_f \leftarrow x_f - \theta$
        end if
        Let $f \in F$ and $T \leftarrow T + e - f$
    end if
end while
Return $x$ as optimal
```

Notice: Similar to Simplex Method in LP, even though in worst case may be inefficient. In most cases it is simple and empirically efficient. Also, similarily, there will be cycling problems.

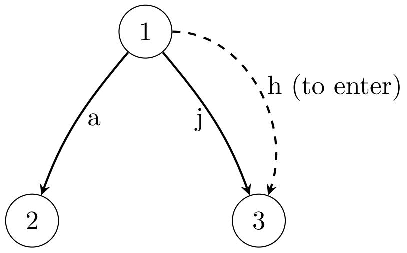
The following is an example of cycling



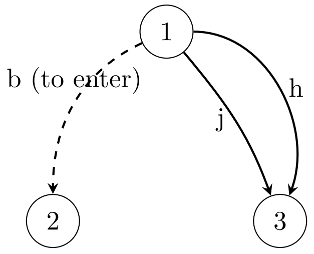
Then for



$w(T, j) = w_j - w_i = -3 - 3 = -6$



$w(T, h) = w_h + w_j - w_i = 3 - 3 - 1 = -1$



Keep going and we will find cycling.
To Avoid cycling we will introduce the modified network Simplex Method. Let T be a **rooted** spanning tree. Let f be an arc in T, we say f is **away** from the root r if $t(f)$ is the component of $T - f$. Otherwise we say f is **towards** r.

Let x be a feasible tree solution associated with T, then we say T is a **strong feasible tree** if for every arc $f \in T$ with $x_f = 0$ then f is away from $r \in T$.
Modification to NSM:

- The algorithm is initialed with a strong feasible tree.
- f in pivot phase is chosen to be the first reverse arc of $C(T, e)$ having $x_f = \theta$. By "first", we mean the first arc encountered in traversing $C(T, e)$ in the direction of e, starting at the vertex i of $C(T, e)$ that minimizes the number of arcs in the unique (r, i)-path in T.

Notice: In the second rule above, r could also be in the cycle, in that case, i is r.

Finding initial strong feasible tree.
Pick a vertex in D to be root r. The tree T has an arc e with the $t(e) = r$ and $h(e) = v$. For each $v \in V \setminus r$ with $b_v \geq 0$ and has an arc e with $h(e) = r$ and $t(e) = v$ for each $v \in V \setminus r$ for which $b_v < 0$. Wherever possible the arcs of T are chosen from A, where an appropriate arc doesn't exist. We create an **artificial arc** and give its weight $|V|(\max\{w_e : e \in A\} + 1$. This is similar to Big-M method and if optimal solution contains artificial arcs ongoing arc problem is infeasible.

Notice: This algorithm can be really bad, its mimic of Simplex Method of LP, which means we can run into exponential operations

1.3 Transshipment Problem and Circulation Problem

Definition 1.3.1. The minimum weight circulation problem is defined as follows:

$$\begin{aligned} \min \quad & wx && (1.6) \\ \text{s.t.} \quad & Nx = 0 && (1.7) \\ & l \leq x \leq u && (1.8) \end{aligned}$$

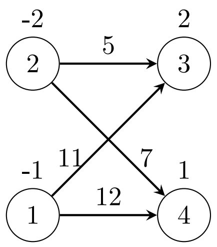
It turns out that the circulation problem is equivalent with transshipment problem.

We will show how to transform any transshipment into circulation.
Let (D, b, w) be a transshipment problem and define two new vertices s and t.

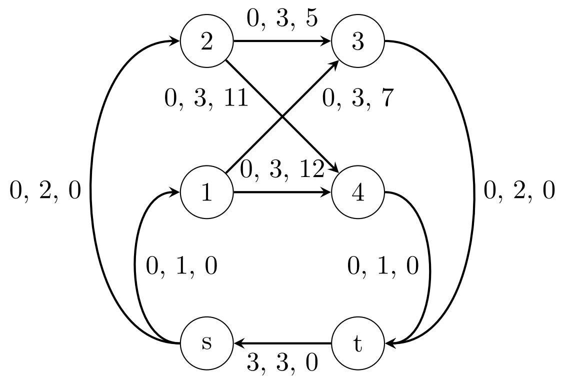
- For each supply vertex x add the arc (s, x) to D with $l_x = 0, u_x = -b_x, w_x = 0$.
- Similarly, for each demand vertex x, add the arc (x, t) to D with $l_x = 0, u_x = b_x, w_x = 0$.
- Finally, add an arc (t, s) having $w_{ts} = 0, l_{ts} = u_{ts} = \sum\{b_x : \forall x, x \text{ is demand vertex}\}$.

- Each original arc is given a $l_x = 0, u_x = \sum\{b_x : \forall x, x \text{ is a demand vertex}\}$, w_x remains unchanged.

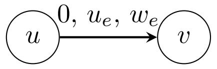
The following is a graph for transshipment problem.



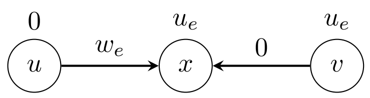
After the above procedures, it is now transformed into a circulation problem.



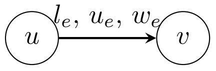
Then, we will show how to transform any circulation problem into transshipment problem.
If the lower bound of the arc is zero, i.e., $l_e = 0$, then for each such arc (u, v), introduce a vertex in between u and v, replace the arc $e = (u, v)$ by $e_1 = (u, x)$ and $e_2 = (x, v)$. Both arcs are uncapacitated. Let $w_1(u, x) = w(u, v)$ and $w_2(x, v) = 0$ be the new weights for the arcs. Let u_e be the demands of newly added vertex x and add u_e to the supplies of vertex v (in v the supplies is the summation from all arcs that go to v).



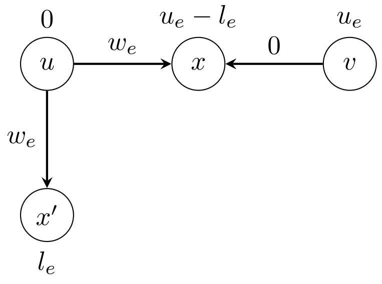
Will be transform into



If the lower bound of the arc is not zero, i.e., $l_e \neq 0$, then for each such arc (u, v), introduce two new vertices, one vertex is in between u and v, similar to the previous case, the difference is the demands of this new vertex is $u_e - l_e$, the others stays the same. Then add the other vertex, this vertex, denoted as x' is added along with an arc between u and x', the weight of this arc is $w(u, v)$, the demands on this new vertex is l_e.



Will be transform into



Perform the above procedures to all the arcs in a circulation problem. In $O(E)$ (polynomially times) transformation, such problem can be transformed into transshipment problem.

1.4 Out-of-Kilter algorithm

This algorithm is a Primal-dual method and is applied to the minimum weight circulation problem.

For LP optimality conditions we need primal feasibility, dual feasibility and complementary slackness, i.e., KKT conditions. Primal and dual feasibility are obvious so we need to show complementary slackness through following theorem.

Theorem 1.3. *Let x be a feasible circulation flow for (D, l, u, w). And suppose there exists a real value vector $\{y_i : i \in V\}$ which we called **vertex-numbers**. For all edges $e \in A$*

$$y_{h(e)} - y_{t(e)} > w_e \text{ implies } x_e = u_e \tag{1.9}$$
$$y_{h(e)} - y_{t(e)} < w_e \text{ implies } x_e = l_e \tag{1.10}$$

Then x is optimal to the circulation problem.

Proof. For each $e \in A$ define

$$\gamma_e = \max\{y_{h(e)} - y_{t(e)} - w_e, 0\} \tag{1.11}$$
$$\mu_e = \max\{w_e - y_{h(e)} + y_{t(e)}, 0\} \tag{1.12}$$

Then

$$\gamma_e - \mu_e = y_{h(e)} - y_{t(e)} - w_e \tag{1.13}$$

Furthermore

$$\sum_{e\in A}(\mu_e l_e - \gamma_e u_e) \tag{1.14}$$

$$= \sum_{e\in A}(\mu_e l_e - \gamma_e u_e) + \sum_{i\in V} y_i(\sum_{h(e)=i} x_e - \sum_{t(e)=i} x_e) \tag{1.15}$$

$$= \sum_{e\in A}(\mu_e l_e - \gamma_e u_e + x_e(y_{h(e)} - y_{t(e)})) \tag{1.16}$$

$$= \sum_{e\in A}(\mu_e l_e - \gamma_e u_e + x_e(\gamma_e - \mu_e + w_e)) \tag{1.17}$$

$$= \sum_{e\in A}(\gamma_e(x_e - u_e) + \mu_e(l_e - x_e) + x_e w_e) \tag{1.18}$$

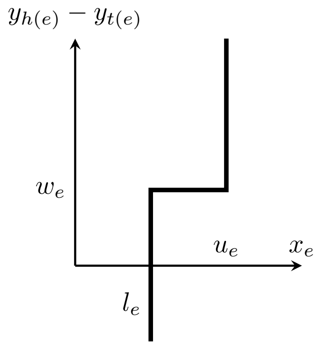
$$\leq \sum_{e\in A} x_e w_e \tag{1.19}$$

The last inequality will be satisfied as equality iff the first two hold. □

The following is the formulation of circulation problem

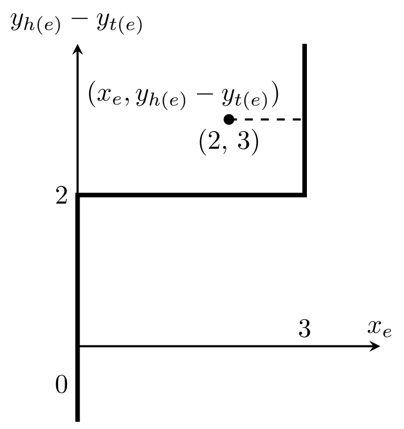
$$\begin{array}{rll} \text{(P)} & \min & wx \quad (1.20)\\ & \text{s.t.} & Nx = 0 \quad y \quad (1.21)\\ & & x \geq l \quad z^l \quad (1.22)\\ & & -x \leq -u \quad z^u \quad (1.23)\\ \text{(D)} & \max & lz^l - uz^u \quad (1.24)\\ & \text{(s.t.)} & yN^{-1} + z^l - z^u \leq w \quad (1.25)\\ & & y \quad free \quad (1.26)\\ & & z^l, z^u \geq 0 \quad (1.27)\\ & \text{(CS)} & y_{h(e)} - y_{t(e)} > w_e \Rightarrow x_e = u_e \quad (1.28)\\ & & y_{h(e)} - y_{t(e)} < w_e \Rightarrow x_e = l_e \quad (1.29) \end{array}$$

There is an alternative way of circulation optimality for a circulation problem. We define a **kilter-diagram** as follows.
For every edge construct the following:



For each point $(x_e, y_{h(e)} - y_{t(e)})$ we define a **kilter-number** k_e, be the minimum positive distance change in x_e required to put in on the kilter line.

Example. For edge $e : w_e = 2, l_e = 0, u_e = 3$, assume $x_e = 2, y_{h(e)} - y_{t(e)} = 3$, then $k_e = 1$



Lemma 1.4. *If for every circulation x and vertex number y we have $\sum_{e\in A} k_e = 0$, then x is optimal.*

Proof. Since k_e is a nonnegative number, then the only way that $\sum_{e\in A} k_e = 0$ is $k_e = 0, \forall e \in A$, which means $\forall e \in A, l_e \leq x_e \leq u_e$. Furthermore, the complementary slackness are satisfied. □

General idea of algorithm follows. Suppose we are given a circulation x and vertex-numbers y (we do not require feasibility). Usually we pick $x = 0, y = 0$. If every edge is in kilter-line then we are optimal.
Otherwise there is at least one edge e^* that is out-of kilter. The algorithm consist of two phases, one called **flow-change** phase (horizontally), then other **number-change** phase (vertically).
In the flow-change phase, we want to find a new circulation for an out-of-kilter edge e^* say $\hat{e}$ such that we reduce the kilter number k_{e^*}, without increasing any other kilter number for other edges.
To do this, denote the edges of e^* to be s and t, where such that k_{e^*} will be decreased by increasing the flow from s to t on e^*.
If $e^* = (s, t)$ this will accomplished by increasing x_{e^*} and if $e = (t, s)$ it is accomplished by decreasing x_{e^*}.
To do this we look for an (s, t)-path p of the following edges.

- If e is forward in p, then increasing x_e does not increase k_e and
- If e is reversed in p, then decreasing x_e dose not increase k_e

In terms of kilter diagram, an arc satisfies "forward" if it is forward and in left side of kilter line, and it satisfies *reversed* if it is reverse and in right side of kilter line.
Suppose we can not find such a path. From s to t, let x be the vertices that can decrease by an augmenting path. Then either we can change the vertex numbers y so that $\sum_{e\in A} k_e$ does not increase but x does, or we can show that problem is infeasible.
INPUT a minimum circulation problem (D, l, u, w) a circulation x and vertex-numbers y OUTPUT conclusion

that (D, l, u, w) is infeasible or an minimum weighted flow.

Step 1: If every arc is in kilter ($k_e = 0, \forall e \in A$). Stop with x is optimal. Otherwise let e^* be an out-of-kilter arc. If increasing x_{e^*} decreases k_{e^*} set $s = h(e^*)$ and $t(e^*)$ otherwise set $s = t(e^*)$ and $t = h(e^*)$

Step 2: If there exists an (s, t) augmenting path p then goto Step 3, otherwise goto Step 4.

STEP 3: Set $y_e = y_{h(e)} - y_{t(e)}, e \in A$ Set $\Delta_1 = \min\{u_e - x_e : e \text{ is forward and } y_e \geq w_e\}$ Set $\Delta_2 = \min\{l_e - x_e : e \text{ is forward and } y_e < w_e\}$ Set $\Delta_3 = \min\{x_e - l_e : e \text{ is reverse and } y_e \leq w_e\}$ Set $\Delta_4 = \min\{x_e - u_e : e \text{ is reverse and } y_e > w_e\}$ $\Delta = \min\{\Delta_i, i = 1, 2, 3, 4\}$

Increase x_e by Δ on each forward arc in p, decrease x_e by Δ on each reverse arc in p.

If $e^* = (s, t)$ decrease x_{e^*} by Δ, otherwise increase x_{e^*} by Δ

If $k_{e^*} > 0$ goto Step 2. otherwise goto Step 1.

Step 4: Let X be the set of vertices reachable from s by augmenting paths, then $t \notin X$, if every arc e with $h(e) \in X$ has $x_e \leq l_e$ and every arc e with $t(e) \in X$ has $x_e \geq u_e$, and at least one of the above inequality is strict, then Stop with problem infeasible

Otherwise set $\delta_1 = \min\{w_e - y_e : t(e) \in X, y_e < w_e, x_e \leq u_e \neq l_e\}$ $\delta_2 = \min\{y_e - w_e : h(e) \in X, y_e > w_e, x_e \geq u_e \neq l_e\}$ $\delta = \min\{\delta_1, \delta_2\}$

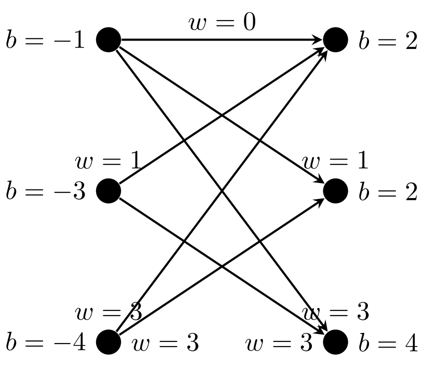
Set $y_i = y_i + \delta$ for $i \notin X$

If $k_{e^*} > 0$, goto Step 2, otherwise goto Step 1.

Out-of-kilter takes $O(|E||V|K)$ where $K = \sum_{e \in A} k_e$. However, there is an algorithm called **scaling algorithm** that uses out-of-kilter as subroutine that runs in $O(R|E|^2|V|)$ where $R = \lceil \max\{\log_2 u_e : e \in A\} \rceil$

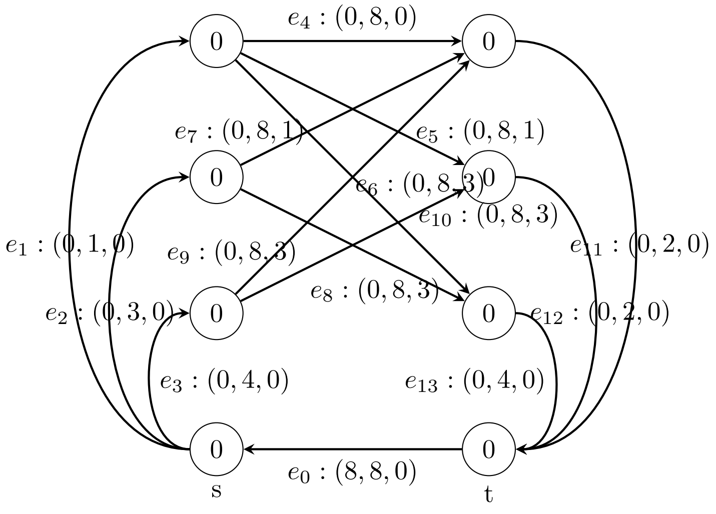
The following is an example of Out-of-Kilter Algorithm.

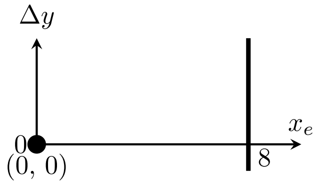
Consider the following transshipment problem:



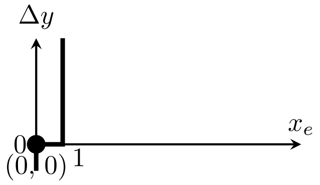
First we transform it into a circulation problem. Let $x_e = 0$ for all edges, let $y_v = 0$ for all vertex

Iteration 1:

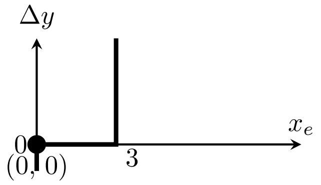




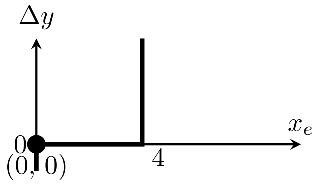
e_0



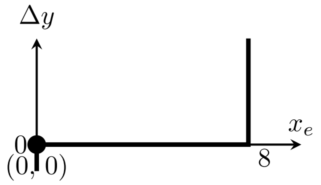
e_1



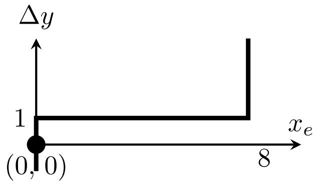
e_2



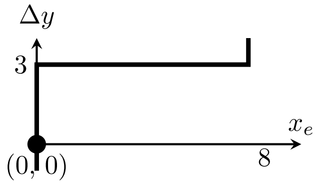
e_3



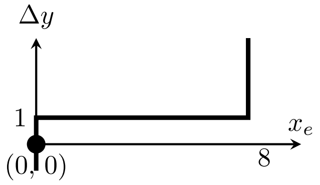
e_4



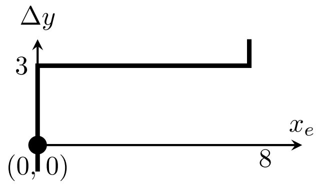
e_5



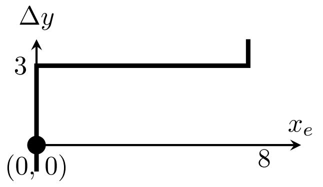
e_6



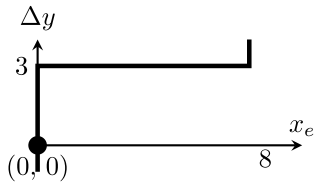
e_7



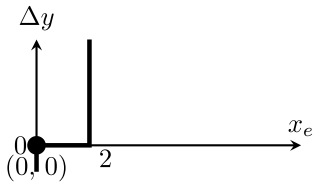
e_8



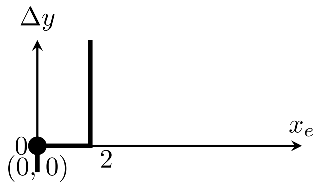
e_9



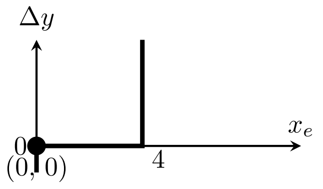
e_{10}



e_{11}



e_{12}



e_{13}

1.5 Complexity of Different Minimum Weighted Flow Algorithms

Let arc capacities between 1 and U, costs between $-C$ and C

Year	Discoverer	Method	Big O
1951	Dantzig	Network Simplex Method	$O(E^2V^2U)$
1960	Minty, Fulkerson	Out-of-Kilter	$O(EVU)$
1958	Jewell	Successive Shortest Path	$O(EVU)$
1962	Ford-Fulkerson	Primal Dual	$O(EV^2U)$
1967	Klein	Cycle Canceling	$O(E^2CU)$
1972	Edmonds-Karp, Dinitz	Capacity Scaling	$O(E^2 \log U)$
1973	Dinitz-Gabow	Improved Capacity Scaling	$O(EV \log U)$
1980	Rock, Bland-Jensen	Cost Scaling	$O(EV^2 \log C)$
1985	Tardos	ϵ-optimality	poly(E, V)
1988	Orlin	Enhanced Capacity Scaling	$O(E^2)$

(1.30)