ALGORITHM FINDING MAXIMUM CONCURRENT MULTICOMMODITY LINEAR FLOW WITH LIMITED COST IN EXTENDED TRAFFIC NETWORK WITH SINGLE REGULATING COEFFICIENT ON TWO-SIDE LINES

Tran Quoc Chien and Nguyen Dinh Lau

The University of Education, University of Danang, Danang, Vietnam

ABSTRACT

Graphs and extended networks are powerful mathematical tools applied in many fields as transportation, communication, information technology, economics, and others. Algorithms to find Maximum Concurrent Multicommodity Flow with Limited Cost on extended traffic networks are introduced in the works we did. However, with those algorithms, capacities of two-sided lines are shared fully for two directions. This work studies the more general and practical case, where flows are limited to use two-sided lines with a single parameter called regulating coefficient. The algorithm is presented in the programming language Java. The algorithm is coded in programming language Java with extended network database in database management system MySQL and offers exact results.

KEYWORDS

Graph, Network, Multicommodity Flow, Optimization, Approximation.

1. INTRODUCTION

Graphs are useful mathematical tools applied in various fields such as transportation, communication, information technology, economics, and others. So far in the graph there has been great and separate concern of the weight of the edges and vertices, in which the length of the road is simply the total of the weight of the edges and vertices. However, in many practical problems, weights at a vertex are different with various routes and depend on arrival edges and departure edge. For example, the period of time to go through the intersection of network traffic depends on the direction of movement of vehicles: turn right, go straight or turn left, even go to restricted direction.

The works [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25] built up maximum flow problem in the network without the capacity of vertex and cost at vertex. The above mentioned works don't mention single regulating coefficient on two-side lines either.

Our article is far optimal than the above works because of two main reasons. First, we add the capacity of vertex in network to solve the problems in real situation, section 4 is devoted to
illustrate the examples and our solutions. Secondly, we use single regulating coefficient on two-side lines \((Y)\) to solve two direction traffic network problems in reality.

The work [4] constructs extended traffic network model and maximum concurrent multicommodity linear flow problem with limited cost to build model for the actual problems to have accurate and efficient solutions. Algorithms to find approximate maximum concurrent multicommodity linear flow with limited cost are developed in the works [4, 5]. Nevertheless, in this algorithm, the capacities of the routes are shared fully for both directions. In some cases, vehicle of one direction will occupy a two-way route, causing traffic digestion. In this work, capacity of traffic flow on the reserve direction route will be regulated by parameter \(Y\). \(Y\) is so-called the regulating coefficient in two-way route. Regulating coefficient \(Y\) ranges from 0.5 to 1. If \(Y = 0.5\), the vehicles must not encroach upon the other side of the street. This is consistent with the principles of solid lines. If \(Y > 0.5\), vehicles from reverse direction can encroach upon the other side of the street with fixed rate regulated by regulating coefficient \(Y\).

The main result of the article is the approximation algorithm to find maximum concurrent multicommodity linear flow with limited cost on two-side lines. The algorithm is coded in programming language Java with extended network database in database management system MySQL and offers exact results.

2. MAXIMUM CONCURRENT MULTICOMMODITY LINEAR FLOW WITH LIMITED COST PROBLEM

Given a mixed graph \(G = (V, E)\) with node set \(V\) and edge set \(E\). The edges can be directed or undirected. There are many kinds of vehicles circulating in the graph. The undirected edges are assigned two-way route and commodities with the same line in the opposite direction shares the capacities. Let us consider the following functions [1], [6], [7], [8], [9], [10].

Edge capacity function \(c_E: E \rightarrow \mathbb{R}^+\), \(c_E(e)\) is the capacity of edge \(e \in E\).

Node capacity function \(c_V: V \rightarrow \mathbb{R}^+\), \(c_V(u)\) is the capacity of node \(u \in V\).

\(b_E\) cost function: \(E \rightarrow \mathbb{R}^+\), \(b_E(e)\) is the cost we have to pay to transfer a commodity to edge \(e\).

Note that on two-side lines, the cost for the two directions might be different. \(v \in V\), labeled as \(E_v\), is the edge set belonging to node \(v\).

\(b_V\) cost function: \(V \times E \times E \rightarrow \mathbb{R}^+\), \(b_V(u, e, e')\) là the cost we have to pay to transfer a commodity from line \(e\) via link node \(u\) to line \(e'\).

Set \((V, E, c_E, c_V, b_E, b_V)\) is called extended traffic network.

Given that \(p\) is the path from node \(u\) via link edges \(e_i\) to node \(v\), \(i = 1…(h+1)\), and nodes \(u_i, i = 1…h\),

\[p = [u, e_1, u_1, e_2, u_2, ..., e_h, u_h, e_{h+1}, v]\]

The cost of circulating of a commodity on the path \(p\) is defined as:

\[b(p) = \sum_{i=1}^{h+1} b_E(e_i) + \sum_{i=1}^{h} b_V(u_i, e_i, e_{i+1}) \quad (1)\]

Given an extended traffic graph \(G = (V, E, c_E, c_V, b_E, b_V)\). It is assumed that \(G\) has \(k\) pairs of source-sink nodes \((s, t)\). Each pair is assigned with commodity \(j\), \(j=1..k\). \(\Pi_j\) is a path
set from \( s_j \) to \( t_j \) in \( G \), \( j = 1..k \), and for \( \Pi = \bigcap_{j=1}^{k} \Pi_j \).

For each path \( p \in \Pi_j \), \( j = 1..k \), \( x(p) \) is commodity \( j \) circulating on path \( p \).
\( \Pi_e \) refers to a set of paths in \( \Pi \) visiting edge \( e \), \( \forall e \in E \).
\( \Pi_v \) refers to a set of paths in \( \Pi \) visiting node \( v \), \( \forall v \in V \).

\( F = \{ x(p) | p \in \Pi_j, j=1..k \} \)
is defined as multicommodity flow in extended traffic network if it it satisfies the following capacity constraint:

\[
\sum_{p \in \Pi_j} x(p) \leq c_E(e), \forall e \in E
\]  \hspace{1cm} (2)

\[
\sum_{p \in \Pi_j} x(p) \leq c_v(v), \forall v \in V
\]  \hspace{1cm} (3)

Expression

\[
v_j = \sum_{p \in \Pi_j} x(p), j = 1..k
\]  \hspace{1cm} (4)

is defined as the value of flow \( F \) for source-sink pair \( (s_j, t_j) \).

Given an extended traffic graph \( G = (V, E, c_E, c_V, b_E, b_V) \). It is assumed that \( G \) has \( k \) pairs of source-sink nodes \( (s_j, t_j) \). Each pair is assigned with commodity \( j \), \( j = 1, ..., k \). Each commodity requests flow value \( d(j) \) from source node \( s_j \) to sink node \( t_j \), \( \forall j = 1, ..., k \). Let \( B \) denote as the limited cost. The goal of the problem is to find the maximum value \( \lambda \) so that there is an available maximum multicommodity to transfer flow value \( \lambda d(j) \) with commodity \( j \) through flow, \( \forall j = 1, ..., k \). Simultaneously, the total cost of flow does not exceed limited cost \( B \).

The problem is illustrated in linear program mode \((P)\) as following:

\[
\lambda \rightarrow \max
\]

Then:

\[
\begin{align*}
\sum_{p \in \Pi_j} x(p) &\leq c_E(e), \forall e \in E \\
\sum_{p \in \Pi_j} x(p) &\leq c_v(v), \forall v \in V \\
\sum_{p \in \Pi_j} x(p) &\geq \lambda d(j), \forall j = 1..k \\
\sum_{p \in \Pi_j} b(p).x(p) &\leq B \\
x &\geq 0, \lambda \geq 0
\end{align*}
\]  \hspace{1cm} (P)
3. Algorithm to Find Maximum Concurrent Multicommodity Linear Flow with Limited Cost on Extended Traffic Network with Single Regulating Coefficient on Two-Side Lines

◊ Input:
1. Extended network $G = (V, E, c_E, c_V, b_E, b_V)$, $n = |V|, m = |E|$.
2. Demand $(s_j, t_j, d_j), j = 1..k$.
3. Limited cost $B$.
4. Approximation coefficient $\omega > 0$.
5. Regulating coefficient $Y \in [0.5; 1]$.

◊ Output:
1. Maximum $\lambda : \lambda_{\text{max}}$
2. Actual flow: 
   $\{fe_j(a), fv_j(u, e, e') | a \in E, (e, u, e') \in B_v, j = 1,...,k\}$.
3. Actual cost $B_f \leq B$.

◊ Algorithm:
// Initialize values
\[
\epsilon = 1 - \frac{1}{1 + \omega}; \quad \delta = \left(\frac{m + n + 1}{1 - \epsilon}\right)^{\frac{1}{r}}; \\
le(e) = \delta / c_E(e), \forall e \in E; lv(v) = \delta / c_V(v), \forall v \in V; \\
\phi = \delta / B; \\
D = (m+n+1)\delta; \\
f_{e_j}(a) = 0; \forall a \in E, \\
f_{v_j}(u, e, e') = 0; \forall u \in V, \forall (e, u, e') \in B_v, j = 1..k \\
t = 1; // period count variable \\
B_{ex} = 0; // estimated cost \\
while D < 1 do // period 
  \{
    for j = 1 to k do // repeated cycles within j 
      \{
        d' = d_j; // the amount of commodity 
        // flowing from $s_j$ to $t_j$ 
        while d' > 0 do // steps in each period 
          \{
            // Algorithm finding the shortest path ([2,3]) is used to find the shortest path $p$ from $s_j$ to $t_j$ and use the following length function**
            \[
length(p) = \sum_{i=1}^{h+1} le(e_i) + \sum_{i=1}^{h} lv(u_i) + b(p) \cdot \phi
\]
\[
= \sum_{i=1}^{k} [\varphi b_e(e_i) + le(e_i)] + \sum_{i=1}^{k} [\varphi b_v(u_i, e_i, e_{i+1}) + lv(u_i)]
\]

//In which

\[
b(p) = \sum_{i=1}^{k} b_e(e_i) + \sum_{i=1}^{k} b_v(u_i, e_i, e_{i+1})
\]

\[
f' = \min\{d', c_E(e), c_V(v) \mid e \in p, v \in p\};
\]

\[
B' = b(p) * f';
\]

\[
\text{if } B' > B \{ f' = f' * B / B'; B' = B \};
\]

// editing flows

\[
fe(a) = fe(a) + f'; \forall a \in p
\]

\[
fv_j(u, e, e') = fv_j(u, e, e') + f'; \forall (e, u, e') \in p
\]

// editing other parameters

\[
d' = d' - f'; \varphi = \varphi * (1 + \epsilon * B' / B);
\]

\[
le(e) = le(e) * (1 + \epsilon * f' / c_E(e)); \forall e \in p
\]

\[
lv(v) = lv(v) * (1 + \epsilon * f' / c_V(v)); \forall v \in p
\]

\[
D = D + \epsilon * f' * \text{length}(p);
\]

\[
B_{ex} = B_{ex} + B';
\]

} //While

} //While d'

} //**** Check two-way route ****/

for e=(u,v) two-way

\{

fe(u, v) = \sum_{j=1}^{k} fe_j(u, v);

fe(v, u) = \sum_{j=1}^{k} fe_j(v, u);

\}

\}

} //**** finish checking ****/

\]

} //While D<1

} //edit actual flows

\[
c' = \max\left\{ \frac{le(e)}{\delta / c_E(e)}, \frac{lv(v)}{\delta / c_V(v)}, \frac{\varphi}{\delta / B} \mid e \in E, v \in V \right\}
\]

61
$c_{ex} = \log_{1+\varepsilon} c'$

$f_{ej}(a) = \frac{f_{ej}(a)}{c_{ex}}; \forall a \in E, j=1..k$

$f_{vj}(u,e,e') = \frac{f_{vj}(u,e,e')}{c_{ex}}; \forall u \in V, \forall (e,u,e') \in \mathcal{B}_v, j=1..k$

$B_{f} = \frac{B_{ex}}{c_{ex}}; // acutal cost$

$\lambda_{max} = \frac{t}{c_{ex}}; // Maximum rate$

// Edit reverse direction flow in two-way route
for $e \in E, e=(u,v)$ two-way
    for $j = 1$ to $k$
        if $(f_{ej}(u,v)> f_{ej}(v,u))$ and $(f_{ej}(v,u)>0)$
        {
            $f_{ej}(u,v) = f_{ej}(u,v) - f_{ej}(v,u);$
            $B_{f} = B_{f} - (b_{E}(u,v) + b_{E}(v,u))* f_{ej}(v,u);$
            $f_{ej}(v,u) = 0;$
        }
        if $(f_{ej}(v,u)\geq f_{ej}(u,v))$ and $(f_{ej}(u,v)>0)$
        {
            $f_{ej}(v,u) = f_{ej}(v,u) - f_{ej}(u,v);$
            $B_{f} = B_{f} - (b_{E}(u,v) + b_{E}(v,u))* f_{ej}(u,v);$
            $f_{ej}(u,v) = 0;$
        }

4. PROGRAM AND EXAMPLES

The algorithm is coded in programming language Java with extended network database in database management system MySQL.

Given an extended network as in figure 1 with 6 nodes, 6 directed edges và 3 undirected edges. The cost of edge $b_{E}$ is shown in table 1 and the cost of node $b_{v}$ in table 2. The capacity of each edge is 10, the capacity for each node is là 20. There are 3 source-sink pairs (1,5), (2,4) and (3,6) within an amount of commodities. $d(1) = 10, d(2) = 10$ và $d(3) = 10$.

![Figure 1. Extended network G](image-url)
Table 1. The cost of edge $b_E$

<table>
<thead>
<tr>
<th>Edges</th>
<th>$b_E$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,2)</td>
<td>10</td>
</tr>
<tr>
<td>(1,3)</td>
<td>9</td>
</tr>
<tr>
<td>(2,3)</td>
<td>10</td>
</tr>
<tr>
<td>(2,5)</td>
<td>10</td>
</tr>
<tr>
<td>(3,4)</td>
<td>15</td>
</tr>
<tr>
<td>(3,5)</td>
<td>11</td>
</tr>
<tr>
<td>(4,6)</td>
<td>10</td>
</tr>
<tr>
<td>(4,5)</td>
<td>10</td>
</tr>
<tr>
<td>(5,6)</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 2. The cost of node $b_V$

<table>
<thead>
<tr>
<th>Node</th>
<th>Edge 1</th>
<th>Edge 2</th>
<th>$b_V$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>(1,2)</td>
<td>(2,3)</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>(1,2)</td>
<td>(2,5)</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>(3,2)</td>
<td>(2,5)</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>(1,3)</td>
<td>(3,4)</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>(1,3)</td>
<td>(3,5)</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>(1,3)</td>
<td>(3,2)</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>(5,3)</td>
<td>(3,2)</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>(5,3)</td>
<td>(3,4)</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>(2,3)</td>
<td>(3,4)</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>(2,3)</td>
<td>(3,5)</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>(3,4)</td>
<td>(4,6)</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>(3,4)</td>
<td>(4,5)</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>(5,4)</td>
<td>(4,6)</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>(2,5)</td>
<td>(5,3)</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>(2,5)</td>
<td>(5,4)</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>(2,5)</td>
<td>(5,6)</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>(3,5)</td>
<td>(5,4)</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>(3,5)</td>
<td>(5,6)</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>(4,5)</td>
<td>(5,3)</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>(4,5)</td>
<td>(5,6)</td>
<td>1</td>
</tr>
</tbody>
</table>

First, we consider the absence of regulatory constraints, i.e., regulating coefficient $Y = 1$.

Limited costs $B = 600$.
Choose approximation coefficient $\omega = 0.024$.
The results of maximum flow are demonstrated as following:
Maximum value $\lambda = 0.8968841421395084$
Actual cost: 589,634,279,458,095.6

The number of commodities is directed or distributed for each source-sink pair is 9.0.
Flow distribution for commodities from source 1 to sink 5, source 2 to sink 4 và source 3 to sink 6 is illustrated in figure 2, figure 3 and figure 4.
It can be seen that the total of flows from node 3 to node 5 is 2.8 + 6.2 = 9.0.

Thus, the total of flows from node 3 to node 5 is nearly 10. It means that it almost encroaches upon the other side of the route from node 5 to node 3.

Now for regulating coefficient $Y = 0.6$.

This means that the total of flows of one direction in two-way route do not exceed 60% of the capacity.

Let limited cost $B = 600$.

Choose approximation coefficient $\omega = 0.024$.

The results of maximum flow are demonstrated as following:

Maximum value $\lambda = 0.798637155967107$

Actual cost: 546,3976778135944

The number of commodities is directed or distributed for each source-sink pair is 8.0.

Flow distribution for commodities from source 1 to sink 5, source 2 to sink 4 and source 3 to sink 6 is illustrated in figure 5, figure 6 and figure 7.
It can be seen that the total of flows 8,0 is smaller than 9,0. In this case, it is impossible to regulate two-way route.

In contrast, the total of flows from node 3 to node 5 is 2,5 + 3,5 = 6,0

It is clear that the regulated flows do not exceed 60% of capacity.

5. CONCLUSION

The main result of the article is the approximation algorithm to find maximum concurrent multicommodity linear flow with limited cost on two-side lines. The algorithm is presented in the programming.

The algorithm applied to solve many practical problems can be the form of the optimization problem through extented multicommodities network, such as the traffic management problem, freight, flow distribution in the Internet, means circulation and others.

The algorithm is coded in programming language Java with extended network database in database management system MySQL and offers exact demonstrations and reliable results.
REFERENCES


Authors
