EAI Endorsed Transactions
on Industrial Networks and Intelligent Systems Research Article **EAI.EU**

Applying algorithm finding shortest path in the multiple-weighted graphs to find maximal flow in extended linear multicomodity multicost network

Chien Tran Quoc [1], **Hung Ho Van** [2]

[1] *University of Da Nang*, tqchien@dce.udn.vn
[2] *Quang Nam University*, hovanhung@qnamuni.edu.vn

Abstract

The shortest path finding algorithm is used in many problems on graphs and networks. This article will introduce the algorithm to find the shortest path between two vertices on the extended graph. Next, the algorithm finds the shortest path between the pairs of vertices on the extended graph with multiple weights is developed. Then, the shortest path finding algorithms is used to find the maximum flow on the multicommodity multicost extended network is developed in the article [12].

Key word: Graph; Network; Multicommodity Multicost flow; Optimization; Linear Programming.

Received on 12 October 2017, accepted on 7 December 2017, published on 21 December 2017

doi: 10.4108/eai.21-12-2017.153499

[2] Corresponding author: Hung Ho Van. Email: hovanhung@qnamuni.edu.vn

1. Introduction

The shortest path finding algorithm is used in many problems on graphs and networks. This article will introduce the algorithm to find the shortest path between two vertices on the extended graph. Next, the algorithm finds the shortest path between the pairs of vertices on the extended graph with multiple weights is developed. Then, the shortest path finding algorithms is used to find the maximum flow on the multicommodity multicost extended network is developed in the article [12].

2. The problem of finding the shortest path in extended graph

Given extended graph $G = (V, E)$ with a set of vertices V and a set of edges E, where edges can be directed or undirected. Each edge $e \in E$ is assigned a *weight* $we(e)$. For each vertex $v \in V$, we denote E_v the set of edges incident vertex v. For each vertex $v \in V$ and each of pair of edges $(e,e') \in E_v \times E_v$, $e \neq e'$ is assigned switch weight $wv(v,e,e')$.

The sets (V, E, we, wv) are called *extended graph*.

Let p be a path from a vertex u to a vertex v through the edges e_i, $i = 1, \dots, h+1$, and vertices u_i, $i = 1, \dots, h$, as following:

$$p = [u, e_1, u_1, e_2, u_2, \dots, e_h, u_h, e_{h+1}, v] \quad (1)$$

Define the length of the path p, denoted $l(p)$, as following:

$$l(p) = \sum_{j=1}^{h+1} we(e_j) + \sum_{j=1}^{h} wv(u_j, e_j, e_{j+1}) \quad (2)$$

• *The problem of finding the shortest path*

Given extended graph $G = (V, E, we, wv)$ and vertices $s, t \in V$. Find the shortest path from s to t.

• **Algorithm**

◊ *Input*. The extended graph $G = (V, E, we, wv)$ and vertices $s, t \in V$.

◊ *Output*. $l(t)$ is the length of the shortest path from s to t, and the shortest path (if $l(t) < +\infty$).

◊ Procedure

The algorithm uses the following notations:

S is a set of the vertices that found the shortest path starting from s;

$T = V - S$;

$l(v)$ is the length of the shortest path from s to v;

$le(v)$ is the edge that leads to the vertex v on the shortest path from s to v;

$VE = \{(v,e) \mid v \in V - \{s\} \ \& \ e \in E_v\} \cup \{(s,\varnothing)\}$ is the set of pairs of vertices and incident edges;

SE is a set of vertex-edge excluded from VE;

$TE = VE - SE$;

$L(v, e)$ is the label of the vertex-edge pair $(v,e) \in VE$

$P(v, e)$ is the vertex-edge adjacent before $(v,e) \in VE$.

// *Initialization*

Asign to

$S = \varnothing$; $T = V$;

$VE = \{(v,e) \mid v \in V - \{s\} \ \& \ e \in E_v\} \cup \{(s,\varnothing)\}$;

$SE = \varnothing$; $TE = VE$;

$L(v,e) = +\infty$; $\forall (v,e) \in VE$, $L(s,\varnothing) = 0$;

for $(v,e) \in VE$: $P(v,e) = \varnothing$;

do

{

Calculate $m = \min\{L(v,e) \mid (v,e) \in TE\}$.

if ($m < +\infty$)

{

Choose $(v_{min},e_{min}) \in TE$ such that

$L(v_{min},e_{min}) = m$;

$TE = TE - \{(v_{min},e_{min})\}$; $SE = SE \cup \{(v_{min},e_{min})\}$;

if ($v_{min} \notin S$)

{

$le(v_{min}) = e_{min}$; $S = S \cup \{v_{min}\}$;

$l(v_{min}) = L(v_{min},e_{min})$; $T = T - \{v_{min}\}$;

}

if ($t <> v_{min}$)

{

for $(v,e) \in TE$ adjacent after (v_{min},e_{min})

{

if ($v_{min} == s$)

$L'(v,e) = L(s,\varnothing) + we(v_{min},v)$;

else

$L'(v,e) = L(v_{min},e_{min}) + we(v_{min},v) + wv(v_{min},e_{min},e)$;

if ($L(v,e) > L'(v,e)$)

{

$L(v,e) = L'(v,e)$; $P(v,e) = (v_{min},e_{min})$;

}

}

}

}

} while ($m < +\infty$ or $t <> v_{min}$)

if ($m == +\infty$) 'no path exists from s to t';

else // finding the shortest path

{

Assign to $l(t) = L(t,le(t))$; // shortest path length from s to t.

// Moves from t, in reverse direction, to the preceding vertex-edges, we get the shortest path as follows:

$k=1$; $(v_k,e_k) = P(t,le(t))$;

while ($(v_k,e_k) <> (s,\varnothing)$)

{

$k=k+1$; $(v_k,e_k) = P(v_{k-1},e_{k-1})$;

}

}

// Describe the shortest path is

$$s \to v_k \to v_{k-1} \to \ldots \to v_1 \to t$$

// End

● *Theorem* 2.1. The algorithm that finds the shortest path in the extended graph is correct and has an algorithmic complexity of $O(n^3)$ (n is the number of vertices in the graph).

Proof [7] [8]

3. The problem of finding the shortest path on the multiple-weighted extended graph

Given extended graph $G = (V, E)$ with a set of vertices V and a set of edges E, where edges can be directed or undriected. On the graph there are r edge weights we_i and switch weights wv_i, $i=1..r$.

The set $(V, E, \{we_i, wv_i \mid i=1..r\})$ is called *the multiple-weighted extended graph*

Let p be the path from the u to v through the edges e_i, $i = 1, \ldots, h+1$, and vertices u_i, $i = 1, \ldots, h$, as follows

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

Define the length of the path p by edge weight we_i and switch weights wv_i, the symbol $l_i(p)$, $i=1..r$, using the following formula:

$$l_i(p) = \sum_{j=1}^{h+1} we_i(e_j) + \sum_{j=1}^{h} wv_i(u_j, e_j, e_{j+1})$$

• *The problem of finding the shortest path*

Given the multiple-weighted extended graph $G = (V, E, \{we_i, wv_i \mid i=1..r\})$. Assume for each weight i, $i=1..r$, there are k_i source-destination pairs $(s_{i,j}, t_{i,j})$, $j=1..k_i$.

The path length from the source node $s_{i,j}$ to the destination node $t_{i,j}$ is given by the function l_i, $i=1..r$, $j=1..k_i$.

The problem is to find, among the source-destination pairs ($s_{i,j}$, $t_{i,j}$), i=1..r, j=1..k_i, the one that has the smallest shortest path length.

• Algorithm

◊ *Input*. Multiple-weighted extended graph $G = (V, E, \{we_i, wv_i \mid i=1..r\})$. The source-destination pairs ($s_{i,j}$, $t_{i,j}$), i=1..r, j=1..k_i.
◊ *Output*. The source-destination pair ($s_{imin,jmin}$, $t_{imin,jmin}$) with the smallest shortest path length. *lmin* is the shortest path length from $s_{imin,jmin}$ to $t_{imin,jmin}$, and the shortest parth (if $lmin < +\infty$).
◊ *Procedure*

$lmin = +\infty$;
for (i=1 ; i<=r ; i++)
for (j=1 ; j <= k_i ; j++)
{
    $S = \varnothing$; $T = V$;
    $VE = \{(v,e) \mid v\in V-\{s_{i,j}\}$ & $e\in E_v\}\cup\{(s_{i,j},\varnothing)\}$;
    $SE = \varnothing$; $TE = VE$;
    $L(v,e) = +\infty$; $\forall(v,e)\in VE$, $L(s_{i,j},\varnothing) = 0$;
    for $(v,e)\in VE$: $P(v,e) = \varnothing$;
do
{
   Calculate $m = \min\{L(v,e) \mid (v,e)\in TE\}$.
    if ($m < lmin$)
    {
        Choose $(v_{min},e_{min})\in TE$ such that
        $L(v_{min},e_{min}) == m$;
        $TE = TE - \{(v_{min},e_{min})\}$;
        $SE = SE \cup \{(v_{min},e_{min})\}$;
          if ($v_{min}\notin S$)
          {
              $le(v_{min}) = e_{min}$; $S = S\cup\{v_{min}\}$;
              $l(v_{min}) = L(v_{min},e_{min})$; $T = T-\{v_{min}\}$;
          }
          if ($t_{i,j}$ <> v_{min})
          {
              for $(v,e)\in TE$ adjacent after (v_{min},e_{min})
              {
                 if (v_{min}==$s_{i,j}$)
                      $L'(v,e) = L(s,\varnothing)+we(v_{min},v)$;
                else
                      $L'(v,e) = L(v_{min},e_{min})+$
                                      $we(v_{min},v)+wv(v_{min},e_{min},e)$;
                if ($L(v,e) > L'(v,e)$)
                {
                      $L(v,e) = L'(v,e)$;
          $P(v,e) = (v_{min},e_{min})$;
        }
          }
        }
      }
} while ($m < lmin$ or $t_{i,j}$ <> v_{min})
if ($L(t_{i,j},le(t_{i,j}))<lmin$) and ($t_{i,j} = v_{min}$) //edge to ($i$,$j$)
   { $imin = i$; $jmin = j$; $lmin = L(t_{i,j},le(t_{i,j}))$;
        $emin= le(t_{i,j})$; // edge to $t_{i,j}$
         for $(v,e)\in VE$: $Pmin(v,e) = P(v,e)$;
    }
}// end for...for
// find the shortest path
 if ($lmin < +\infty$)
 {
    // Moves from $t_{imin,jmin}$, in reverse direction, to the preceding vertex-edges, we get the shortest path as follows:
      k=1; $(v_k,e_k) = Pmin(t_{imin,jmin}, emin)$;
       while ((v_k,e_k) <> $(s_{imin,jmin},\varnothing)$)
       {
          $k=k+1$; $(v_k,e_k) = P(v_{k-1},e_{k-1})$;
       }
// Deduced the shortest path is
     $s_{imin,jmin} \to v_k \to v_{k-1} \to \ldots \to v_1 \to t_{imin,jmin}$
 // End

• *Theorem* 3.1. The algorithm that finds the smallest shortest path between the pairs of vertices on the multiple-weighted extended graph is correct and has an algorithmic complexity $O(k.n^3)$, where n is the number of vertices and $k=k_1+\ldots+k_r$.
Proof. The correctness of the algorithm derives from theorem 2.1. The algorithm that finds the shortest path between the source-destination vertices has the complexity $O(n^3)$, which inferred the algorithm finding the smallest shortest path between the k of the destination source pair has complexity $O(k.n^3)$.

4. The problem of maximum flow on extended Linear multicommodity multicost network

The model of multicommodity multicost network was built in the article [12].

Given a multicommodity multicost extended G=(V,E, ce, ze, cv, zv, $\{be_i, bv_i, q_i \mid i=1..r\}$). Assume for each commodity i, i=1..r, with k_i source-destination pairs ($s_{i,j}$, $t_{i,j}$), j=1..k_i, each of pair assigned a quantity of commodity type i, which needs to be transferred from source node $s_{i,j}$ to destination node $t_{i,j}$.

The problem is to find the multicommodity flow such that the flow value is maximal.

• **Algorithm**

◊ *Input*: Given a multicommodity multicost extended G=(V,E, ce, ze, cv, zv, $\{be_i, bv_i, q_i \mid i=1..r\}$). Assume for each of commodity type i, i=1..r, there are k_i source-destination pairs ($s_{i,j}$, $t_{i,j}$), j=1..k_i, each pair assigned a quantity of commodity type i which needs to be transferred from source node $s_{i,j}$ to destination node $t_{i,j}$.

ω is the approximation to be achieved.

◊ *Output*: Maximum flow F represents the set of converged streams at the edges

$$F = \{x_{i,j}(e) \mid e \in E,\ i=1..r,\ j=1..k_i\}$$

◊ *Procedure*

// The symbol n=|V|, m=|E|. Calculate ε and δ

$\varepsilon = 1 - \sqrt{1/(1+\omega)}$;

$\delta = (1+\varepsilon)\dfrac{1}{\left[(1+\varepsilon)^2(m+n)\right]^{1/\varepsilon}}$; fv=0;

for $e \in E$: $le(e)=\delta$; $x_{i,j}(e)=0$;

for $v \in V$: $lv(v)=\delta$;

do

{

Using the algorithm to find the source-destination pair ($s_{i,j}$, $t_{i,j}$), $1 \le i \le r$ and $1 \le j \le k_i$, with the smallest shortest path from $s_{i,j}$ to $t_{i,j}$ with edge weight $le(e)$, $\forall e \in E$, and swicth weights at nodes are $lv(v)$, $\forall v \in V$.

Symbol

imin và *jmin* are index pairs of the source-destination nodes has the shortest path.

α is the shortest path length;

p is the shortest path;

c is the smallest capacity of passing edges and vertice of p:

$c=\min\{\min\{ce(e).ze(e) \mid e \in p\}, \min\{cv(v).zv(v) \mid v \in p\}\}$;

// Adjust the flow:

$\forall e \in p$, $x_{imin,jmin}(e) = x_{imin,jmin}(e)+c$; $fv=fv+c$;

$le(e) = le(e).(1+\varepsilon.c/(ce(e).ze(e)))$;

$\forall v \in p$, $lv(v) = lv(v).(1+\varepsilon.c/(cv(v).zv(v)))$;

} while (α <1)

// Calculating the value resulted from flow F and value of flow fv.

$x_{i,j}(e)=x_{i,j}(e)/\log_{1+\varepsilon}\dfrac{1+\varepsilon}{\delta}$, $\forall i=1..r, j=1..k_i, \forall e \in E$;

$fv = fv/\log_{1+\varepsilon}\dfrac{1+\varepsilon}{\delta}$;

// Calculating the flow on the undriected edge

for (i=1 ; i<=r ;i++)

for (j=1 ; j<=k_i ;j++)

for $e \in E$, e undricted

if $x_{i,j}(e) >= x_{i,j}(e')$// e' is the opposite edge e

{

$x_{i,j}(e)=x_{i,j}(e)-x_{i,j}(e')$;

$x_{i,j}(e')=0$;

}

else

{

$x_{i,j}(e')=x_{i,j}(e')-x_{i,j}(e)$;

$x_{i,j}(e)=0$;

}

//End

• *Theorem* 4.1. The algorithm is correct and has an algorithmic complexity

$$O(\omega^{-2}.k.n^3.(m+n).\ln(m+n)),$$

where n is the number of vertices, m is the number of edges and $k=k_1+\ldots+k_r$.

Proof. See [12].

5. Example

Showing an extended network diagram in Figure 1.

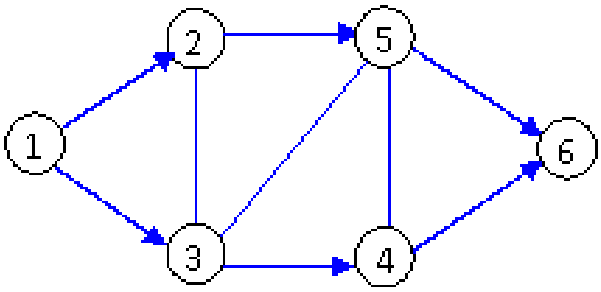

Figure 1. The Network has 6 nodes, 6 directed edges and 3 undirected ones.

The data given in the following tables

Table 1. Node flow capability

Nodes	cv
1	100
2	100
3	50
4	100
5	50
6	100

Table 2. Commodity converting coefficient

Commodity	q
1	1
2	2
3	3

Table 3: Pairs of source-target nodes

Commodity	$s_{i,j}$	$t_{i,j}$
1	1	5
2	2	4
3	3	6

Table 4: Edge capability and cost
Notes: Type 1 is directional, type 0 is undirectional.

Edge	Type	*ce*	be_1	be_2	be_3
(1,2)	1	50	4	5	6
(1,3)	1	50	4	5	6
(2,3)	0	70	4	5	6
(3,2)	0	70	3	4	5
(2,5)	1	50	∞	5	6
(3,4)	1	50	4	5	6
(3,5)	0	70	4	5	∞
(5,3)	0	70	3	∞	5
(4,6)	1	50	4	5	6
(4,5)	0	70	4	∞	6
(5,4)	0	70	3	5	∞
(5,6)	1	50	4	5	6

Table 5. Switch cost

Node	Edge 1	Edge 2	bv_1	bv_2	bv_3
2	(1,2)	(2,3)	1	2	3
2	(1,2)	(2,5)	1	2	3
2	(3,2)	(2,5)	1	2	3
3	(1,3)	(3,4)	1	2	3
3	(1,3)	(3,5)	1	∞	∞
3	(1,3)	(3,2)	1	∞	∞
3	(5,3)	(3,2)	1	2	3
3	(5,3)	(3,4)	1	2	3
3	(2,3)	(3,4)	1	2	3
3	(2,3)	(3,5)	1	2	3
4	(3,4)	(4,6)	1	2	3
4	(3,4)	(4,5)	1	2	3
4	(5,4)	(4,6)	1	2	3
5	(2,5)	(5,3)	1	∞	∞
5	(2,5)	(5,4)	1	∞	∞
5	(2,5)	(5,6)	1	2	3
5	(3,5)	(5,4)	1	2	3
5	(3,5)	(5,6)	1	2	3
5	(4,5)	(5,3)	1	2	3
5	(4,5)	(5,6)	1	2	3

The algorithm is coded in C++ and gives correct results. Below is the result of the above example.

```
Coeficient of approximation:0.070000
Total output: 148.908624
Total cost  : 1877.662162
```

Flow for commodity type 1, which needs to be transferred from source node 1 to destination node 5

```
1   2       8.396823
1   3       41.500042
2   3       8.396823
3   5       49.896862
```

Flow for commodity type 2, which needs to be transferred from source node 2 to destination node 4

```
2   3       0.093091
3   4       0.093091
```

Flow for commodity type 3, which needs to be transferred from source node 3 to destination node 6

```
3   2       49.375553
2   5       49.375553
3   4       49.543118
4   6       49.543118
5   6       49.375553
```

6. Conclusion

The article develops the algorithm finding the shortest path in extended graphs (Section 2), the algorithm finding the shortest path on the multiple-weighted extended graph (Section 3). Based on the duality theory of linear programming, an approximation algorithm with polynomial complexity is developed on the base of the algorithm finding shortest paths in section 2 and 3. This is also the main result of the article. Correctness and algorithm complexity are justified and the algorithm is stored in C++ and given an exact result. The results of this article are the basis for studying the applications of multicomodity multicost flow optimization .

7. References

[1] Naveen Garg, Jochen Könemann: Faster and Simpler Algorithms for Multicommodity Flow and Other Fractional Packing Problems, SIAM J. Comput, Canada, 37 (2), 2007, pp. 630-652.

[2] Xiaolong Ma, Jie Zhou: An Extended Shortest Path Problem with Switch Between Arcs, Proceedings of the International Conference on Engineers and Computer Scientists 2008 Vol IIMECS 2008, 19-21 March, 2008, Hong Kong.

[3] Tran Quoc Chien: Linear multitransport network calculation, ministerial scientific research, code B2010DN-03-52.

[4] Tran Quoc Chien, Tran Thi My Dung: Applying of algorithm finding the shortest path for maximum multicommodity flow. Journal of Science & Technology, University of Danang, 3 (44) 2011.

[5] Tran Quoc Chien: Applying of algorithm finding the shortest path for maximum simultaneous multicommodity flow. Journal of Science & Technology, University of Danang, 4 (53) 2012.

[6] Tran Quoc Chien: Applying of algorithm finding the shortest path for minimum cost maximum simultaneous multicommodity flow. Journal of Science & Technology, University of Danang, 5 (54) 2012.

[7] Tran Quoc Chien: The algorithm findsing the shortest path in the general graph, Journal of Science & Technology, University of Da Nang, 12 (61) / 2012, 16-21.

[8] Tran Quoc Chien, Nguyen Mau Tue, Tran Ngoc Viet: The algorithm finding the shortest path on the extension graph. Proceeding of the 6th National Conference on Fundamental and Applied Information Technology (FAIR), Proceedings of the Sixth National Conference on Scientific Research and Applying, Hue, 20-21 June 2013. Natural Science and Technology Publishing House. Hanoi 2013. p.522-527.

[9] Tran Quoc Chien: Applying of algorithm finding the fastest path for maximum multi-transport linearity of minimum cost on extension transportation networks, Journal of Science & Technology, University of Da Nang . 10 (71) 2013, 85-91.

[10] Tran Ngoc Viet, Tran Quoc Chien, Nguyen Mau Tue: Optimized Linear Multiplexing Algorithm on Extension Transportation Networks, Journal of Science & Technology, University of Da Nang. 3 (76) 2014, 121-124.

[11] Tran Ngoc Viet, Tran Quoc Chien, Nguyen Mau Tue: The calculation of linear multitransport on transportation network. Proceedings of the 7th National Conference on Fundamental and Applied Information Technology Research (FAIR'7), ISBN: 978-604-913-300-8, Proceedings of the 7th National Scientific Conference on "Fundamental and Applied Research IT ", Thai Nguyen, 19-20 / 6/2014. NXB Natural Science and Technology. Hanoi 2014. p.31-39.

[12] Tran Quoc Chien, Ho Van Hung: The multicommodity multi flow expansion network and the algorithm used to find the maximum flow FAIR-2017, August, 2017.