A New(?) k-Shortest Path Algorithm

Stanislav Palúch[1]

Abstract. k-shortests path problem is to find k shortest point-to-point paths in a graph or a digraph. Solution of this problem is useful in many practical cases when we are looking for a sufficiently short path which complies to some additional conditions that cannot be simply formulated. Such problems can be solved by choosing a feasible path among k shortest paths. This paper presents a new simple algorithm for k-shortest paths problem.

Keywords: digraph, shortest path, k-shortest paths problem

JEL classification: C44
AMS classification: 90C15

1 Introduction

Many problems from practical life lead to the problem to find in a network relatively short point to point path which has to fulfill certain special constraints that cannot be simply formulated by means of graph theory. Such problems are very often NP–hard and can be approximately solved by successive enumeration of k shortest paths and consequently by choosing that one from them which complies with given constraints.

The k – shortest path problem is frequently studied in literature. Plesnik presents in [4] a procedure based on prohibiting edges of till now best paths, Yen gives in [5], [6] a deviation algorithm with running time $O(kn(m + n \log n))$ (where n is number of vertices and m is number of edges of undrelying graph) – this worst case assesment has been the best known for long time perion. Algorithm by Gotthilfa, and Lewenstein [1] (issued in March, 2009) runs in time $O(kn(m + n \log \log n))$.

2 Terminology

Since graph terminology is non uniform I present here several essential notions.

A **digraph** (a directed graph) is an ordered pair $G = (V, A)$, where V is a nonempty finite set and A is a set of ordered pairs of the type (u, v) such that $u \in V$, $v \in V$ and $u \neq v$. The elements of V are called **vertices** and the elements of A are called **arcs** of the digraph G.

A $(v1, vk)$ - *walk* in digraph $G = (V, A)$ is an alternating sequence of vertices and arcs of the form

$$\mu(v_1, v_k) = (v_1, (v_1, v_2), v_2, \ldots, v_{k-1}, (v_{k-1}, v_k), v_k).$$

A trivial walk is a walk containing only one vertex. A (v_1, v_k)-**trail** in G is a (v_1, v_k)-walk with no repeated arcs. A (v_1, v_k)-**path** in G is a (v_1, v_k)-walk with no repeated vertices.

An **arc weighted digraph** $G = (V, A, c)$ is an ordered triple where $G = (V, A)$ is a digraph and $c : A \to R$ is a real function defined on the arc set A, the value $c(a)$ for $a \in A$ is called the **weight** of the arc a (or sometimes the **arc-weight**, the **length** or the **cost** of the arc a).

In this paper we will assume that $c(a) > 0$. This condition is fulfilled in many practical applications. The length of the walk $\mu(u, v)$ in a digraph $G = (V, A, c)$ is the total sum of arc-weights of it's arcs, whereas the arc weight is added to the total sum so many times how many times it appears in the walk $\mu(u, v)$.

[1]University of Žilina /Faculty of Management Science and Informatics, Department of Mathematical Methods, Univerzitná 8215/1, 010 26 Žilina, Slovakia, e-mail: stanislav.paluch@fri.uniza.sk

Suppose that $\mu(v_1, v_k) = (v_1, (v_1, v_2), \ldots, (v_{k-1}, v_k), v_k$ and $\nu(u_1, u_l) = (u_1, (u_1, u_2), \ldots, (u_{l-1}, u_l), u_l)$ are two walks and let $v_k = u_1$. The **concatenation** $\mu(v_1, v_k) \oplus \nu(u_1, u_l)$ of walks $\mu(v_1, v_k)$ and $\nu(u_1, u_l)$ is the walk

$$\rho(v_1, u_l) = (v_1, (v_1, v_2), v_2, \ldots, (v_{k-1}, v_k), v_k \equiv u_1, (u_1, u_2), \ldots, (u_{l-1}, u_l), u_l)$$

Denote $V^+(v) = \{w|(v, w) \in A\}$, $A^+(v) = \{(v, w)|(v, w) \in A\}$.

3 Properties of set of shortest k-paths

Let $G = (V, A, c)$ be an arc weighted digraph. The existence of two different (u, v)-paths with the same length could result in difficulties when considering the k-shortest paths problem. If there are two different (u, v)-paths having the same length, we need another priority rule how to determine which one of them is k-th and which $(k + 1)$-th shortest path. This decision can be made in such a way that in the case of two (u, v)-paths with equal length, we can compare them in lexicographicaly order.

Another way how to awoid just mentioned problem is to enumerate the arcs of the set $A - a_1, a_2, \ldots, a_m$ and introduce new arc cost

$$\overline{c}(a_i) = c(a_i) + \frac{\varepsilon}{2^i} \tag{1}$$

For simplicity we will assume whithout loss of generality that there are no two different (u, v)-paths with the same length.

Proposition 1. *Let*

$$\mu_k(u, v) = (u \equiv v_1, (v_1, v_2), v_2, \ldots, v_{r-1}, (v_{r-1}, v_r), v_r \equiv v) \tag{2}$$

be the k-th shortest (u, v)-path. Let $q < r$ and let $\mu(u, v_q) = (u \equiv v_1, (v_1, v_2), v_2, \ldots, v_{q-1}, (v_{q-1}, v_q), v_q)$. Then $\mu(u, v_q)$ is the l-th shortest (u, v_q)-path for some $l \leq k$.

Proof.
Let $\mu_k(u, v)$ be the k-the shortest (u, v)-path, let $q < r$. Then $\mu_k(u, v)$ can be written as concatenation

$$\mu_k(u, v) = \underbrace{(u \equiv v_1, v_2, \ldots, v_q)}_{\mu_l(u, v_q)} \oplus \underbrace{(v_q, v_{q+1}, \ldots, v_r \equiv v)}_{\mu(v_q, v)} = \mu_l(u, v_q) \oplus \mu(v_q, v) \tag{3}$$

If $l > k$, i. e. if $l - 1 \geq k$ then $\mu_1(u, v_q), \mu_2(u, v_q), \ldots, \mu_{l-1}(u, v_q)$ are $l - 1$ shortest (v, v_q) paths. Therefore we have at least $l - 1 \geq k$ paths – namely

$$\mu_1(v, v_q) \oplus \mu(v_q, v), \mu_2(v, v_q) \oplus \mu(v_q, v), \ldots, \mu_{l-1}(v, v_q) \oplus \mu(v_q, v)$$

shorter than $\mu_k(u, v)$ what is in contradiction with the fact that $\mu_k(u, v)$ is the k-th shortest (u, v)-path. □

Colorary. Let $\mu_k(s, w)$ be the k-th shortest (s, w) path having the length equal to d. Then $\mu_k(s, w)$ is concatenation of the following type

$$\mu_k(s, w) = \mu_l(s, v) \oplus (v, (v, w), w), \tag{4}$$

where $\mu_l(s, v)$ is the l-th shortest (s, v)-path, $l \leq k$, $(v, w) \in A$ and $d(\mu_k(s, w)) = d(\mu_l(s, v)) + c(v, w)$.

Definition 1. Let $\mu_k(s, w)$, $\mu_k(s, w)$ be two paths fulfilling (4). We will say that $\mu_k(s, w)$ is **extension** of $\mu_l(s, v)$ and $\mu_l(s, v)$ is **parent path** of $\mu_k(s, w)$.

Definition 2. Let $\mu(s, w)$ be the k-th shortest (s, w)-path in digraph $G = (V, H, c)$. The number k is called the **rank** of the path $\mu(s, w)$. The rank of $\mu(s, w)$ will be denoted as $\text{rank}(\mu(s, w))$.

Denote by $\mathcal{P}$ the set of paths in $G = (V, A, c)$ containing for every $w \in V$ all shortest (s, w)-paths with rank less or equal to K. Every $\mu_k(s, w)$ is uniquely defined by it's last vertex w, last but one vertex v and the l-th shortest (s, v)-path $\mu_l(s, v)$.

Let $\mathcal{D} \subset \mathcal{P}$, let dmax($\mathcal{D}$) be the length of longest path in $\mathcal{D}$, i. e. dmax($\mathcal{D}$) = max$\{d(path)|path \in \mathcal{D}\}$, let $\mathcal{D}$ contain all paths from $\mathcal{P}$ having the length less or equal to dmax($\mathcal{D}$). Denote by $\mathcal{E}$ the set of all extensions of paths from $\mathcal{D}$ with length greater than dmax($\mathcal{D}$).

If $\mathcal{D}$ does not contain all required shortest paths we can add to it next paths by repeating the following procedure. Choose the shortest path from $\mathcal{E}$ denoted by $\mu(s,w)$ and remove it from $\mathcal{E}$. If no (s,w) path is in $\mathcal{D}$, $\mu(s,w) = \mu_1(s,w)$ is the shortest (s,v)-path. If $\mathcal{D}$ contains k-shortest (s,w)-paths where $k < K$, $\mu(s,w) = \mu_{k+1}(s,w)$ is the $(k+1)$-th (s,w) path. In both mentioned cases insert the path $\mu(s,w)$ into the set $\mathcal{D}$ and insert all extensions of the path $\mu(s,w)$ into the set $\mathcal{E}$. If $\mathcal{D}$ contains K shortest (s,w)-paths, throw away the path $\mu(s,w)$.

The question arises whether $\mu(s,w)$ is realy the $(k+1)$-th shortest (s,w)-path. $\mu(s,w)$ is an extension of a path $\mu_l(s,v) \in \mathcal{D}$:

$$\mu(s,w) = \mu_l(s,v) \oplus (v,(v,w),w) \tag{5}$$

Let $\nu(s,w)$ be a (s,w)-path, $\nu(s,w) \notin \mathcal{D}$ and let $d(\nu(s,w)) < d(\mu(s,w))$. Since $\nu(s,w) \notin \mathcal{D}$, dmax($\mathcal{D}$) $< d(\nu(s,w))$. The path $\nu(s,w)$ can be written as

$$\nu(s,w) = (\underbrace{s \equiv w_1,(w_1,w_2),w_2,\ldots,(w_{t-1},w_t),w_t}_{\nu(s,w_t)},\underbrace{(w_t,w_{t+1}),\ldots,w)}_{\nu(w_t,s)} = \nu(s,w_t) \oplus \nu(w_t,s) \tag{6}$$

where t is largest index such that path $\nu(s,w_t) \in \mathcal{D}$.
Therefore the extension $\nu(s,w_t) \oplus (w_t,(w_t,w_{t+1}),w_{t+1}) \in \mathcal{E}$. But the path $\mu(s,w)$ was chosen as the path with the least length in $\mathcal{E}$ and hence $d(\mu(s,w)) < d[\nu(s,w_t) \oplus (w_t,(w_t,w_{t+1}),w_{t+1})] < d(\nu(s,w))$.

Remember that after just described changes the sets $\mathcal{D}$ and $\mathcal{E}$ still keep their properties – namely $\mathcal{D}$ contains all paths from $\mathcal{P}$ with length less or equal than dmax($\mathcal{D}$) and $\mathcal{E}$ contains all extensions of paths from $\mathcal{D}$ longer than dmax($\mathcal{D}$).

This procedure starts with $\mathcal{D} = \{\mu_1(s,s) = (s)\}$ – $\mathcal{D}$ contains only trivial (s,s)-path and $\mathcal{E} = \{(s,(s,w),w)|(s,w) \in H^+(s)\}$.

4 Algorithm

Assume that $V = \{1,2,...,n\}$. Denote

- n – the number of nodes of the digraph $G = (V,A,c)$
- m – the number of arcs of the digraph $G = (V,A,c)$
- s,f – starting and finishing vertex
- K – the number of searched shortest (s,f) paths
- $c(u,v)$ – the length of the arc $(u,v) \in A$

For every vertex $w \in V$ and for $k = 1,2,...,K$ we will compute step by step at most K definitive tripple labels

$$length[w][k], \quad l\_b\_one[w][k] \quad and \quad rankpp[w][k]. \tag{7}$$

having the following meaning:

- $length[w][k]$ – the length of the k-th shortest (s,w)-path $\mu_k(s,w)$
- $l\_b\_one[w][k]$ – the last but one vertex of the k-th shortest (s,w)-path $\mu_k(s,w)$
- $rankpp[w][k]$ – the rank of the parent path of $\mu_k(s,w)$
- $n\_deflab[w]$ – the number of determined definitive labels of the vertex $w \in V$.

The fact that the labels $length[w][k]$, $l\_b\_one[w][k] = 0$ and $rankpp[w][k]$ are defined for some $w \in V$ and k, $1 \leq k \leq K$ implies that k-th shortest (s, w)-path was discovered.

The sequence of computing of labels (7) will start with setting $n\_deflab[s] = 1$, $length[s][1] = 0$, $l\_b\_one[s][1] = 0$ and $rankpp[s][1] = 0$ for the starting vertex s. This means that one shortest (s, s)-path $\mu_1(s, s)$ was discovered with the length 0 and with no parent path. Subsequently, the next the shortest (s, w)-path $\mu(s, w)$ with property

$$\mu(s, w) \notin \mathcal{D} \quad \text{and} \quad \text{rank}(\mu(s, w) \leq K$$

will be inserted into the set $\mathcal{D}$.

This procedure ensures that if labels (7) are defined for some w and k, i.e. if $\mu_k(s, w) \in \mathcal{D}$, then the parent path $\mu(s, v)$ of $\mu_k(s, w)$ is an element of $\mathcal{D}$, too. The labels of $\mu(s, v)$ define parent path of $\mu(s, v)$ etc, etc. The reverse sequence of all vertices of the path $\mu_k(s, w)$ can be determined step by step in this way.

Just mentioned labels fully characterize the set $\mathcal{D}$. The set $\mathcal{D}$ contains all paths with length less or equal than $\text{dmax}(\mathcal{D})$ and with rank less or equal to K.

The set $\mathcal{E}$ was defined as the set of all extensions of paths from $\mathcal{D}$. Every such extension $\mu(s, w)$ can be fully specified by quadruple (w, t, x, k) where w is the final vertex of $\mu(s, w)$, t is the length of $\mu(s, w)$, x is the final vertex of parent paths of $\mu(s, w)$ and k is the rank of parent path $\mu(s, v)$, i.e. $\mu(s, v) = \mu_k(s, v)$ and $\mu(s, w) = \mu_k(s, v) \oplus (v, (v, w), w)$. The set $\mathcal{E}$ will initially be

$$\mathcal{E} = \{(w, c(s, w), s, 1) | w \in V^+(s)\}.$$

Step 0. Initialization.

Set $n\_deflab[s] = 1$, $length[s][1] = 0$, $l\_b\_one[s][1] = 0$ and $rankpp[s][1] = 0$ for the starting vertex s.

For all $w \in V$ such that $w \neq s$ set: $n\_deflab[w] = 0$ (no definitive label is assigned to w).

For all $w \in V$ such that $w \neq s$:
consider all labels $length[w][j]$, $l\_b\_one[w][j]$ and $rankpp[w][j]$ to be undefined

Step 1. Extracting shortest walk from $\mathcal{E}$.

Extract from $\mathcal{E}$ a quadruple (w^*, t^*, x^*, k^*) having the minimum value of the second item t^* in $\mathcal{E}$.

If $n\_deflab[w^*] \geq K$, we need not another shortest (s, w^*)-path. GoTo Step4.

Quadruple (w^*, t^*, x^*, k^*) determines a (s, w^*)-walk $\mu_{k^*}(s, x^*) \oplus (x^*, (x^*, w^*), w^*)$ with the length t^* and last but one vertex x^*.

Examine whether the path $\mu_{k^*}(s, x^*)$ contains the vertex w.
If yes, the walk $\mu_{k^*}(s, x^*) \oplus (x^*, (x^*, w), w^*)$ contains a cycle and therefore it is not a path.
GoTo Step4.

Step 2. Inserting path (s, w^*) into the set $\mathcal{D}$.

Set:

$$\begin{aligned} n\_deflab[w^*] &= n\_deflab[w^*] + 1 \\ k &= n\_deflab[w^*] \\ length[w^*][k] &= t^* \\ l\_b\_one[w^*][k] &= x^* \\ rankpp[w^*][k] &= k^* \end{aligned}$$

Step 3. Inserting extensions of the path (s, w^*) into the set $\mathcal{E}$.

For all vertices $u \in V^+(w^*)$ such that $n\_deflab[u] < K$
insert the quadruple $(u, t^* + c(w^*, u), w^*, k)$ into the set $\mathcal{E}$.

Step 4. Test of ending conditions.
If $n\_deflab[w] = K$ for all $w \in V$, STOP.
If $\mathcal{E} = \emptyset$, STOP.
Otherwise GoTo Step 1.

The k-th shortest (s, f)-path does exist if $n\_deflab[f] \geq k$ after the algorithm ends. The k-th shortest (s, f)-path $\mu_k(s, f) = (s \equiv v, (v1, v2), v2, \ldots, v_{l-1}, (v_{l-1}, v_l), v_l \equiv f$ can be calculated by making use of labels (7) as follows:

$$\begin{aligned}
v_l &= f \\
k_l &= k \\
v_{l-1} &= l_b_one[v_l][k_l] \\
k_{l-1} &= rankpp[v_l][k_l] \\
v_{l-2} &= l_b_one[v_{l-1}][k_{l-1}] \\
k_{l-2} &= rankpp[v_{l-1}][k_{l-1}] \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
v_2 &= l_b_one[v_3][k_3] \\
k_2 &= rankpp[v_3][k_3] \\
v_1 &= l_b_one[v_2][k_2] \\
k_1 &= rankpp[v_2][k_2]
\end{aligned}$$

The same procedure can be used in step 3. a) for checking whether the path (s,wmin) k contains the wertex w.

5 The complexity of proposed algorithm

Recall that $n = |V|$, $m = |A|$, suppose $m > n$. A quadruple (w, t, x, k) can be inserted into $\mathcal{E}$ at most $K.m$ times. In the case that is $\mathcal{E}$ organized as a Fibonacci heap all insertions into the set $\mathcal{E}$ require $O(K.m)$ steps. Before every insertion a cycle occurrence check is necessary. A single cycle occurrence check requires $O(n)$ steps; all cycle occurrence checks will require at most $O(K.m.n)$ steps. Every quadruple can be extracted from $\mathcal{E}$ at most once, a single extraction requires (in the case of Fibonacci heap) $O(\log(K.m))$ steps, all extractions will need at most $O(Km.\log(Km))$ steps. So the complexity of proposed algorithm is $O(Km(\log(Km) + n))$. Let us remark that we get the same complexity if $\mathcal{E}$ is organized as a binary heap.

6 Modification of algorithm for directed acyclic graphs - DAGs

There are many applications where the studied digraph $G = (V, A, c)$ is a directed acyclic graph – DAG. For example – the underlying digraph for CPM and PERT methods is DAG, the digraph used for optimum bus and/or train connections search is also DAG. Since there are no cycles in DAGs cycle check in Step 1. is not necessary – it can be skipped – and therefore the complexity of proposed algorithm is $O(Km(\log(Km))$.

Let us remark that the omitting the Step 1. in propsed algorithm leads in a digraph which is not acyclic to an algorithm which computes K shortest walks.

7 Conclusion

The proposed algorithm was used for bus and/or train connection search problem with great success. This algorithm was used also to compute shortest feasible trail with respect to prohibited maneuvers as presented by Tomáš Majer in his article on conference MME 2010 in České Budějovice in [3].

Acknowledgements

This research was supported by grant VEGA 1/0374/11 – Modelling and optimization of mobility and infrastructure in logistic networks.

References

[1] Gotthilfa, Z., Lewenstein, M.: Improved algorithms for the k simple shortest paths and the replacement paths problems, *Information Processing Letters*, Vol. **109**, Issue 7, (16 March 2009), 352–355.

[2] Martins, E., Q., W., Pascoal, M., M., B.. Santos, J., L., E.: A New Implementation of Yen's Ranking Looples Path Algorithm, *Investigacao Operacional*, (2000)

[3] Majer, T.: Shortest trail problem with respect to prohibited maneuvers *Mathematical methods in economics 2010 : proceedings of the 28th international conference, September 8-10. 2010, part II* České Budějovice, University of South Bohemia, (2010.) – ISBN 978-80-7394-218-2. 418–422.

[4] Plesnik, J.: *Grafov algoritmy – Graph Algorithms*, VEDA Bratislava, (1983)

[5] Yen, J., Y.: Finding k Shortest Loopless Paths in a Network, *Managements Science*, **17**, (1971), 712–760.

[6] Yen, J., Y.: Shortest Paths Network Problems, *Mathematical Systems in Economics*, Heft **18**, Meisennheim am Glan, (1975)