Proceedings of the Eastern Asia Society for Transportation Studies, Vol. 5, pp. 1209 - 1217, 2005

# AN ALGORITHM FOR LOGIT NETWORK LOADING PROBLEM BY TOPOLOGICAL SORTING

Jun LI
Dr. Eng.
School of Engineering
Sun Yat-sen University
135 Xingangxi Lu, Guangzhou,
Guangdong 510275, China
Fax: +86-20-84113689
E-mail: stslijun@zsu.edu.cn

Songxin XIN
School of Engineering
Sun Yat-sen University
135 Xingangxi Lu, Guangzhou,
Guangdong 510275, China
Fax: +86-20-84113689
E-mail: stits@zsu.edu.cn

Chunlu LIU
Dr. Eng.
School of Architecture and Building
Geelong Waterfront Campus
Deakin University
1 Gheringhap Street, Geelong
Victoria 3217, Australia
E-mail: chunlu@deakin.edu.au

**Abstract:** This paper presents a topological sorting based algorithm for logit network loading problem to exclude all cycles by removing certain links from loops. The new algorithm calculates the link weights and flows according to topological order. It produces the theoretical results for networks without loops. Numerical examples show that the new algorithm can reduce errors introduced by the strict definition of "reasonable route" in Dial's algorithm.

**Key Words:** Logit network loading, Dial's algorithm, Topological sort

## 1. INTRODUCTION

For networks with fixed costs, logit and probit stochastic user equilibrium models are two of the most important models. The models can be solved through the method of successive averages proposed by Powell and Sheffi (1982), both of them require a process of network loading. Up to now, simulation method remains the most feasible method to load network of probit model (Sheffi 1985). Dial (1971) proposed an algorithm for the logit network loading problem. The algorithm is very efficient and does not require path enumeration. However, it is found that the definition of "reasonable" paths in Dial's algorithm is so strict that it may lead to some problems. For example, some paths with smaller cost are excluded, while the larger ones are included in the reasonable paths (Li et al 2004). Akamatsu (1996, 1997) also shows tht Dial's algorithm fails in the real world application.

Bell (1995) proposed two algorithms for logit network loading problems. The first method considers any partial set of all possible paths, including all paths without loops and some of paths with loops, however the proof for equivalence between the model and logit model is not given. The second method, later proved by Akamatsu (1996), considers all possible paths which those paths with finite and infinite loops. So the number of paths is infinite. Li et al

(2004) proposed an iteration algorithm for this problem. In addition, Huang and Bell (1998) proposed an assignment excludes all cyclic flows, but it difficult to be applied to the large-scale networks because the algorithm requires path enumeration.

The remained part of this paper is organized as follows. Dial's algorithm is reviewed and the related issues are discussed in details. The a new algorithm based on topological sorting is presented, which excludes all cycles by removing certain links from loops only when it is necessary. The new algorithm loosens the definition of reasonable paths, improved the order of calculating, so as to reduce the errors of Dial's algorithm while retaining the efficiency of Dial's. Finally, numerical examples are given to test the new algorithm.

## 2. REVIEW AND ANALYSIS OF DIAL'S ALGORITHM

Consider an urban transportation network $D(N, A)$, where $N$ denotes set of nodes and $A$ denotes set of links. Let $rs$ be a given OD pair, where $r$ is the origin and $s$ is the destination. Let $r(i)$ denote the minimum travel time cost from origin node $r$ to node $i$, and $s(i)$ denote the minimum travel time from node $i$ to destination node $s$. Dial presented two methods, namely "double-pass" method and "single-pass" method. Only "single-pass" method is discussed in this paper, since definition of "reasonable path" in "double-pass" method is more strict than "single-pass" method.

The steps of Dial's algorithm for one OD pair $rs$ are outlined below. The total link flows can be obtained by repeating the steps for each OD pair in the network. First, compute the minimum travel time from node $r$ to all other nodes, obtain $r(i)$ for each node $i$. For each link $i \to j$ compute the "link likelihood" as follows

$$L_{(i\to j)} = \begin{cases} \exp\left[\theta \cdot \left(r(j) - r(i) - t_{(i\to j)}\right)\right] & \text{if } r(i) < r(j) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $\theta$ is a positive parameter measuring the sensitivity of route choices to travel cost, assumed to be given. Eq. (1) excludes all paths that include link $i \to j$ such that $r(i) \geq r(j)$ from "reasonable" paths. The nodes are then ordered in ascending order of $r(i)$, starting with the origin $r$. For each node $i$, the "link weight", $w_{(i\to j)}$, is given by

$$w_{(i\to j)} = \begin{cases} L_{(i\to j)} & \text{if} \quad i = r \\ L_{(i\to j)} \sum_{m\in I(i)} w_{(m\to i)} & \text{otherwise} \end{cases} \tag{2}$$

where $I(i)$ is the set of upstream nodes of all links arriving at node $i$. Now consider nodes in descending values of $r(i)$, starting with the destination, $s$. For each node $j$, compute the link flow $x_{(i\to j)}$ for each $i \in I(j)$ as follows:

$$x_{(i\to j)} = \frac{w_{(i\to j)}}{\sum_{m\in O(j)} w_{(m\to j)}} \cdot \left[ q_{rj} + \sum_{m\in I(j)} x_{(j\to m)} \right] \tag{3}$$

where $q_{rj}$ is the demand from the origin, $r$, to the node $j$; and $O(j)$ is the set of downstream

nodes of all links leaving node $j$. If $O(j)$ is null, then $x_{(i\to j)}$ is zero.

Let $P_{rs}$ be the set of all paths link from $r$ to $s$ including only such that $r(i) < r(j)$, then the probability that path $p$ is chosen for a trip from $r$ to $s$ is

$$\Pr(p) = \frac{\exp(-\theta \cdot c_p)}{\sum_{l \in P_{rs}} \exp(-\theta \cdot c_l)} \tag{4}$$

where $c_p$ is the travel time of path $p$. Eq. (4) is the definition of logit choice model among the reduced "reasonable paths".

Dial's algorithm uses Eq. (1) to exclude all the "unreasonable" links so that the remained network is acyclic, which ensures that as all variables in the right hand side of Eqs. (2) and (3) are either calculated or zero. However, the definition of "reasonable path" in either "single pass" or "double path" method is so strict that the links often used in real life are excluded. The issue limits the application of Dial's algorithm (Li et al, 2004). An improved algorithm will be presented in the next section.

Moreover, if one calculates link likelihood by removing "reasonable" constraints, namely, link likelihood is given by

$$L_{(i\to j)} = \exp\left[\theta \cdot \left(r(j) - r(i) - t_{(i\to j)}\right)\right] \tag{5}$$

Now solve Eq. (2) for link weights by assuming that link weights are unknown variables, and solve Eq. (3) for link flow by assuming that link flows are unknown variables, the resulted flow pattern is the same as all-path logit model discussed by Bell (1997) and Akamatsu (1996, 1997), see Li et al (2004) for more details.

## 3. IMPROVED DIAL'S ALGORITHM

Consider the network in Fig. 1, the flow on path $(1 \to 3 \to 2)$ is zero according to Dial's algorithm. The node computation order in the "forward pass" step of Dial's algorithm is $r-1-2-3-s$. It is not difficult to find that if the calculating order is changed to $r-1-3-2-s$ and remove the limit to the "reasonable path", namely, like likelihood is calculated by Eq.(5), the result will be identical to the theoretical results of logit choice model.

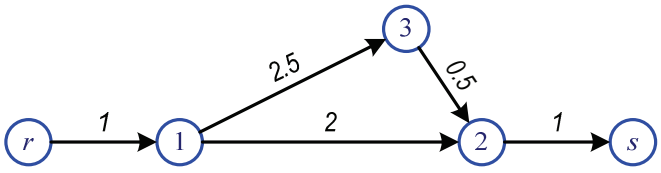


**Fig. 1 Example network 1**

The network in Fig. 1 is a directed graph containing no loops, and $r-1-3-2-s$ is exactly its topological order. As the illustration shown, it could produce the theoretical results for networks without loops in the topological order. However topological sort does not exist in a network with loops, so that this method cannot be applied. It is obvious that a network

containing loops can become acyclic by removing certain links which form loops, so that topological order can be applied to the reduced network. The key issue is what kinds of links should be removed from the network. As shown before, "unreasonable links" are removed in Dial's algorithm. To solve this issue, the new algorithm only removes links when they are parts of loops. For example, link $(3 \to 2)$ is defined as "unreasonable" link according Dial's algorithm, however, it should be kept as it not part of loop.

First of all, define a state variable $\xi_{(i\to j)}$ for each link $(i \to j)$. If the link has not been processed, then $\xi_{(i\to j)} = 0$; if is removed from the network, then $\xi_{(i\to j)} = -1$ and $L_{(i\to j)} = 0$; otherwise $\xi_{(i\to j)} = 1$, and its "link likelihood" is given by

$$L_{(i\to j)} = \exp\left[\theta \cdot \left(s(i) - s(j) - t_{(i\to j)}\right)\right] \tag{6}$$

Note that the "link likelihood" is calculated according to $s(i)$ rather than $r(i)$, the reason will be discussed later. Moreover, it is easy to show that the use of $s(i)$ or $r(i)$ to calculate link likelihoods has no impact on the proof of equivalence between proposed model and logit assignment formulation.

For each OD pair *rs*, the steps to establish the calculating order and the state of the links are outlined below.

**Step 0:**
Set state variable for each link to 0, i.e., $\xi_{(i\to j)} = 0$ for each link $(i \to j)$. Let $K = \{r\}$ be a node set containing origin *r*, and *Q* be an empty queue. Delete all the links that satisfy one of the following conditions from the network, namely set link state variable to $\xi_{(i\to j)} = -1$:

(1) links arriving at origin *r*, i.e., links such that $(j \to r)$ ;
(2) links leaving destination *s*, i.e., links such that $(s \to i)$ ;
(3) links that do not connect origin or destination, i.e., links connecting to nodes such that $r(i) = \infty$ or $s(i) = \infty$.

**Step 1:**
If *K* is empty, stop, *Q* is the topological order and the state variable $\xi_{(i\to j)}$ for each link $(i \to j)$ is set to either 1 or -1. If *K* is not empty, go to Step 2.

**Step 2:**
Find one node *i* from *K*, where all the links arriving at node *i* have been deal with, i.e. $\xi_{(j\to i)} \neq 0$. Delete *i* from *K*, and append *i* to the end of queue *Q*. For each link $(i \to j)$, if the state variable $\xi_{(i\to j)} = 0$, then set $\xi_{(i\to j)} = 1$, and add *j* into set *K*, and go to Step 1. If one cannot find such node in *K*, which indicates the network contains cycles, then go to Step 3.

**Step 3:**
Find one node *i* from *K* according to the following criteria: (1) find node with maximum $s(\cdot)$; (2) if there are more than one node satisfy (1), then select node with minimum $r(\cdot)$; (3) if there are more than one node satisfy then a random node is selected. For each link $(j \to i)$, set $\xi_{(j\to i)} = -1$ if $\xi_{(j\to i)} = 0$ and keep $\xi_{(j\to i)}$ unchanged if $\xi_{(j\to i)} = 1$. Go to Step 2.

Step 0 indicates that the users who depart origin cannot return to origin, and users who arrive at destination cannot leave. It also deletes all nodes that do not connect to origin or destination. It can be seen that for a network with no loop, the aforementioned method is identical to find topological order and Step 3 is never utilized. For networks with loops, the networks are reduced to acyclic networks by Step 3. The criteria for selecting node to remove its unprocessed upstream links in Step 3 are understandable, because generally those links have longer travel time than other links. Step 3 guarantees that Step 2 is always feasible, since there exists at least one node whose stat variable either equals to 1 or -1. Moreover because $s(s) = 0$ for the destination *s*, so it is the latest one in the topological sequence, therefore the topological sequence starts from the origin *r* and ends at the destination *s*. The node computing order ensures that users travel closer to the destination when loops exist. If there exist two or more nodes with the same value of $s(\cdot)$, one can choose node with smaller $r(\cdot)$ so that users will travel further away from origin loops exist. Generally, $s(\cdot)$ is a real number so that it seldom occurs that two values of $s(\cdot)$ are identical, so that that choosing node according to $r(\cdot)$ can be dropped. This is the reason why Eq. (6) utilizes $s(\cdot)$ rather than $r(\cdot)$ to calculate link likelihoods. In such case, only one calculation of shortest paths is needed, the resulted method is similar to the "single-pass" method of Dial's algorithm so that the computation requirement can be halved.

Once node computing order is found, the forward pass and backward pass of Dial's algorithm can be utilized to calculate the link weight and flow. The steps of this algorithm for one OD pair *rs* are outlined below. These steps should be repeated for each OD pair in the network to get final link flows.

**Step 0 (Preliminary):**
Compute the minimum travel time from each node *i* to the destination *s*, and the minimum travel time from origin *r* to each node *i*.

**Step 1**
Determine the node computing order and state variable for each link by the aforementioned method. Then calculate the "link likelihood" by Eq. (6).

**Step 2 (Forward pass)**
Calculate the link weight in ascending topological order by Eq. (2). Note that for each link $(i \to j)$ such that $\xi_{(i\to j)} = 0$, $w_{(i\to j)} = 0$.

**Step 3 (Backward pass)**
Set all link flow to zero, $x_{(i\to j)} = 0$. Update the link flow in descending topological order:

$$x_{(i\to j)} = \begin{cases} \dfrac{w_{(i\to j)}}{\sum_{m\in I(j)} w_{(m\to j)}} \cdot q_{rs} & \text{if } j = s \\ \dfrac{w_{(i\to j)}}{\sum_{m\in I(j)} w_{(m\to j)}} \cdot \left[ \sum_{m\in O(j)} x_{(j\to m)} \right] & \text{otherwise} \end{cases} \tag{7}$$

It is clear that there is no flow on a link $(i \to j)$ such that $\xi_{(i\to j)} = 0$, i.e., $x_{(i\to j)} = 0$.

In "forward pass", all variable in the right hand side of Eq. (2) are either calculated or set to be zero, therefore no loop is considered in Eq. (2). The same conclusion holds for "backward pass". By removing those links which are assumed to have longer travel time, the resulted network is acyclic and topological order can be applied.

Since only the set of "reasonable paths" is changed, the proof of the equivalence between the new algorithm and logit assignment formulation is similar to Dial's algorithm. The readers are referred Dial (1971) for details.

Assume that network $D(V, A)$ has $m$ nodes and $n$ links. The complexity of Dijkstra's algorithm to calculate the shortest paths by priority queue is $O((m+n)\log m)$, and the complexity of topological sorting is $O(m+n)$. Therefore, the complexity of new algorithm is $O((m+n)\log m)$. Thus the new algorithm can be easily applied to large-scale networks.

## 4. EXAMPLE NETWORK FOR APPLYING NEW ALGORITHM

To test the new algorithm, a $3 \times 4$ grid network is used, as shown in Fig. 2. The network contains 12 nodes and 34 links. Node *A* is the origin and node *L* is the destination. Assume that the OD flow equals 1000. Fig. 2 gives the link travel time for each link. Assume that logit parameter is $\theta = 1$.

The link flows calculated by different algorithms are given in Fig. 3, where the theoretical result is calculated by enumerating all the paths without loops. Note that only 1 path is "reasonable" according to Dial's algorithm, i.e. A-B-C-D-H-L, as shown in Fig. 4; while 10 paths are "reasonable" in the proposed algorithm, as shown in Fig. 5. Comparing with the theoretical result, the error of the new algorithm is about 3.5%, while the error of Dial's algorithm is about 50%. The proposed algorithm produces significantly better assignment result than that of Dial's algorithm in this example.

The node computing order of proposed algorithm is *A-E-I-B-F-C-J-D-G-H-K-L*. If only selecting node from *K* according to $s(i)$, the order of node *B* and node *F* cannot be determined because $s(B) = s(F) = 4$. The error increases to 16% if node *F* is calculated before node *B*. while the error is 3.5% when node *B* is computed before node *F*. In this example, the assignment result is optimized if the node computing order is further tuned by $r(i)$. If computer power is not concerned, "double-pass" method is recommended over "single-pass" method.

If the costs of each link in the network are the same, say, all link costs are set to be 2, the "reasonable paths" of both Dial's algorithm and the proposed algorithm are identical, which are the same as those shown in Fig. 5. Of course, the flow patterns will be identical. However, numerical examples show that the proposed algorithm will produce more "reasonable paths" than Dial's algorithm in most cases.

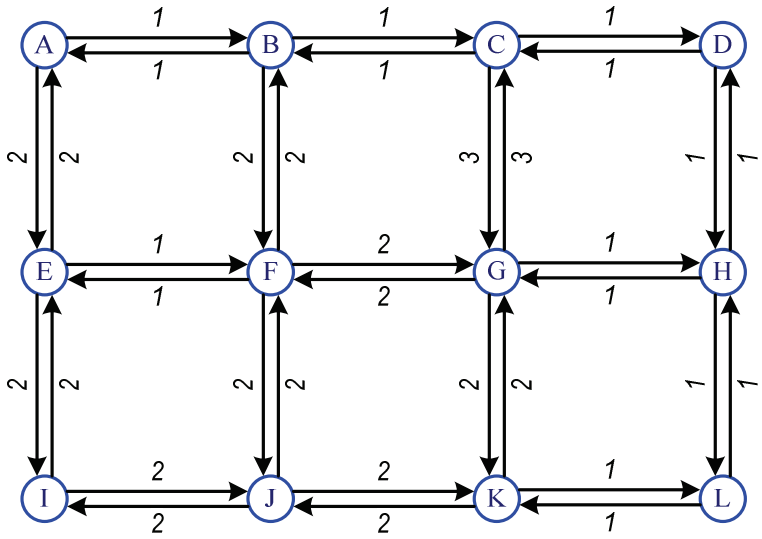


**Fig. 2. Example network 2**

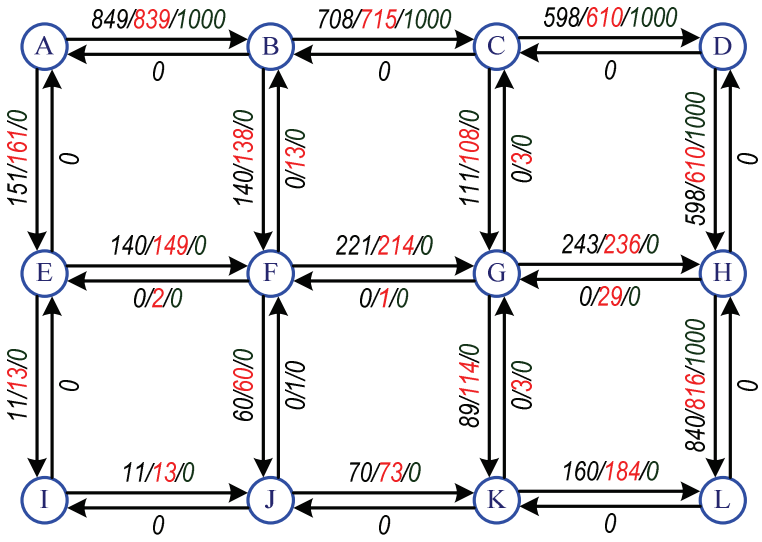


**Fig. 3. The flow patterns for different algorithms (New/Theoretic/Dial)**

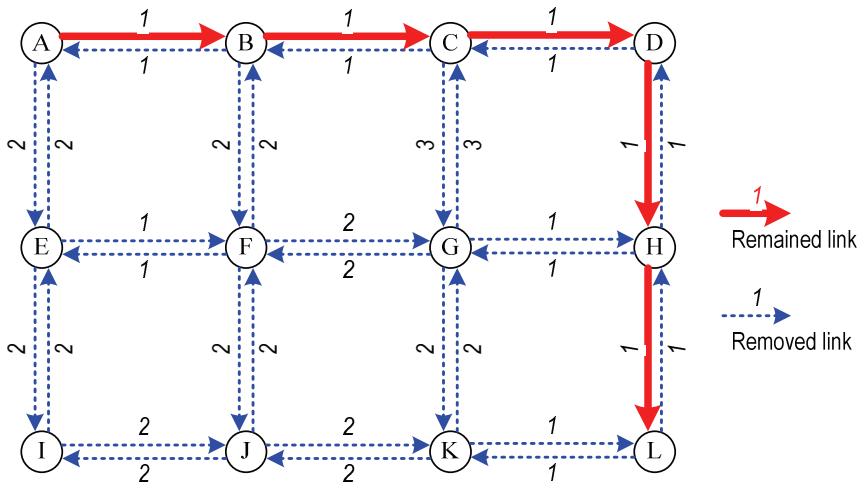


**Fig. 4. "Reasonable" path of Dial's algorithm**

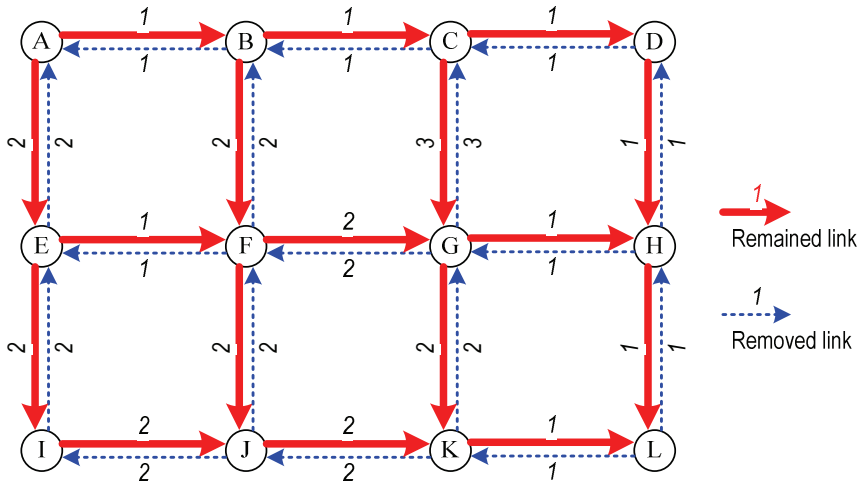


**Fig. 5. "Reasonable" path of proposed algorithm**

## 5. CONCLUSION

In this paper, a topological sorting based algorithm for logit network loading problem is presented. The algorithm removes certain links when they are part of loops so that the remained network is acyclic. Numerical examples shows that most of "reasonable" paths in common sense are remained, the number of the remained "reasonable" paths are significant larger than that of Dial's algorithm in the test network. The probability to utilize the removed paths is very small. The efficiency of Dial's algorithm is remained in the proposed algorithm.

## ACKNOWLEDGEMENTS

This research is sponsored by the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry.

## REFERENCES

Dial R. B. (1971) A probabilistic multipath traffic assignment model which obviates path enumeration, **Transportation Research**, Vol. 5, 83-111.

Bell, G. H. (1995) Alternatives to Dial's LOGIT Assignment Algorithm, **Transportation Research**, Vol. 29B, 287-296.

Akamatsu, T. (1996) Cyclic Flows, Markov Process and Stochastic Traffic Assignment, **Transportation Research**, Vol. 30B, 369-386.

Akamatsu, T. (1997) Decomposition of path choice entropy in general transport networks, **Transportation Science**, Vol. 31, 349-362.

Huang, H. and Bell, M. G. H. (1997) A Study on Logit Assignment Which Excludes All Cyclic

Flows, **Transportation Research**, 1997, 29B, 401-412.

Li, J., Nie, P. L. and Yu, Z. (2004) Solving all-path logit assignment problems by iteration method, **Acta Scientiarum Naturalium Niversitatis Sunyatseni**, Vol. 43, No. 5, 124-126. (in Chinese)

Sheffi, Y. (1985) **Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods**. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.