Extend the quickest path problem to the system reliability evaluation for a stochastic-flow network

Yi-Kuei Lin*

Department of Information Management, Van Nung Institute of Technology, Chung-Li, Tao-Yuan, Taiwan 320, Republic of China

Received 1 March 2001; received in revised form 1 August 2001; accepted 1 November 2001

Abstract

From the point of view of quality management, it is an important issue to reduce the transmission time in the network. The quickest path problem is to find the path in the network to send a given amount of data from the source to the sink such that the transmission time is minimized. Traditionally, this problem assumed that the capacity of each arc in the network is deterministic. However, the capacity of each arc is stochastic due to failure, maintenance, etc. in many real-life networks. This paper proposes a simple algorithm to evaluate the probability that \( d \) units of data can be sent from the source to the sink through the stochastic-flow network within \( T \) units of time. Such a probability is called the system reliability. The proposed algorithm firstly generates all lower boundary points for \((d, T)\) and the system reliability can then be computed in terms of such points.

Scope and purpose

The shortest path problem is a well-known problem in operations research, computer science, etc. Chen and Chin have proposed a variant of the shortest path problem, termed the quickest path problem. It is to find a path in the network to send a given amount of data from the source to the sink with minimum transmission time. More specifically, the capacity of each arc in the network is assumed to be deterministic. However, in many real-life networks such as computer systems, telecommunication systems, etc., the capacity of each arc is stochastic due to failure, maintenance, etc. Such a network is named a stochastic-flow network. Hence, the minimum transmission time is not a fixed number. This paper proposes a simple algorithm to evaluate the probability that the specified amount of data can be sent from the source to the sink through the network within a given time. Such a probability is called the system reliability. © 2002 Elsevier Science Ltd. All rights reserved.

* Tel.: +886-3-4515811; fax: +886-3-4621348.
E-mail address: yklin@cc.vit.edu.tw (Y.-K. Lin).
1. Introduction

The shortest path problem is one of the most important problems in operations research, computer science, networking and other areas. When data (goods or commodities) are transmitted from one node to another node in a flow network, it is desirable to adopt the shortest path, least cost path, largest capacity path, shortest delay path, or some combination of multiple criteria [1–4], which are all variants of the shortest path problem. From the point of view of quality management, it is an important issue to reduce the transmission time in the network. Hence, a version of the shortest path problem, which is called the quickest path problem, is proposed by Chen and Chin [5]. Basically, this problem is to find a path (named quickest path) with minimum transmission time to send a given amount of data from the source to the sink, where each arc has two attributes; the capacity and the lead time [5–7]. More specifically, the capacity and the lead time of each arc are both assumed to be deterministic. Since then, several variants of quickest path problems are proposed; constrained quickest path problem [8,9], k quickest path problem [9–11] and all-pairs quickest path problem [12,13].

However, due to failure, maintenance, etc., the capacity of each arc is stochastic in many real flow networks such as computer systems, telecommunication systems, urban traffic systems, logistics systems, etc. Such a network is named a stochastic-flow network [14–22]. The purpose of this paper is mainly to extend the quickest path problem from the deterministic case to the stochastic case. We evaluate the probability that the stochastic-flow network can send $d$ units of data from the source to the sink within a given time $T$. Such a probability is named the system reliability throughout this paper. A simple algorithm based on minimal paths (MPs) is proposed firstly to find all lower boundary points for $(d;T)$, and then to calculate the system reliability in terms of such points, where a MP is an ordered sequence of arcs from the source to the sink without loops, and a lower boundary point for $(d;T)$ is a vector representing the capacity of each arc. The algorithm and an illustrative example are presented in Sections 3 and 4, respectively.

2. Assumptions and nomenclatures

Let $G \equiv (N,A,L,M)$ denote a stochastic-flow network with a source $s$ and a sink $t$ where $N$ denotes the set of nodes, $A \equiv \{a_i|1 \leq i \leq n\}$ denotes the set of arcs, $L \equiv (l_1,l_2,\ldots,l_n)$ with $l_i$ denoting the lead time of $a_i$ and $M \equiv (M_1,M_2,\ldots,M_n)$ with $M_i$ denote the maximal capacity of $a_i$. The capacity is the maximal number of data sent through the medium (an arc or a path) per unit of time. The (current) capacity of arc $a_i$, denoted by $x_i$, takes values $0 = b_{i1} < b_{i2} < \cdots < b_{iri} = M_i$, where $b_{ij}$ is an integer for $j = 1,2,\ldots,r_i$. The vector $X \equiv (x_1,x_2,\ldots,x_n)$ denotes the capacity vector of $G$. Such a $G$ is assumed to further satisfy the following assumptions:

(1) Each node is perfectly reliable.
(2) The capacity of each arc is stochastic with a given probability distribution.
(3) The capacities of different arcs are statistically independent.
(4) All data are sent through one MP.

\[ Y \succ X \equiv (y_1, y_2, \ldots, y_n) \succ (x_1, x_2, \ldots, x_n): y_i \succ x_i \text{ for each } i = 1, 2, \ldots, n. \]

\[ Y \succ X \equiv (y_1, y_2, \ldots, y_n) \succ (x_1, x_2, \ldots, x_n): Y \succ X \text{ and } y_i > x_i \text{ for at least one } i. \]

3. The algorithm

Suppose that \( P_1, P_2, \ldots, P_m \) are MPs of \( G \) from \( s \) to \( t \). With respect to each MP \( P_j = \{a_{j1}, a_{j2}, \ldots, a_{jn}\} \), \( j = 1, 2, \ldots, m \), the capacity of \( P_j \) under the capacity vector \( X \) is \( \min_{1 \leq k \leq n} (x_{jk}) \). If \( d \) units of data are required to be transmitted through \( P_j \) under the capacity vector \( X \), then the transmission time, denoted by \( \psi(d, X, P_j) \), is

\[ \text{the lead time of } P_j + \left\lfloor \frac{d}{\text{the capacity of } P_j} \right\rfloor = \sum_{k=1}^{n_j} l_{jk} + \left\lfloor \frac{d}{\min_{1 \leq k \leq n_j} x_{jk}} \right\rfloor, \]

where \( \left\lfloor x \right\rfloor \) is the smallest integer such that \( \left\lfloor x \right\rfloor \geq x \). Let \( \zeta(d, X) \) denote the minimum transmission time for the network to send \( d \) units of data from \( s \) to \( t \) under the capacity vector \( X \). Then \( \zeta(d, X) = \min_{1 \leq j \leq m} \psi(d, X, P_j) \).

3.1. Definition of lower boundary points for \( (d, T) \)

Any capacity vector \( X \) with \( \zeta(d, X) \leq T \) means that the network can send \( d \) units of data from \( s \) to \( t \) within \( T \) units of time under \( X \). If \( X \) is a minimal capacity vector such that the network can send \( d \) units of data from the source to the sink within \( T \) units of time, then \( X \) is called a lower boundary point for \((d, T)\). That is, \( X \) is a lower boundary point for \((d, T)\) if and only if (i) \( \zeta(d, X) \leq T \) and (ii) \( \zeta(d, Y) > T \) for any capacity vector \( Y \) with \( Y < X \). Hence, the system reliability \( R_{d,T} \) to meet such a requirement is \( \Pr\{X|\zeta(d, X) \leq T\} \). Lemma 1 implies that \( \Pr\{X|\zeta(d, X) \leq T\} = \Pr\{X|X \geq X_j \text{ for a lower boundary point } X_j \text{ for } (d, T)\} \). Several methods such as inclusion-exclusion rule [16–20,23], disjoint-event method [23,24] and state-space decomposition [14,15,25,26] can be applied to calculate \( \Pr\{X|X \geq X_j \text{ for a lower boundary point } X_j \text{ for } (d, T)\} \). Note that \( \Pr\{X \geq Y\} = \Pr\{x_1 \geq y_1\} \times \Pr\{x_2 \geq y_2\} \times \cdots \times \Pr\{x_n \geq y_n\} \) by assumption 3 if \( Y = (y_1, y_2, \ldots, y_n) \).

Lemma 1. If \( X \) is a lower boundary point for \((d, T)\), then \( \zeta(d, Y) \leq T \) for any \( Y \geq X \).

3.2. The algorithm to evaluate the system reliability

As those algorithms in [15–17,19–22], the proposed algorithm supposes that all MPs have been precomputed. MPs can be efficiently derived from those algorithms discussed in [27–29]. The algorithm of Al-Ghanim [27] showed an approximate linear time response versus the number of network nodes. And for a network of 101 nodes, including 10 branch nodes, the computational time per path is 0.518 s (run on a PC 486 machine). Kobayashi and Yamamoto [28] showed that to generate all minimal paths for a random network with 30 nodes and 100 arcs takes no more than 1300 s.
Lemma 2. The set of lower boundary points for \((d, T)\) is the set of \(X_1, X_2, \ldots, X_m\), where \(X_j\), \(j = 1, 2, \ldots, m\), is generated from the algorithm and exists.

Proof. We first claim that any obtained \(X_j\) is a lower boundary point for \((d, T)\). (i) It is trivial that \(\zeta(d, X_j) \leq T\) since \(\psi(d, X, P_j) \leq T\) and \(\psi(d, X, P_k) = \infty\) for each \(k \neq j\). (ii) If \(Y = (y_1, y_2, \ldots, y_n) < X_j = (x_1, x_2, \ldots, x_n)\), then there exists an arc \(a_u \in P_j\) such that \(y_u < x_u\). Then the capacity of \(P_j\) under \(Y\) is less than that under \(X\). Hence, \(\psi(d, Y, P_j) > T\) and \(\zeta(d, Y) > T\). By (i) and (ii), we know each \(X_j\) is a lower boundary point for \((d, T)\). Conversely, we claim that any lower boundary point for \((d, T)\) belongs to \(\{X_1, X_2, \ldots, X_m\}\). Let \(X\) be a lower boundary point for \((d, T)\). Without loss of generality for \(P_j\), we assume that \(\psi(d, X, P_j) \leq T\) and \(\psi(d, X, P_k) > T\) for each \(k \neq j\). Suppose to the contrary that \(X \notin \{X_1, X_2, \ldots, X_m\}\), then there exists an arc \(a_i \notin P_j\) such that \(x_i > 0\). Set \(Y = (x_1, x_2, \ldots, x_i - z, \ldots, x_n)\), where \((x_i - z)\) is the maximal capacity of \(a_i\) such that \((x_i - z) < x_j\). Then \(\psi(d, Y, P_j) \leq T\) and \(\psi(d, Y, P_k) > T\) for each \(k \neq j\). That contradicts to that \(X\) is a lower boundary point for \((d, T)\). Hence, any lower boundary point for \((d, T)\) belongs to \(\{X_1, X_2, \ldots, X_m\}\). \(\square\)

4. An illustrative example

We use the benchmark network in Fig. 1 [5,8,12] to illustrate the proposed algorithm. The capacity and the lead time of each arc are both shown in Table 1. There are six MPs: \(P_1 = \{a_1, a_4\}\), \(P_2 = \{a_1, a_5, a_8\}\), \(P_3 = \{a_1, a_2, a_6\}\), \(P_4 = \{a_1, a_2, a_7, a_8\}\), \(P_5 = \{a_2, a_6\}\) and \(P_6 = \{a_3, a_7, a_8\}\). If 8 units of data are required to be sent from \(s\) to \(t\) within 9 units of time. Then all lower boundary points for \((8, 9)\) and the system reliability \(R_{8,9}\) to meet such a requirement can be derived as follows:

Step 1: (1.1). The lead time of \(P_1 = \{a_1, a_4\}\) is \(l_1 + l_4 = 5\). Then \(v = 2\) is the smallest integer such that \((5 + \left\lceil \frac{2}{v} \right\rceil) \leq 9\).
Fig. 1. A benchmark network.

Table 1
The arc data of Fig. 1

<table>
<thead>
<tr>
<th>Arc</th>
<th>Capacity</th>
<th>Probability</th>
<th>Lead time</th>
<th>Arc</th>
<th>Capacity</th>
<th>Probability</th>
<th>Lead time</th>
</tr>
</thead>
<tbody>
<tr>
<td>a₁</td>
<td>3</td>
<td>0.80</td>
<td></td>
<td>a₂</td>
<td>2</td>
<td>0.10</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.10</td>
<td>2</td>
<td>a₆</td>
<td>2</td>
<td>0.10</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0.05</td>
<td></td>
<td></td>
<td>1</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0.05</td>
<td></td>
<td></td>
<td>0</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>a₃</td>
<td>3</td>
<td>0.80</td>
<td></td>
<td>a₇</td>
<td>3</td>
<td>0.10</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.10</td>
<td>1</td>
<td></td>
<td>2</td>
<td>0.10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0.05</td>
<td></td>
<td></td>
<td>0</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>a₄</td>
<td>1</td>
<td>0.90</td>
<td>3</td>
<td>a₈</td>
<td>2</td>
<td>0.10</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0.10</td>
<td></td>
<td></td>
<td>1</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>a₅</td>
<td>1</td>
<td>0.90</td>
<td>1</td>
<td></td>
<td>0</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0.10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Pr{the capacity of a₁ is 3} = 0.80.

(1.2). The maximal capacity of P₁ is only 1. Hence, X₁ does not exist.
(1.1). The lead time of P₂ = {a₁,a₅,a₈} is l₁ + l₅ + l₈ = 4. Then v = 2 is the smallest integer such that \(4 + \left\lceil \frac{8}{v} \right\rceil \leq 9\).

(1.2). The maximal capacity of P₂ is only 1. Hence, X₂ does not exist.
(1.1). The lead time of P₃ = {a₁,a₂,a₆} is l₁ + l₂ + l₆ = 5. Then v = 2 is the smallest integer such that \(5 + \left\lceil \frac{8}{v} \right\rceil \leq 9\).

(1.2). The maximal capacity of P₃ is 3. Hence, x₁ = x₂ = x₆ = 2 and xᵢ = 0 for others. So we obtain X₃ = (2,2,0,0,0,2,0,0).
(1.1). The lead time of P₄ = {a₁,a₂,a₇,a₈} is l₁ + l₂ + l₇ + l₈ = 6. Then v = 3 is the smallest integer such that \(6 + \left\lceil \frac{8}{v} \right\rceil \leq 9\).
(1.2). The maximal capacity of \( P_4 \) is 3. Hence, \( x_1 = x_2 = x_7 = x_8 = 3 \) and \( x_i = 0 \) for others. So we obtain \( X_4 = (3, 3, 0, 0, 0, 0, 3, 3) \).

(1.1). The lead time of \( P_5 = \{a_3, a_6\} \) is \( l_3 + l_6 = 5 \). Then \( v = 2 \) is the smallest integer \( v \) such that \( \left( 5 + \left\lceil \frac{8}{5} \right\rceil \right) \leq 9 \).

(1.2). The maximal capacity of \( P_5 \) is 2. Hence, \( x_3 = x_6 = 2 \) and \( x_i = 0 \) for others. So we obtain \( X_5 = (0, 0, 2, 0, 0, 2, 0, 0) \).

(1.1). The lead time of \( P_6 = \{a_3, a_7, a_8\} \) is \( l_3 + l_7 + l_8 = 6 \). Then \( v = 3 \) is the smallest integer such that \( \left( 6 + \left\lceil \frac{8}{5} \right\rceil \right) \leq 9 \).

(1.2). The maximal capacity of \( P_6 \) is only 2. Hence, \( X_6 \) does not exist.

Step 2: Three lower boundary points for \((8, 9)\) are generated by step 1. Let \( B_3 = \{X|X \geq X_3\} \), \( B_4 = \{X|X \geq X_4\} \) and \( B_5 = \{X|X \geq X_5\} \). The system reliability \( R_{8, 9} = \Pr \{B_3 \cup B_4 \cup B_5\} = 0.91275 \) by applying inclusion–exclusion rule. In the calculating process,

\[
\Pr\{B_3\} = \Pr\{X \geq (2, 2, 0, 0, 0, 2, 0, 0)\} = \Pr\{x_1 \geq 2\} \times \Pr\{x_2 \geq 2\} \times \Pr\{x_3 \geq 0\} \times \Pr\{x_4 \geq 0\} \times \Pr\{x_5 \geq 0\}
\]

\[
\Pr\{x_6 \geq 2\} \times \Pr\{x_7 \geq 0\} \times \Pr\{x_8 \geq 0\}
\]

\[
= 0.9 \times 0.9 \times 1 \times 1 \times 0.9 \times 1 \times 1 = 0.729,
\]

\[
\Pr\{B_3 \cap B_4\} = \Pr\{(X \geq (2, 2, 0, 0, 0, 2, 0, 0)) \cap (X \geq (3, 3, 0, 0, 0, 0, 3, 3))\}
\]

\[
= \Pr\{X \geq (3, 3, 0, 0, 0, 2, 3, 3)\} = 0.3456,
\]

\[
\Pr\{B_3 \cap B_4 \cap B_5\} = \Pr\{(X \geq (2, 2, 0, 0, 0, 2, 0, 0)) \cap (X \geq (3, 3, 0, 0, 0, 0, 3, 3)) \cap (X \geq (0, 0, 2, 0, 0, 2, 0, 0))\} = \Pr\{X \geq (3, 3, 2, 0, 0, 2, 3, 3)\} = 0.29376.
\]

If \( d = 8 \) and \( T \) is loosened to be 12, then five lower boundary points for \((8, 12)\) are generated:

\( X_2 = (1, 0, 0, 0, 0, 1, 0, 0, 1) \), \( X_3 = (2, 2, 0, 0, 0, 0, 2, 0, 0) \), \( X_4 = (2, 2, 0, 0, 0, 0, 2, 2) \), \( X_5 = (0, 0, 2, 0, 0, 2, 0, 0) \) and \( X_6 = (0, 0, 2, 0, 0, 0, 2, 2) \). The system reliability \( R_{8, 12} \) increases to 0.987359625.

5. Discussion, conclusions and further research

In Step 1, each minimal path needs \( O(n) \) time to execute both constraint (1) and Eq. (2) in the worst case, respectively. Hence, step 1 takes \( O(mn) \) time in the worst case. Step 2 needs \( O(m^2n) \) time to evaluate the system reliability in the worst case by applying disjoint-event method [23,24]. The algorithm is run for 5 cases: 10 nodes with 15 arcs, 15 nodes with 25 arcs, 20 nodes with 30 arcs, 25 nodes with 40 arcs and 30 nodes with 46 arcs. For each case, we generate seven random networks. Table 2 shows the CPU time of the proposed algorithm programmed in C language. All programs were executed on IBM Netfinity-3500 with Red Hat Linux 6.1 operation system.

This paper uses the capacity vector \( X \) and MPs to describe the stochastic-flow network. If \( d \) units of data are required to be sent from the source to the sink through the stochastic-flow network
Table 2
CPU time (s) for five cases

<table>
<thead>
<tr>
<th>No. of nodes</th>
<th>No. of arcs</th>
<th>Step</th>
<th>Random network</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>15</td>
<td>Step 1</td>
<td>0a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Step 2</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Total</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>25</td>
<td>Step 1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Step 2</td>
<td>0.030</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Total</td>
<td>0.030</td>
</tr>
<tr>
<td>20</td>
<td>30</td>
<td>Step 1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Step 2</td>
<td>0.174</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Total</td>
<td>0.174</td>
</tr>
<tr>
<td>25</td>
<td>40</td>
<td>Step 1</td>
<td>0.005</td>
</tr>
<tr>
<td>30</td>
<td>46</td>
<td>Step 1</td>
<td>0.024</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Step 2</td>
<td>150.072</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Total</td>
<td>150.096</td>
</tr>
</tbody>
</table>

*a0 means that the computational time is fewer than 0.01 s.

within \( T \) units of time, then an algorithm is proposed to evaluate the system reliability to meet such a requirement. The idea of lower boundary points for \((d, T)\) is proposed, which are the minimal capacity vectors satisfying the requirement. At most \( m \) lower boundary points for \((d, T)\) are generated if there are \( m \) minimal paths. The system reliability can be computed in terms of all lower boundary points for \((d, T)\). Given a \( T, \sum_{d} R_{d,T} \times d \) is the expected amount of data sent from \( s \) to \( t \) within \( T \) units of time. From the point of view of quality management, we can treat the system reliability as a performance index, and conduct the sensitive analysis to improve the most important component (e.g., switch or server in computer network) which will increase the system reliability significantly.

Future research can extend the problem to the case that the data are sent through several MPs simultaneously. Similarly, to develop a process to evaluate the system reliability for a stochastic-flow network to send \( d \) units of data from the source to the sink within \( T \) units of time. Moreover, researchers can discuss the constrained quickest path problem, \( k \) quickest paths problem and all-pairs quickest path problem for a stochastic-flow network.

Acknowledgements

I would like to thank the anonymous referees for helpful and constructive comments. This work was supported in part by the National Science Council, Taiwan, Republic of China, under Grant No. NSC 90-2218-E-238-002.
References

Yi-Kuei Lin is currently a Professor in the Department of Information Management, Van Nung Institute of Technology, Taiwan. He received his Bachelor degree in Applied Mathematics from National Chiao Tung University, and his Master degree and Ph.D. degree from National Tsing Hua University, all in Taiwan. His research interest includes stochastic network reliability, telecommunication management, multimedia communications and operations research. He has published papers in *Computers & Operations Research; Reliability Engineering & System Safety; Computers and Mathematics with Applications; Journal of the Operations Research Society of Japan; International Journal of Reliability, Quality and Safety Engineering; Journal of the Chinese Institute of Engineers and Journal of the Chinese Institute of Industrial Engineers.*