International Journal of Reliability, Quality and Safety Engineering
Vol. 24, No. 1 (2017) 1750005 (12 pages)

DOI: 10.1142/S021853931750005X

A Merge Search Approach to Find Minimal Path Vectors in Multistate Networks

Yi-Kuei Lin

Department of Industrial Management
National Taiwan University of Science and Technology
Taipei, Taiwan 106, R. O. C.
yklin@mail.ntust.edu.tw

Shin-Guang Chen*

Department of Industrial Management
Tungnan University, 152, Sec. 3, Bei-Shen Rd.
Shen-Ken Dist., New Taipei City, Taiwan 222, R. O. C.
bobchen@mail.tnu.edu.tw

Received 15 May 2016
Revised 2 August 2016
Accepted 10 October 2016
Published 15 November 2016

A new approach namely merge search (MS) is proposed to search for minimal path vectors (MPV) in multistate networks (MSN). Also, a new advance in solving integer programming problems namely fast enumeration (FE) is integrated in this approach. Such an integrated approach can greatly improve the time efficiency of searching for MPV in MSN. Traditionally, searching for MPV in MSN involves three steps: (a) enumerate all feasible flow vectors; (b) transform these vectors to corresponding state vectors; (c) filter out MPV from these state vectors. Steps (a) and (c) are bottlenecks. Explicit enumeration is usually engaged in solving Step (a), and pairwise comparison is usually employed in solving Step (c). The integrated approach uses FE to solve Step (a), and MS to solve Step (c) instead. Some numerical examples are explored to show the superior time efficiency of the proposed approach. The results show that the proposed new approach is valuable in solving the search of MPV in MSN.

Keywords: Minimal path vector (MPV); multistate network; merge search; fast enumeration (FE); minimal path.

1. Introduction

Aggarwal *et al.* firstly discussed the reliability problem without flow in a binary-state network.[1] The network reliability is the probability of a live connection between source nodes and sink nodes. Lee[2] extended it to cover the flow cases. Then, the reliability is the probability of a live connection between source nodes and sink

*Corresponding author.

nodes with the maximal flow no less than a demand d. Aggarwal *et al.*[3] presented the minimal path (MP) method to solve the network reliability in a binary-state flow network. A path is a set of nodes and edges whose existence results in the connection of one source node and one sink node. A MP is a path whose proper subset is not a path. Rueger[4] extended it to cover the node failure cases. El-Neweihi *et al.*[5] firstly discussed the multistate systems (MSN). Xue[6] began to discuss the reliability analysis in a multistate network (MSN), which is also referred to the stochastic-flow network. Lin *et al.*[7] illustrated the reliability calculation of a MSN in terms of MP or minimal cuts (MC).[8] A cut is a set of edges whose removal results in the disconnection of all source nodes and all sink nodes. A MC is a cut whose proper subset is no longer a cut. They also showed the basic stages in this calculation: (a) searching for all MP[9–11]/MC[12–15]; (b) searching for all minimal path vectors (MPV)[16]/minimal cut vectors (MCV)[17] from these MP/MC; (c) calculating union probability from these MPV[18–20]/MCV.[9,21]

Network theories have been developed for decades.[22,23] They are now widely applied to solve many real-life problems, such as supply-chain management,[24,25] project management,[26] information systems,[27,28] human resource management,[29] production system planning,[30] telecommunication system planning,[31,32] multi-commodities applications,[33–37] logistics management,[25,38] assignment problems,[39] etc. In tradition, the factors in network like flow, transportation time, transportation cost are deterministic and treated separately, such as the maximal flow problem, the shortest path problem, the least cost path problem, the largest capacity path problem, and the shortest delay path problem. Chen and Chin[40] firstly introduced the multi-factors problem — the quickest path problem. After them, lots of similar problems were investigated, such as the constrained quickest path problem,[41,42] the k quickest paths problem,[43,44] the all-pairs quickest path problem,[45] etc. Although they involved time factor, they were deterministic and did not concern reliability.

Traditionally, searching for MPV in MSN involves three steps[16]: (a) enumerate all feasible flow vectors; (b) transform these vectors to corresponding state vectors; (c) filter out MPV from these state vectors. Steps (a) and (c) are bottlenecks. Explicit enumeration is usually engaged in solving Step (a), and pairwise comparison is usually employed in solving Step (c).[46]

A new approach namely merge search (MS) is proposed to search for MPV in MSN. Also, a new advance in solving integer programming problems namely fast enumeration (FE)[47] is integrated in this approach. Such an integrated approach can greatly improve the time efficiency of searching for MPV in MSN. The proposed approach uses FE to solve Step (a), and MS to solve Step (c) instead. MS takes the ideas from merge sort, which was originated from John Von Neumann in 1945.[48] The time complexity for merge sort in all cases is $O(\pi \log(\pi))$ (π is the number of elements to sort). So, merge sort is a very stable and efficient algorithm for sort. However, it cannot be directly applied to vectors.

The remainder of the work is described as follows. The mathematic preliminaries for the approach are presented in Sec. 2. The proposed algorithm is given in Sec. 3. Some benchmarks are examined and compared in Sec. 4. Finally, Sec. 5 draws the conclusion of this paper.

2. Preliminaries

Let (A, N, M) be a MSN, where $A = \{a_i | 1 \leq i \leq n\}$ is the set of arcs, N is the set of nodes, $M = (m_1, m_2, \ldots, m_n)$ is a vector with m_i (an integer) being the maximal capacity of a_i. The network is assumed to satisfy the following assumptions.

(1) The nodes are perfect.
(2) The states in arcs are statistically independent from each other.
(3) Flow in the network satisfies the flow-conservation law.[22]
(4) The capacity of a_i is an integer-valued random variable which takes values from the set $\{0, 1, 2, \ldots, m_i\}$ according to a given distribution μ_i.

2.1. *The MSN model*

Suppose ρ_1, $\rho_2, \ldots, \rho_z$ are totally the MP from source node to sink node. Thus, the network model can be described in terms of two vectors: the system state vector $X = (x_1, x_2, \ldots, x_n)$ and the flow vector $F = (f_1, f_2, \ldots, f_z)$, where x_i denotes the current state of a_i and f_j denotes the current flow on ρ_j. Then, F is feasible iff

$$\sum_{j=1}^{z} \{f_j | a_i \in \rho_j\} \leq m_i, \quad \text{for } i = 1, 2, \ldots, n. \tag{1}$$

This constraint describes that the total flow through a_i cannot exceed the maximal capacity of a_i. We denote such set of F as $\kappa_M \equiv \{F | F \text{ is feasible under } M\}$, and Ψ_F as the set of formulas in Eq. (1).

Similarly, F is feasible under $X = (x_1, x_2, \ldots, x_n)$ iff

$$\sum_{j=1}^{z} \{f_j | a_i \in \rho_j\} \leq x_i, \quad \text{for } i = 1, 2, \ldots, n. \tag{2}$$

For clarity, let $\kappa_X = \{F | F \text{ is feasible under } X\}$. The maximal flow under X is defined as $\varpi_X \equiv \max\{\sum_{j=1}^{z} f_j | F \in \kappa_X\}$.

At first, we find the flow vector $F \in \kappa_M$ such that the total flow of F equals d. It is defined in the following demand constraint,

$$\sum_{j=1}^{z} f_j = d. \tag{3}$$

Then, let $\mathbf{F}$=$\{F | F \in \kappa_M$ and satisfies Eq. (3)$\}$. If X is a MPV for d, then there is an $F \in \mathbf{F}$ such that

$$x_i = \sum_{j=1}^{z} \{f_j | a_i \in \rho_j\}, \quad \text{for each } i = 1, 2, \ldots, n. \tag{4}$$

This is a necessary condition for a MPV. To explain that, consider a MPV X and an $F \in \kappa_X$ feasible under $\mathbf{F}$. It is known that $\sum_{j=1}^{z}\{f_j|a_i \in \rho_j\} \leq x_i,\ \forall i$. Suppose there is a k such that $x_k > \sum_{j=1}^{z}\{f_j|a_k \in \rho_j\}$. Set $Y = (y_1,\ y_2, \ldots, y_{k-1},\ y_k,\ y_{k+1}, \ldots, y_n) = (x_1,\ x_2, \ldots, x_{k-1},\ x_k - 1,\ x_{k+1}, \ldots, x_n)$. Hence $Y < X$ and $F \in \kappa_Y$ (since $\sum_{j=1}^{z}\{f_j|a_i \in \rho_j\} \leq y_i,\ \forall i$), which indicates that $\varpi_Y \geq d$ and contradicts to that X is a MPV for d. Thus $x_i = \sum_{j=1}^{z}\{f_j|a_i \in \rho_j\},\ \forall i$.

Given $F \in \mathbf{F}$, we generate a capacity vector $X_F = (x_1, x_2, \ldots, x_n)$ via Eq. (4). The set $\Lambda = \{X_F|F \in \mathbf{F}\}$ is built. Let $\Lambda_{\min} = \{X|X$ is a minimal vector in $\Lambda\}$. Then, $\Lambda_{\min}$ is the set of MPV for d.

2.2. *The reliability evaluation*

Given a demand d, the reliability denoted by τ_d is the probability at the sink node that the maximal flow of the network is no less than d, i.e., $\tau_d \equiv \Pr\{X|\varpi_X \geq d\}$. To calculate τ_d, it is advantageous to find the minimal vector in the set $\{X|\varpi_X \geq d\}$. A minimal vector X is said to be a MPV for d iff (i) $\varpi_X \geq d$ and (ii) $\varpi_Y < d$ for any other vector Y such that $Y < X$, in which $Y \leq X$ iff $y_j \leq x_j$ for each $j = 1, 2, \ldots, n$ and $Y < X$ iff $Y \leq X$ and $y_j < x_j$ for at least one j. Suppose there are totally q MPV for d: $X_1, X_2, \ldots, X_q$. The reliability is equal to

$$\tau_d = \Pr\left\{\bigcup_{k=1}^{q}\{X|X \geq X_k\}\right\}, \tag{5}$$

which can be calculated by inclusion-exclusion principle or RRIEP method.[20]

3. The Integrated Approach

3.1. *Fast enumeration*

Assume there are α decision variables and β constraints in a common formulation of an integer programming problem like follows:

$$\text{Maximize} \quad g = f(x_1, x_2, \ldots, x_\alpha) \tag{6}$$

$$\begin{aligned} \text{Subject to} \quad & c_1(x_1, x_2, \ldots, x_\alpha) \\ & c_2(x_1, x_2, \ldots, x_\alpha) \\ & \vdots \\ & c_\beta(x_1, x_2, \ldots, x_\alpha) \\ & x_j \in \mu_j,\ x_j \in \mathbb{Z}, \end{aligned} \tag{7}$$

where $f(\bullet)$ is the objective function, and $c_i(\bullet)$ is the ith constraint. Let e_{ij} be the coefficient of x_j in Constraint c_i and $\delta_i = \{x_j|e_{ij} \neq 0,\ 1 \leq j \leq \alpha\}$.

Definition 1. $\eta_i = ||\delta_i||$.

We define an ordered set $\Omega = \{\eta_{(i)} | 1 \leq i \leq \beta\}$ such that

$$\eta_{(1)} \leq \eta_{(2)} \leq \cdots \leq \eta_{(\beta)}.$$

Following this order, we get the order of δ_i as:

$$\delta_{(1)} \preceq \delta_{(2)} \preceq \cdots \preceq \delta_{(\beta)}.$$

Then, we group the constraint c_i by the rule $\xi_k = \{c_i | \delta_i \cap \delta_{(l)} \neq \phi,\, 1 \leq l \leq k\}$ into r groups.

Let q_k be the total number of enumerations in ξ_k group. The complexity of re-arrangement is now by $O(\prod_{k=1}^{r} q_k)$. The efficiency of FE is greatly improved because $q_k \ll \prod_{j=k-1}^{k} \mu_j$, where μ_j is the value range of x_j.

3.2. *MS for vectors*

At first, we explain how merge sort works on quantities. Merge sort is an efficient, general-purpose, comparison-based algorithm for sorting integers, real numbers, or strings, which was originated from John Von Neumann in 1945.[48] Merge sort works conceptually by merging two sorted sublists into one ascending ordered list, where each sublist is sorted recursively by merge sort itself. The algorithm of merge sort is conceptually described as follows.

Algorithm 1: Merge sort for quantities.

Step 1. Input L. // L is a list of nonordered elements.
Step 2. Divide L into two sublists L_0 and L_1 with almost equal lengths.
Step 3. Call merge sort for L_0, and get L_{sorted_0}.
Step 4. Call merge sort for L_1, and get L_{sorted_1}.
Step 5. Merge L_{sorted_0} and L_{sorted_1} into one ascending ordered list L_{sorted}.
Step 6. Return L_{sorted}.

Since merge sort divides the list into two sublists and merge them, it can be shown that the time complexity is $O(\pi\log(\pi))$ in all cases.[48] π is the length of the list. The space complexity is $O(\pi)$ in all cases. Therefore, merge sort is very efficient for both time and memory usage.

MS takes the idea of merge sort, but do not sort things. It search for MPV from two sublists of MPV candidates, where each sublist is searched recursively by MS itself. Then, it can be shown that the time complexity is also $O(\pi\log(\pi))$ in all cases. The algorithm is described as follows.

Algorithm 2: MS for vectors.

Step 1. Input Λ. // Λ is the list of MPV vectors.
Step 2. If $\Lambda = \phi$, then return ϕ. // empty list.
Step 3. If $||\Lambda|| = 1$, then return Λ. // Only one element in the list.

Step 4. Divide Λ into two sublists Λ_0 and Λ_1 with almost equal lengths.
Step 5. Call Algorithm 2 on Λ_0, and get $\Lambda_{\min_0}$.
Step 6. Call Algorithm 2 on Λ_1, and get $\Lambda_{\min_1}$.
Step 7. Call Algorithm 3 on $\Lambda_{\min_0}$ and $\Lambda_{\min_1}$, and return $\Lambda_{\min}$.

Algorithm 3: Merging.

Step 1. Input $\Lambda_{\min_0}$ and $\Lambda_{\min_1}$. // $\Lambda_{\min_0}$ and $\Lambda_{\min_1}$ are the set of vectors.
Step 2. If $\Lambda_{\min_0} = \phi$, then return $\Lambda_{\min_1}$.
Step 3. If $\Lambda_{\min_0} \neq \phi$ and $\Lambda_{\min_1} = \phi$, then return $\Lambda_{\min_0}$.
Step 4. Let $x_i \in \Lambda_{\min_0}$ and $x_j \in \Lambda_{\min_1}$.
Step 5. If $x_i \leq x_j$, then call Algorithm 3 on $\Lambda_{\min_0}$, $\Lambda_{\min_1} - \{x_j\}$ and return $\Lambda_{\min}$.
Step 6. Else if $x_j \leq x_i$, then call Algorithm 3 on $\Lambda_{\min_0} - \{x_i\}$, $\Lambda_{\min_1}$ and return $\Lambda_{\min}$.
Step 7. Else if $||\Lambda_{\min_1}|| = 1$, then call Algorithm 3 on $\Lambda_{\min_0} - \{x_i\}$, $\Lambda_{\min_1}$ and get $\Lambda_{\min}$; return $\{x_i\} \cup \Lambda_{\min}$.
Step 8. Else if $||\Lambda_{\min_0}|| = 1$, then call Algorithm 3 on $\Lambda_{\min_0}$, $\Lambda_{\min_1} - \{x_j\}$ and get $\Lambda_{\min}$; return $\{x_j\} \cup \Lambda_{\min}$.
Step 9. Else call Algorithm 3 on $\Lambda_{\min_0}$, $\Lambda_{\min_1} - \{x_j\}$ and get Λ_{tmp}, and call Algorithm 3 on Λ_{tmp} and $\{x_j\}$ and return $\Lambda_{\min}$. // Λ_{tmp} is temporary list.

Steps 2 and 3 of Algorithm 3 are boundary conditions. Steps 5 to 9 are the searching cases for MPV.

3.3. *The approach*

We recalled that $\mathbf{F}$ can be enumerated by fitting the formulas in Ψ_F and Eq. (3). Then, X_F can be built by $F \in \mathbf{F}$ via Eq. (4). Next, $\Lambda = \{X_F | F \in \mathbf{F}\}$ is constructed for all X_F. Finally, $\Lambda_{\min} = \{X | X$ is a minimal vector in $\Lambda\}$ is built. Then, $\Lambda_{\min}$ is the set of MPV for d. Let Ξ_{Ψ_F} be FE-form of formulas in Ψ_F, $s_j = \min\{m_i | a_i \in \rho_j\}$, and $J = \{j | X_j \notin \Lambda_{\min}\}$.

The integrated algorithm to search for MPV is as follows.

Algorithm 4: Searching for MPV for d**.**

Step 1. Find the set of feasible system state vectors Λ for d.

(a) fast enumerate $0 \leq f_j \leq u_j$ with Ξ_{Ψ_F} do
(b) if f_j satisfies $\sum_{j=1}^{z} f_j = d$, then
(c) $x_i = \sum_{j=1}^{z} \{f_j | a_i \in \rho_j\}$ for each $i = 1, 2, \ldots, s$, and
(d) $\Lambda = \Lambda \cup \{X_F\}$.
end if.
end fast enumerate.

Step 2. Call Algorithm 2 on Λ, and return $\Lambda_{\min}$. // $\Lambda_{\min}$ is the set of MPV.

Step 1(a) of Algorithm 4 only requires $O(\prod_{k=1}^{r} h_k)$ to generate all feasible F and X_F, since FE-form is by re-arranging the original Ψ_F into r groups of alternative orders, and h_k is the total number of enumerations in the kth group. The time complexity of Algorithm 2 is $O(\pi\log(\pi))$, where $\pi = ||\Lambda||$. The total time complexity of the proposed method is thus $\max\{O(\prod_{k=1}^{r} h_k), O(\pi\log(\pi))\}$, which is greatly smaller than $O(d^z)$, which is the time complexity of traditional ways.

4. Benchmarking

Five rectangular grids of different sizes are employed to do the benchmarking comparisons with Lin's approach,[16] Chen and Lin's approach.[46] The demand is 3 for all cases. The capacities for all these arcs are set to 3. Figure 1 presents these rectangular grids.

All approaches were coded by PROLOG running on a PC with CPU of Intel Core i7 4790 @ 3.60 GHz, 3.60 GHz and 16 GB RAM. Table 1 gives the comparisons

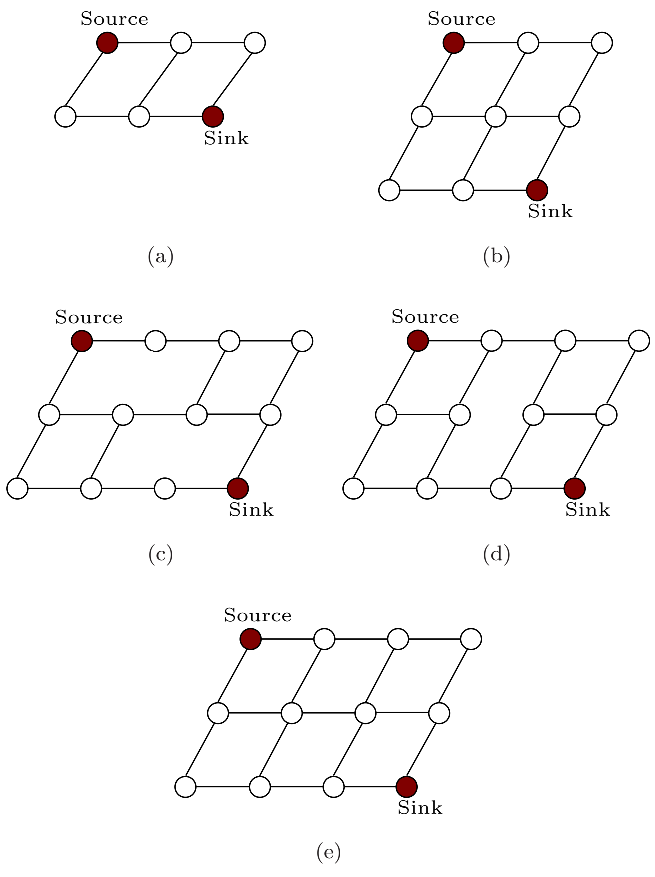

Fig. 1. Five rectangular grids.

Table 1. The comparisons between well known approaches and the proposed approach.

Figure 1(#) (Nodes × Arcs)	# of MP	π	# of MPV	CPU Time (s)		
				Lin[21]	Chen and Lin[46]	The proposed
Figure 1(a) (6 × 7)	4	20	16	0.016/0.002 (0.018)[a]	0.00001/0.002 (0.00201)	0.00001/0.00001 (0.00002)
Figure 1(b) (9 × 12)	12	364	150	6.100/0.374 (6.474)	0.031/0.374 (0.405)	0.031/0.031 (0.062)
Figure 1(c) (12 × 15)	12	364	150	5.944/0.39 (6.334)	0.031/0.39 (0.421)	0.031/0.047 (0.078)
Figure 1(d) (12 × 16)	24	2600	606	NA[b]/35.540 (NA)	2.618/35.540 (38.158)	2.618/0.671 (3.289)
Figure 1(e) (12 × 17)	38	9880	1483	NA/1124.039 (NA)	323.935/1124.039 (1447.974)	323.935/4.976 (328.911)

Notes: [a] Step 1+2/Step 3 (total). [b] Not completed within one day.

between these approaches. The network in Fig. 1(a) has 4 MP found, and can be derived 16 MPV. The networks in Figs. 1(b) and 1(c) both have 12 MP found, and can be derived 150 MPV. The network in Fig. 1(d) has 24 MP found, and can be derived 606 MPV. The network in Fig. 1(e) has 38 MP found, and can be derived 1483 MPV. The CPU times of the proposed approach are significantly shorter than the other approaches in this comparisons.

5. Conclusion

This paper proposes a novel approach to search for MPV in MSN. This approach involves two main techniques: one is FE,[47] which was a recent advance in the Integer programming problem-solving; the other is MS for vectors, which is proposed in this paper. FE can speed up the time-consuming explicit enumeration algorithm (EEA) by a way without changing their formulas, just re-arranging their order of formulas in the algorithm. MS is similar to merge sort proposed by John Von Neumann in 1945.[48] The benchmarking comparisons are conducted with the well-known approaches and the proposed approach. The results show that the proposed approach can significantly improve the efficiency of searching for MPV in MSN.

Future researches are encouraged to inspect multi-terminal MSN, or multi-commodities networks to solve more real-life problems.

Acknowledgments

This work was supported in part by the Ministry of Science and Technology, Taiwan, Republic of China, under Grant No. MOST 105-2221-E-236-004.

References

1. K. K. Aggarwal, J. S. Gupta and K. B. Misra, A simple method for reliability evaluation of a communication system, *IEEE Trans. Commun.* **23** (1975) 563–565.
2. S. H. Lee, Reliability evaluation of a flow network, *IEEE Trans. Rel.* **29** (1980) 24–26.
3. K. K. Aggarwal, Y. C. Chopra and J. S. Bajwa, Capacity consideration in reliability analysis of communication systems, *IEEE Trans. Rel.* **31** (1982) 177–180.
4. W. J. Rueger, Reliability analysis of networks with capacity-constraints and failures at branches and nodes, *IEEE Trans. Rel.* **35** (1986) 523–528.
5. E. El-Neweihi, F. Proschan and J. Sethuraman, Multistate coherent systems, *J. Appl. Probab.* **15**(4) (1978) 675–688.
6. J. Xue, On multistate system analysis, *IEEE Trans. Rel.* **34**(4) (1985) 329–337.
7. J. S. Lin, C. C. Jane and J. Yuan, On reliability evaluation of a capacitated-flow network in terms of minimal pathsets, *Networks* **25** (1995) 131–138.
8. C. C. Jane, J. S. Lin and J. Yuan, On reliability evaluation of a limited-flow network in terms of minimal cutsets, *IEEE Trans. Rel.* **42** (1993) 354–361, 368.
9. C. J. Colbourn, *The Combinatorics of Network Reliability* (Oxford University Press, UK, 1987).
10. Y. Shen, A new simple algorithm for enumerating all minimal paths and cuts of a graph, *Microelectron. Reliab.* **35** (1995) 973–976.

11. S. G. Chen and Y. K. Lin, Search for all minimal paths in a general large flow network, *IEEE Trans. Rel.* **61**(4) (2012) 949–956.
12. U. Abel and R. Bicker, Determination of all minimal cut-sets between a vertex pair in an undirected graph, *IEEE Trans. Rel.* **R-31** (1982) 167–171.
13. G. B. Jasmon and K. W. Foong, A method for evaluating all the minimal cuts of a graph, *IEEE Trans. Rel.* **R-36** (1987) 538–545.
14. W. C. Yeh, A simple algorithm to search for all MCs in networks, *Eur. J. Oper. Res.* **174** (2006) 1694–1705.
15. S. G. Chen, Search for all MCs with backtracking, *Int. J. Reliab. Qual. Perf.* **6**(2) (2015) 101–106.
16. Y. K. Lin, A simple algorithm for reliability evaluation of a stochastic-flow network with node failure, *Comput. Oper. Res.* **28**(13) (2001) 1277–1285.
17. W. C. Yeh, A simple approach to search for all *d*-MCs of a limited-flow network, *Reliab. Eng. Syst. Safe.* **71** (2001) 15–19.
18. M. J. Zuo, Z. Tian and H.-Z. Huang, An efficient method for reliability evaluation of multistate networks given all minimal path vectors, *IIE Trans.* **39** (2007) 811–817.
19. S. G. Chen, Reduced recursive sum of disjoint product in network reliability, in *Proc. 20th ISSAT Int. Conf. Reliability and Quality in Design*, Seattle, Washington, USA (2014), pp. 170–173.
20. S. G. Chen, Reduced recursive inclusion-exclusion principle for the probability of union events, *2014 IEEE Int. Conf. Industrial Engineering and Engineering Management*, Selangor, Malaysia (2014), pp. 1–3.
21. Y. K. Lin, On reliability evaluation of a stochastic-flow network in terms of minimal cuts, *J. Chin. Ins. Ind. Eng.* **18** (2001) 49–54.
22. L. R. Ford and D. R. Fulkerson, *Flows in Networks* (Princeton University Press, NJ, 1962).
23. L. R. Ford and D. R. Fulkerson, A suggested computation for maximal multicommodity network flows, *Manage. Sci.* **20** (1974) 822–844.
24. S. G. Chen, Optimal logistic planning in robust supply chain management, *The 40th Int. Conf. Computers & Industrial Engineering*, Awaji Island, Japan (2010), pp. 1–6.
25. S. G. Chen, Fuzzy-scorecard based logistics management in robust SCM, *Comput. Ind. Eng.* **62**(3) (2012) 740–745.
26. Y. K. Lin, Study on the multicommodity reliability of a capacitated-flow network, *Comput. Math. Appl.* **42** (2001) 255–264.
27. S. G. Chen and Y. K. Lin, An evaluation method for enterprise resource planning systems, *J. Oper. Res. Soc. Jpn.* **51**(4) (2008) 299–309.
28. S. G. Chen and Y. K. Lin, On performance evaluation of ERP systems with fuzzy mathematics, *Expert Syst. Appl.* **36**(3) (2009) 6362–6367.
29. S. G. Chen, Optimal staffing for a SOA company, *Int. J. Oper. Res.* **7**(3) (2010) 15–22.
30. S. G. Chen, Optimal device planning and performance evaluation in AMS, in *Proc. APARM 2010*, Wellington, New Zealand (2010), pp. 113–120.
31. Y. K. Lin, Reliability evaluation for an information network with node failure under cost constraint, *IEEE Trans. Syst. Man Cybern. A*: *Syst. Hum.* **37**(2) (2007) 180–188.
32. S. G. Chen, Optimal network planning in robust two site's communication, in *Proc. 2010 Int. Symp. Information Theory and Its Applications*, Taichung, Taiwan (2010), pp. 337–342.
33. T. C. Hu, Multi-commodity network flows, *Oper. Res.* **11** (1963) 344–360.
34. A. A. Assad, Multicommodity network flows- A survey, *Networks* **8** (1978) 37–91.
35. Y. K. Lin, Reliability evaluation for a multi-commodity flow model under budget constraint, *J. Chin. Inst. Eng.* **25** (2002) 109–116.

36. Y. K. Lin, On a multicommodity stochastic-flow network with unreliable nodes subject to budget constraint, *Eur. J. Oper. Res.* **176**(1) (2007) 347–360.
37. Y. K. Lin, System capacity for a two-commodity multistate flow network with unreliable nodes and capacity weight, *Comput. Oper. Res.* **34**(10) (2007) 3043–3054.
38. Y. K. Lin, Study on longer and shorter boundary duration vectors with arbitrary duration and cost values, *J. Oper. Res. Soc. Jpn.* **50**(2) (2007) 73–81.
39. S. G. Chen, Optimal double-resource assignment for the robust design problem in multistate computer networks, *Appl. Math. Model.* **38**(1) (2014) 263–277.
40. Y. L. Chen and Y. H. Chin, The quickest path problem, *Comput. Oper. Res.* **17** (1990) 153–161.
41. G. H. Chen and Y. C. Hung, Algorithms for the constrained quickest path problem and the enumeration of quickest paths, *Comput. Oper. Res.* **21** (1994) 113–118.
42. J. B. Rosen, S. Z. Sun and G. L. Xue, Algorithms for the quickest path problem and the enumeration of quickest paths, *Comput. Oper. Res.* **18** (1991) 579–584.
43. Y. L. Chen, An algorithm for finding the k quickest paths in a network, *Comput. Oper. Res.* **20** (1993) 59–65.
44. Y. L. Chen, Finding the k quickest simples paths in a network, *Inform. Process. Lett.* **50** (1994) 89–92.
45. D. T. Lee and E. Papadopoulou, The all-pairs quickest path problem, *Inform. Process. Lett.* **45** (1993) 261–267.
46. S. G. Chen and Y. K. Lin, Searching for d-MPs with fast enumeration, *J. Comput. Sci.*, in press (2016), http://dx.doi.org/10.1016/j.jocs.2016.05.011.
47. S. G. Chen, Efficiency improvement in explicit enumeration for integer programming problems, *2013 IEEE Int. Conf. Industrial Engineering and Engineering Management*, Bangkok, Thailand (2013), pp. 98–100.
48. D. Knuth, *The Art of Computer Programming 3*, 2nd edn. (Addison-Wesley, MA, 1998).

About the Authors

Yi-Kuei Lin is currently a Chair Professor of Industrial Management Department, National Taiwan University of Science and Technology, Taiwan, Republic of China. He received a Bachelor degree in Applied Mathematics Department from National Chiao Tung University, Taiwan. He obtained his Master degree and Ph.D. degree in the Department of Industrial Engineering and Engineering Management at National Tsing Hua University, Taiwan, Republic of China. He has the honor to get the Outstanding Research Awards from Ministry of Science and Technology of Taiwan in 2008, 2010, and 2013, respectively. His research interest includes performance evaluation, stochastic network reliability, operations research, and telecommunication management. He has published over 140 papers in refereed journals including European Journal of Operational Research; Computers and Operations Research; Reliability Engineering & System Safety; IEEE Transactions on Reliability; IEEE Transactions on Systems, Man, and Cybernetics–Part A: Systems and Humans; International Journal of Industrial Engineering; Computers and Mathematics with Applications; International Journal of Advanced Manufacturing Technology; International Journal of Reliability, Quality and Safety Engineering; Journal of the Operations Research Society of Japan; Mathematical and Computer

Modelling; Computers and Industrial Engineering; International Journal of Production Economics; and Applied Mathematics and Computation.

Shin-Guang Chen received his B.Sc. in Computer Engineering from NCTU in Taiwan and his Ph.D. in IE from NCTU in Taiwan. He has been with the Department of Industrial Management at Tungnan University in Taiwan, where he holds the rank of Professor. His research interests are in management, reliability, ERPS and sustainable energy technologies. His recent work involves problems related to the analysis of flow networks. He has published papers in IJPR, IJIE, ESWA, JORSJ, CSTM, IJOR, JCIIE, IJRQP, APEN, IJPE, Omega, IEEE T Reliab, APM, JIMO, APJOR, AOR, etc. He is a member of IEEE, IEEE Reliab, POMS, CIIE, CERPS, and CSQ.