Counting the Number of Minimum Cuts in Undirected Multigraphs

Hiroshi Nagamochi
Kyoto University, Kyoto
Zheng Sun
Toyohashi University of Technology, Toyohashi
Toshihide Ibaraki, Member IEEE
Kyoto University, Kyoto

Key Words — **Undirected graph, multigraph, edge-connectivity, minimum cut, polynomial-time algorithm, spanning forest, spanning subgraph, edge contraction, super-λ.**

Reader Aids —
Purpose: Present an algorithm and widen the state of the art
Special math needed for explanations: Graph theory and theory of algorithms
Special math needed to use results: Same
Results useful to: Reliability analysts and theoreticians

Abstract — **The problem of counting the number of cuts with the minimum cardinality in an undirected multigraph arises in various applications such as testing the super-λ-ness of a graph and calculating upper and lower bounds on the probabilistic connectedness of a stochastic graph G in which edges are subject to failure. This paper shows that the number $|C(G)|$ of cuts with the minimum cardinality $\lambda(G)$ in a multiple graph $G = (V, E)$ can be computed in $O(|E| + \lambda(G)|V|^2 + \lambda(G)|C(G)||V|)$ time.**

1. INTRODUCTION

The problem of counting the number of cuts with the minimum cardinality in an undirected multigraph arises in various aspects of network reliability, testing the super-λ-ness of a graph [3], estimating the probabilistic connectedness of a stochastic graph in which edges are subject to failure with probability p [1, 8-10], and other areas [7]. For example, the probabilistic connectedness of G can be expressed by the polynomial:

$$P(G, p) = 1 - \sum_{i=\lambda(G)}^{|E|} m_i(G)p^i(1-p)^{|E|-i}. \qquad (1\text{-}1)$$

Notation

$\lambda(G)$ cardinality of minimum cuts in G
$\lambda(s,t;G)$ cardinality of minimum s-t cuts (cuts with the minimum cardinality that separate two nodes s and t) in G
$|C(G)|$ number of minimum cuts in G
$|C(s,t;G)|$ number of minimum s-t cuts in G
$m_i(G)$ number of cuts with cardinality i [1, 8].

For a sufficiently small p, (1-1) is approximated by:

$$P(G,p) \approx 1 - m_{\lambda(G)}(G)p^{\lambda(G)}(1-p)^{|E|-\lambda(G)}, \qquad (1\text{-}2)$$

suggesting the importance of counting the number of minimum cuts, $m_{\lambda(G)}(G)$.

Ball & Provan [1] developed an $O(T+|C(s,t;G)||E|)$ time algorithm for counting the number of minimum s-t cuts in $G=(V,E)$, where T is the time required to find the maximum number of edge disjoint paths between s and t (T is bounded from above by $O(|V||E|)$ by the algorithm of [4]). They also presented an $O(|V|T+|C(G)||E|)$ time algorithm for computing the number $|C(G)|$ of minimum cuts in G. This bound can be simplified to $O(|V|^2|E|)$ by $T=O(|V||E|)$ and $|C(G)|=O(|V|^2)$ [2, theorem E].

In this paper, we propose an

$$O(|E|+(|C(s,t;G)|+\min\{|V|,\ \lambda(s,t;G),\ |E|^{1/2}\})\ \min\{|E|,\ \lambda(s,t;G)|V|\})$$

time algorithm for computing the number of minimum s-t cuts in a multigraph G. Based on this, we present an $O(|E|+\lambda(G)|V|^2+\lambda(G)|C(G)||V|)$ time algorithm for computing the number of minimum cuts. This bound is never worse than the bound $O(|V|T+|C(G)||E|)$ of [1], because $O(\lambda(G)|V|^2) \le O(T|V|)$ and $O(\lambda(G)|C(G)||V|) \le O(|C(G)||E|)$ always hold by an obvious relation $O(\lambda(G)|V|) \le O(|E|) \le O(T)$; recall that $\lambda(G) \le$ minimum degree $\le 2|E|/|V|$. Furthermore, the new bound is better if $O(\lambda(G)|V|)$ is smaller than $O(|E|)$ since the bound $O(|V|T+|C(G)||E|)$ of [1] is irrelevant to $\lambda(G)$; the time complexity of algorithm of [1] depends on $\lambda(G)$ only in minor terms.

2. PRELIMINARIES

Throughout the paper, a graph $G=(V,E)$ stands for a connected undirected multigraph unless otherwise specified. It is assume that G has no self-loop. $O(|E|)$ might not be bounded from above by $O(|V|^2)$ since G has multiple edges. Unless confusion arises, however, an edge e with end notes u, v is denoted by $e=(u,v)$. A graph $G'=(V',E')$ is called a subgraph of $G=(V,E)$ if $V' \subseteq V$ and $E' \subseteq E$. It is a spanning subgraph if $V'=V$. A graph G without cycle is called a forest. For a given graph G (possibly not connected), a spanning forest G' of G is maximal if the subgraph of G' induced by each connected component of G is connected.

Notation

$G/\{x,y\}$ graph obtained from G by contracting nodes x and y and in which self-loops (if edges $e=(x,y)$ exist) resulting from contraction are deleted.

0018-9529/91/1200-0610$01.00©1991 IEEE

$G-F$ graph obtained from a connected graph $G=(V,E)$ by removing a subset F of E

$C(x,y;G)$ set of minimum x-y cuts of G: $C(x,y;G) \equiv \{F \subseteq E |\ x$ and y are disconnected in G-F, $|F| = \lambda(x,y;G)\}$

$C(G)$ set of minimum cuts of G: $C(G) \equiv \{F \subseteq E | F$ is a cut in G, $|F| = \lambda(G)\}$.

F is called a cut if G-F is disconnected. G is called k-edge-connected if G-F is connected for any $F \subseteq E$ with $|F| \leq k-1$. The edge-connectivity $\lambda(G)$ of a graph G is defined to be k if G is k-edge-connected, but not $(k+1)$-edge-connected. The local edge-connectivity $\lambda(x,y;G)$ for $x,y \in V$ is the smallest cardinality $|F|$ of an x-y cut $F \subseteq E$ such that x and y are disconnected in G-F. Clearly,

$$\lambda(G) = \min\{\lambda(x,y;G) | x,y \in V\}. \quad (2\text{-}1)$$

3. k-EDGE-CONNECTED SPANNING SUBGRAPH

The following properties are bases for our algorithm.

Lemma 3.1 Let —

- $G=(V,E)$ be a multigraph
- $E_1,E_2,\ldots,E_{|E|}$ be a sequence of subsets of E such the (V,E_i) is a maximal spanning forest in G-$E_1 \cup E_2 \cup \ldots \cup E_{i-1}$, for $i=1,2,\ldots,|E|$, where possibly $E_i=\phi$ for some i. (Therefore, $E_1,E_2,\ldots,E_{|E|}$ provide a partition of E.)
- $G_i=(V,E_1 \cup E_2 \cup \ldots \cup E_i)$.

Then —

i. $|E_i| \leq |V|-1$, $i=1,2,\ldots,|E|$.
ii. $E_i \neq \phi$ for $i=1,2,\ldots,\lambda(G)$.
iii. For each $i=1,2,\ldots,|E|$, any edge $e=(u,v)$ in E_i (if $E_i \neq \phi$) has i-1 edge-disjoint paths $P_1 \subseteq E_1$, $P_2 \subseteq E_2,\ldots,P_{i-1} \subseteq E_{i-1}$, each of which is connecting u and v (i.e., $\lambda(u,v;G) \geq i$).
iv. $\lambda(x,y;G_i) \geq \min\{\lambda(x,y;G),i\}$ for $x,y, \in V$ and $i=1,2,\ldots,|E|$.
v. $C(x,y;G_i) = C(x,y;G)$ for any $x,y, \in V$ and i such that $\lambda(x,y;G) < i \leq |E|$.

It is known that a partition E_i $(i=1,2,\ldots,|E|)$ of E as stated in lemma 3.1 can be obtained in $O(|V|+|E|)$ time by algorithm FOREST in [5].

4. COUNTING MINIMUM s-t CUTS

Lemma 4.1 is due to Ball & Provan [1].

Lemma 4.1 [1: proposition 1, theorem 6, propositions 2 & 3] Let —

- $G=(V,E)$ be a multiple graph
- s and t be two specified nodes.

Then the number of minimum s-t cuts $|C(s,t;G)|$ can be counted in $O(T+|C(s,t;G)||E|)$ time, where T denotes the time required to find a set of $\lambda(s,t;G)$ edge-disjoint paths between s and t. □

The Ball & Provan algorithm ACYCGEN for counting the number of minimum s-t cuts consists of two main phases.

ACYCGEN(s,t)
Phase 1: Find a set of $\lambda(s,t;G)$ edge-disjoint paths between s and t.
Phase 2: Based on the obtained set, construct the acyclic directed graph $G'=(V',A')$ defind in [1] (such G' can be found in $O(|E|)$ time). There is a one-to-one correspondence between minimum s-t cuts and anti-chains in G'. Therefore count the number of anti-chains in G' one by one spending $O(|A'|)$ $(\leq O(|E|))$ time each □

An antichain W in G' is a subset of nodes such that there is no directed path from any node $v \in W$ to any node $u(\neq v) \in W$.

It is shown [1: proposition 3] that phase 1 requires $O(T)$ time and phase 2 requires $O(|C(s,t;G)||E|)$ time. If $\lambda(s,t;G)$ edge disjoint paths between s and t are already known (phase 1 is done), then testing whether $|C(s,t;G)| \geq k$ or not for a given constant k requires —

$$O(\min\{k,|C(s,t;G)|\}|E|) \quad (4\text{-}1)$$

time, since phase 2 can be conducted until $\min\{k,|C(s,t;G)|\}$ minimum s-t cuts are generated. This modified algorithm is used in the algorithm of section 5, which tests whether $|C(G)| \geq k$ or not for a given k.

As used in [1], T is bounded from above by $O(|V||E|)$ [4]. But the current best bounds on T are [5]:

$$O(|E| + \min\{|V|,\ \lambda,\ \sqrt{m'}\}m') \text{ if } G \text{ is multiple} \quad (4\text{-}2)$$

$$O(|E| + \min\{\lambda, |V|^{2/3}\}m') \text{ if } G \text{ is simple} \quad (4\text{-}3)$$

$$\lambda \equiv \lambda(x,y;G)$$

$$m' \equiv \min\{|E|,\ \lambda|V|\}.$$

(Ref [5] only discusses algorithms to compute connectivity $\lambda(x,y;G)$ of simple and multigraphs. But it is straightforward to modify them to compute a set of $\lambda(x,y;G)$ edge-disjoint paths.) Based on this observation and lemma 3.1, we derive an improved bound to compute the number $|C(s,t;G)|$ of minimum s-t cuts.

Theorem 4.1 For two nodes s,t in a graph $G=(V,E)$, the number $|C(s,t;G)|$ can be computed in —

$$O(|E| + (|C(s,t;G)| + \min\{|V|,\ \lambda,\ \sqrt{|E|}\})m'), \text{if } G \text{ is multiple,}$$

$$O(|E| + (|C(s,t;G)| + \min\{\lambda, |V|^{2/3}\})m'), \text{ if } G \text{ is simple}$$

$$\lambda \equiv \lambda(s,t;G)$$

$$m' \equiv \min\{|E|,\ \lambda|V|\}.$$

5. COUNTING MINIMUM CUTS OF G

An $O(|V|T+|C(G)||E|)$ time algorithm is known [1] to compute the number $|C(G)|$ of minimum cuts in $G=(V,E)$, where T denotes the time required to find a maximum set of edge-disjoint paths between s and t. This time complexity is bounded from above by $O(|V|^2|E|)$ since $T\le O(|V||E|)$ is known in [4] and $|C(G)|$ is bounded by,

$$|C(G)| \le \begin{cases} O(|V|), \text{ if } \lambda(G) \text{ is odd} \\ O(|V|^2), \text{ if } \lambda(G) \text{ is even,} \end{cases} \tag{5-1}$$

as shown in [2: theorem E]. In this section, we present an $O(|E|+\lambda(G)|V|^2+\lambda(G)|C(G)||V|)$ time algorithm. The new bound is an improvement since $O(\lambda(G)|V|) \le O(|E|) \le O(T)$ always holds.

Here define for a $G=(V,E)$ and an integer $i>0$,
$C^i(x,y;G) \equiv \{F\subseteq E | x$ and y are disconnected in G-F, and $|F|=i\}$, $C^i(G) \equiv \{F \subseteq E | F$ is a cut of G with $|F|=i\}$,
$C^i(x,y;G)=\phi$, if $i<\lambda(x,y;G)$,
$C^i(G)=\phi$, if $i<\lambda(G)$.

Lemma 5.1 For a graph $G=(V,E)$ with $|V|\ge 3$ and $\lambda=\lambda(G)$, let $u,v, \in V$. Then —

i. $\lambda(G/\{u,v\})\ge\lambda$.
ii. $C(G)=C^\lambda(u,v;G) \cup C^\lambda(G/\{u,v\})$ and $|C(G)|=|C^\lambda(u,v;G)|+|C^\lambda(G/\{u,v\})|$. □

The algorithm MINCUT, introduced below, computes $|C(G)|$ as follows. First, it computes $\lambda=\lambda(G)$ and reduces G to $G_{\lambda+1}=(V,E_1 \cup E_2\cup\ldots\cup E_{\lambda+1})$ because $C(G)=C(G_{\lambda+1})$ by lemma 3.1.v. Next, it chooses two nodes $u,v \in V$, checks whether $\lambda(u,v;G_{\lambda+1})=\lambda$ or not, and then reduces $G_{\lambda+1}$ to $\tilde{G}=G_{\lambda+1}/\{u,v\}$. If $\lambda(u,v;G_{\lambda+1})=\lambda$, then —

$$|C(G)|=|C(u,v;G_{\lambda+1})|+|C^\lambda(\tilde{G})|$$

by lemma 5.1.ii; otherwise,

$$|C(G)|=|C^\lambda(\tilde{G})|.$$

In any case, the remaining task is to compute $|C^\lambda(\tilde{G})|$. Therefore, by repeating this process, $|C(G)|$ can be eventually computed.

To reduce the time required to check $\lambda(u,v;G_{\lambda+1})=\lambda$, we employ the next rule of selecting nodes $u,v \in V$. Let $G'=(V',E')$ be a graph encountered in the above iterations. By using FOREST [5], E' is partitioned into $E_i(i=1,2,\ldots,|E'|)$ in $O(|E'|)$ time. Since $\lambda(G')\ge\lambda$ holds by lemma 5.1.i, there is at least one edge $e=(u,v)$ in E_λ by lemma 3.1.ii. Choose these u and v. Then λ edge-disjoint paths connecting them can be found in $O(|E'|)$ time by using lemma 3.1.iii.

Procedure MINCUT; {Input: a connected multigraph $G=(V,E)$.
Output: the number $|C(G)|$ of minimum cuts in G.}
begin
1. Find the edge-connectivity $\lambda(G)$ of G and let $\lambda:=\lambda(G)$;
2. Partition E into $E_1,E_2,\ldots,E_{|E|}$ by applying FOREST to G;
3. $G_{\lambda+1}:=(V,E_1 \cup E_2 \cup\ldots\cup E_\lambda \cup E_{\lambda+1})$; {$E_{\lambda+1}=\phi$ if $\lambda=|E|$}
4. $G':=G_{\lambda+1}$; $\alpha:=0$;
5. **while** $|V'|\ge 3$ in $G'=(V',E')$ **do**
begin
6. Partition E' into $E_1,E_2,\ldots,E_{|E'|}$ by applying FOREST to G';
7. Choose an edge $e=(u,v) \in E_\lambda$;
8. Find set $\{P_j | j=1,2,\ldots,\lambda\}$ of λ edge-disjoint paths in G' between u and v; {($\lambda-1$) paths can be uniquely determined from each E_i, $i=1,2,\ldots,\lambda-1$, in $O(|E'|)$ time by Lemma 3.1.iii, since each graph (V',E_i) is a spanning forest. Add $e=(u,v)$.}
9. **If** $\lambda(u,v;G')=\lambda$ {This is checked efficiently by using λ edge disjoint paths obtained above} **then**
begin
10. Compute $|C(u,v;G')|$; {e.g., by Phase 2 of ACYCGEN(u,v)}
11. $\alpha:=\alpha+|C(u,v;G')|$;
end
12. $G'(V',E'):=G'(V',E')/\{u,v\}$
end; {while}
13. **If** $|E'|=\lambda$ **then** $\alpha:=\alpha+1$; {the case of $|V'|=2$}
14. Conclude that $|C(G)|=\alpha$
end. □

Theorem 5.1 The number of minimum cuts $|C(G)|$ of $G=(V,E)$ can be computed by MINCUT in $O(|E|+\lambda(G)|V|^2+\lambda(G)|C(G)||V|)$ time. □

In order to test whether $|C(G)|\ge k$ or not for a given integer $k>0$, line 10 is modified as follows:

Test if $|C(u,v;G')|\ge k-\alpha$ (α denotes the number of minimum cuts enumerated so far). If yes, halt by concluding that $|C(G)|\ge k$. The required time is:

$$O(|E|+\lambda(G)|V|^2+\min\{k,|C(G)|\}\,\lambda(G)|V|). \tag{5-2}$$

This follows from the fact that testing $|C(u,v;G')|\ge k-\alpha$ in line 10 can be done in $O(\min\{k-\alpha,|C(u,v;G')|\}|E'|)$ time as obtained from (4-1).

The Super-λ-ness of a graph, defined as follows, is one of the important reliability measures of a graph [3]. A graph $G=(V,E)$ with $|V|\ge 3$ is called super-λ if:
$C(G)=\{E(v) | v \in V$ with degree$=\delta(G)\}$

Notation

$\delta(G)$ minimum degree of G
$E(v)$ set of edges incident to node v.

The problem for constructing a super-λ graph with specified numbers of nodes and edges has been extensively studies

[3, 11, 12], while no special algorithm for testing whether a given graph is super-λ has been developed. Eq (5-2) implies corollary 5.1.

Corollary 5.1 Whether a graph $G=(V,E)$ is super-λ or not can be tested in $O(|E|+\lambda(G)|V|^2)$ time.

ACKNOWLEDGMENT

We are pleased to thank Professor Kikunobu Kusunoki, Toyohashi University of Technology for his warm encouragement, and anonymous referees for their helpful comments. This work was partially supported by the Scientific Grant-in-Aid by the Ministry of Education, Science and Culture of Japan; the first author was further supported by the subvention to young scientists by Research Foundation of the Electrotechnology of Chubu.

APPENDIX: PROOFS

Proof of Lemma 3.1

i. Obviously by the definition of forest F_i.
ii. [5: lemma 2.6].
iii. Since the edge $e=(u,v)$ is contained in G-$E_1 \cup E_2 \cup \ldots \cup E_{j-1}$, for every $j=1,2,\ldots,i-1$ but is not chosen for E_j, maximal forest $F_j=(V,E_j)$ must contain a path $P_j \subseteq E_j$ connecting u and v (otherwise e must have been added to F_j).
iv. [5: lemma 2.1].
v. Let $k=\lambda(x,y;G)$. As it is trivial for $i=|E|$, assume $k<i<|E|$. By iv for k and $\lambda(x,y;G_k) \le \lambda(x,y;G_{i-1}) \le \lambda(x,y;G_i) \le \lambda(x,y;G)$ (by definition of λ), we have —

$$\lambda(x,y;G_k) = \lambda(x,y;G_{i-1}) = \lambda(x,y;G_i) = \lambda(x,y;G) = k.$$

Any minimum cut $F \in C(x,y;G)$ with cardinality k satisfies $F \subseteq E_1 \cup \ldots \cup E_i$ (otherwise, $F \cap (E_1 \cap E_2 \cup \ldots \cup E_i)$ is a cut in G_i, ie, $\lambda(x,y;G_i) < k$, a contradiction). Thus $C(x,y;G) \subseteq C(x,y;G_i)$. Now we show the converse. Assume that there exists an $F \in C(x,y;G_i)-C(x,y;G)$, that is, x and y are disconnected in G_i-F but are connected in G-F. By the minimality of $|F|$, G_i-F consists of exactly two connected components X and $V-X$, where $x \in X$ and $y \in V-X$ are assumed without loss of generality. From the connectivity of x and y in G-F, there is an edge $e=(u,v) \in E_{i+1} \cup E_{i+2} \cup \ldots \cup E_{|E|}$ such that $u \in X$ and $v \in V-X$. From iii, there is a path $P_i \subseteq E_i$ connecting u and v in G_i, i.e., $F \cap P_i \neq \phi$. This implies $F-P_i \in C(x,y;G_{i-1})$. However, $|F-P_i|<|F|=k$ contradicts $\lambda(x,y;G_{i-1})=k$. *Q.E.D.*

Proof of theorem 4.1

First, compute $\lambda=\lambda(s,t;G)$ as well as a set of λ edge disjoint paths between s and t, spending time of (4-2) or (4-3). Then reduce G to $G_{\lambda+1}=(V,E'=E_1 \cup E_2 \cup \ldots \cup E_\lambda \cup E_{\lambda+1})$ of lemma 3.1 in $O(|E|)$ time by using algorithm FOREST [5]. By lemma 3.1(i), $|E'| \le (\lambda+1)(|V|-1)$, ie, $|E'|=O(m')$. By applying phase 2 of ACYCGEN(s,t) to the resulting graph $G_{\lambda+1}$ (since the set of λ edge disjoint paths computed for G can also be used for G' [5]), $|C(s,t;G_{\lambda+1})|$ ($=|C(s,t;G)|$ by lemma 3.1.v) can be computed in $O(|C(s,t;G)|m')$ time by lemma 4.1. The total time is:

$$O(|E|+\min\{|V|,\lambda,\sqrt{m'}\}m')+O(|E|)+O(|C(s,t;G)|m')$$

$$= O(|E|+(|C(s,t;G)|+\min\{|V|,\lambda,\sqrt{m'}\})m')$$

$$= O(|E|+(|C(s,t;G)|+\min\{|V|,\lambda,\sqrt{|E|},\sqrt{\lambda|V|}\})m')$$

$= O(|E|+(|C(s,t;G)|+\min\{|V|,\lambda,\sqrt{|E|}\})m')$, if G is multiple,

$$O(|E|+\min\{\lambda,|V|^{2/3}\}m')+O(|E|)+O(|C(s,t;G)|m')$$

$= O(|E|+(|C(s,t;G)|+\min\{\lambda,|V|^{2/3}\})m')$, if G is simple. *Q.E.D.*

Proof of lemma 5.1

i. Immediate from that any cut F in $G/\{u,v\}$ is a cut in G. This also implies $C(G) \supseteq C^\lambda(u,v;G) \cup C^\lambda(G/\{u,v\})$.
ii. Clearly, any $F' \in C(G)-C^\lambda(u,v;G)$ is a cut in $G/\{u,v\}$. That is $C(G) \subseteq C^\lambda(u,v;G) \cup C^\lambda(G/\{u,v\})$. The latter relation is obvious since $C^\lambda(u,v;G)$ and $C^\lambda(G/\{u,v\})$ are disjoint. *Q.E.D.*

Proof of therem 5.1

Since the validity of MINCUT has already been discussed, we derive here the stated time bound. It is known [6] that the edge-connectivity $\lambda(G)$ of a multigraph G in line 1 can be computed in $O(|E|+\lambda(G)|V|^2)$ time. Lines 2-4 requires at most $O(|E|)$ time [5]. The while-loop from line 5 to line 12 iterates at most $|V|-2$ times since the number of nodes in G' decreases by 1 in each iteration. Lines 6-8 requires $O(|E'|) \le O(\lambda(G)|V|)$ time by lemma 3.1.i & iii and [5]. Based on these $\lambda(G)$ edge-disjoint paths $P_j, j=1,2,\ldots,\lambda(G)$, we see that $\lambda(u,v;G')>\lambda(G)$ holds if and only if the auxiliary graph introduced by [4] with respect to 0/1-flow $\{P_j\}$ contains another augmenting path from u to v. Therefore, condition $\lambda(u,v;G')=\lambda(G)$ in line 9 can be tested in $O(|E'|)$ time. Then line 10 requires at most $O(|C(u,v;G')||E'|)=O(|C(u,v;G)||E'|)$ time by lemma 4.1; note that $\lambda(u,v;G')(=\lambda(G))$ edge-disjoint paths between u and v have already been obtained in line 8. Lines 12 and 13 can be done in $O(|E'|)$ time. The total time is:

$$O(|E|+\lambda(G)|V|^2)+O(|E|)+(|V|-2)O(|E'|) + O\left(\left(\sum_{G'} |C(u,v;G')|\right)|E'|\right)$$

$$= O(|E|+\lambda(G)|V|^2+\lambda(G)|C(G)||V|). \qquad Q.E.D.$$

Proof of corollary 5.1

$\lambda(G)$ can be computed in $O(|E|+\lambda(G)|V|^2)$ time [6]. The property $C(G)=\{E(v)\,|\,v \in V \text{ with degree}=\delta(G)\}$ can be tested by checking whether $|C(G)|>k$ holds or not, where

$k = |\{E(v) \mid v \in V \text{ with degree} = \delta(G)\}|$. By (5-2), the running time of this part is:

$$O(|E| + \lambda(G)|V|^2 + \min\{k, |C(G)|\} \lambda(G)|V|)$$

$$= O(|E| + \lambda(G)|V|^2). \qquad Q.E.D.$$

REFERENCES

[1] M. O. Ball, J. S. Provan, "Calculating bounds on reachability and connectedness in stochastic networks", *Networks*, vol 13, 1983, pp 253-278.
[2] R. E. Bixby, "The minimum number of edges and vertices in a graph with edge connectivity *n* and *m n*-bonds", *Network*, vol 5, 1975, pp 253-298.
[3] F. T. Boesch, "Synthesis of reliable network – A survey", *IEEE Trans. Reliability*, vol R-35, 1986, pp 240-246.
[4] S. Even, R. E. Tarjan, "Network flow and testing graph connectivity", *SIAM J. Computing*, vol 4, 1975, pp 507-518.
[5] H. Nagamochi, T. Ibaraki, "Linear time algorithm for finding a sparse *k*-connected spannin subgraph of a *k*-connected graph", (to appear in Algorithmica).
[6] H. Nagamochi, T. Ibaraki, "Computing edge-connectivity in multigraphs and capacitated graphs", (to appear in SIAM J. Disc. Math.).
[7] J. C. Picard, M. Queyranne, "On the structure of all minimum cuts in a network and applications", *Mathematical Programming Study*, vol 13, 1980, pp 8-16.
[8] J. S. Provan, "Bounds on the reliability of networks", *IEEE Trans. on Reliability*, vol R-35, 1986, pp 260-268.
[9] J. S. Provan and M. O. Ball, "The complexity of counting cuts and of computing the probability that a graph is connected", *SIAM J. Computing*, vol 12, 1983, pp 777-788.
[10] J. S. Provan, M. O. Ball, "Computing network reliability in time polynomial in the number of cuts", *Operations Research*, vol 32, 1984, pp 516-521.
[11] Z. Sun, H. Nagamochi, K. Kusunoki, "A graph minimizing the number of cut-sets with a specified number of edges (in Japanese)", *Trans. Inst. Electronics, Information and Communication Engineers of Japan*, Section A, vol 72-A, num 10, 1989, pp 1601-1610.
[12] Z. Sun, H. Nagamochi, K. Kusunoki, "Multiple graphs minimizing the number of minimum cut-sets", *Trans. Inst. Electronics, Information and Communication Engineers of Japan*, Section A, vol 73-E, num 6, 1990, pp 915-921.

AUTHORS

Dr. H. Nagamochi; Dept. of Applied Mathematics & Physics; Faculty of Engineering; Kyoto University; Kyoto 606 JAPAN.

H. Nagamochi is an Assistant Professor in the Department of Applied Mathematics and Physics, Kyoto University. His research interests are network flow problems and graph connectivity problems. He received the BA, ME, and DrE degree from Kyoto University, in 1983, 1985 and 1988.

Z. Sun; Dept. of Information & Computer Sciences; Toyohashi University of Technology; Toyohashi 441 JAPAN.

Z. Sun is working toward a PhD in the Department of Information & Computer Sciences at Toyohashi University of Technology. He received the BE degree from Nanjing University, China, in 1981 and the ME degree from Toyohashi University of Technology in 1988. His research interests include the design of reliable networks and computer architecture.

Dr. T. Ibaraki; Dept. of Applied Mathematics & Physics; Faculty of Engineering; Kyoto University; Kyoto 606 JAPAN.

T. Ibaraki (S'63,M'68) is a Professor in the Department of Applied Mathematics and Physics, Kyoto University. He received the BE, ME, and DrE degrees from Kyoto Univesity in 1963, 1965, and 1970. His research interest includes combinatorial optimization, algorithms and computational complexity. He is the author of *Enumerative Approaches to Combinatorial Optimization*, Baltzer AG, 1988, and co-author of *Resource Allocation Problems: Algorithmic Approaches*, MIT Press, 1988.

Manuscript TR90-008 received 1990 January 23; revised 1990 October 1.

IEEE Log Number 00122

◄TR►

System Failure-Frequency Analysis Using a Differential Operator

(continued from page 609)

AUTHOR

Masahiro Hayashi; NTT Telecommunication Networks Labs.; 3-9-11 Midori-cho; Musashino-shi; Tokyo 180 JAPAN.

Masahiro Hayashi (A'89) was born in Tokushima Prefecture, Japan, on 1962 September 5. He received the BS from Nagoya University of Science in 1986. Since 1986 he was been working at NTT Laboratories. His interest is in reliability engineering and network reliability evaluation. He is a Member of the Institute of Electronics, Information, and Communication Engeneers of Japan, IEEE, and IEEE Reliability Society.

Manuscript TR90-050 received 1990 March 27; revised 1991 February 22.

IEEE Log Number 00121

◄TR►