アルゴリズム 30－2
(1992. 11. 20)

多重辺付加を許したk-辺連結化問題

間島利也　田岡智志　渡辺敏正

広島大学工学部第二類(電気系)
724 東広島市鏡山一丁目4-1
(電話) 0824-22-7111 内3442(渡辺)
(ファクス) 0824-22-7195
(電子メール) watanabe@huis.hiroshima-u.ac.jp

あらまし

本稿の主題は以下のように定義される，グラフのk-辺連結化問題(k-ECA)である："グラフG'=(V,E')と各2点間のコストを表すコスト関数c:V×V→Z^+(非負整数)が与えられたとき，Vの2点を結ぶ辺の集合でG'に付加して得られるグラフG"=(V,E'∪E")がk-辺連結となるもののうちコスト総和が最小となるものE"を求めよ"．k-ECA(M)をG'に多重グラフを許し，且つE"の付加によって新しく多重辺が生じることを許すk-ECAとする．k-ECA(M)に対し4つの近似アルゴリズムFSAM, FSMM, SMCM, HBDMを提案し，これらの近似解の理論的，実験的評価を行う．

和文キーワード　辺付加問題，辺連結度，近似アルゴリズム，グラフ，時間複雑度

The k-Edge-Connectivity Augmentation Problem with Multiple-Edge Addition

Toshiya Mashima, Satoshi Taoka and Toshimasa Watanabe

Department of Circuits and Systems, Faculty of Engineering, Hiroshima University
4-1, Kagamiyama 1-chome, Higashi-Hiroshima, 724 Japan
Phone:+81-824-22-7111 Ext.3442(Watanabe), Fax:+81-824-22-7195
E-mail:watanabe@huis.hiroshima-u.ac.jp

Abstract

The subject of the paper is the *k-edge-connectivity augmentation problem* (k-ECA) defined as follows: "Given a graph G'=(V,E'), and a cost function c :V×V→Z^+ (nonnegative integer), where V×V={{u,v}|u,v∈ V,u≠v}, find a set E" of edges, each connecting distinct vertices of V, of minimum total cost such that G"=(V,E'∪E") is k-edge-connected." Let k-ECA(M) denote k-ECA such that G' may have multiple edges and creation of multiple edges in G" is allowed. Four approximation algorithms FSAM, FSMM, SMCM and HBDM for k-ECA(M) are proposed, and both theoretical and experimental evaluation are given.

英文 key words　Augmentation problems, edge-connectivity, approximation algorithms, graphs, time complexity.

1. Introduction

The *k-edge-connectivity augmentation problem* (k-ECA) is defined as follows: "Given a graph G'=(V,E'), and a cost function $c: V\times V \to Z^+$ (nonnegative integer), where $V\times V=\{\{u,v\}|u,v\in V, u\neq v\}$, find a set E" of edges, each connecting distinct vertices of V, of minimum total cost such that G"=(V,E'∪E") is k-edge-connected." We often denote G" as G'+E". The *edge-connectivity* ec(H) of a graph H is the minimum number of edges whose deletion disconnect it, and H is *k-edge-connected* if ec(H)≥k. Such an edge set E" is called a *minimum solution* to the problem. Costs c({u,v}) for {u,v}∈V×V is denoted as c(u,v) for simplicity. The problem is called the *weighted version*, denoted by W-k-ECA, if there may exist some distinct edge costs and the *unweighted one*, denoted by UW-k-ECA, otherwise. Let k-ECA(S) denote k-ECA with the following restriction (i)-(iii) on G', c and E", respectively: (i) G' is a simple graph; (ii) c:E→Z^+, with E={(u,v)|{u,v}∈V×V} (the edge set of a complete graph having V as the vertex set), such that c(e)=0 if e∈E' and c(e)>0 if e∈E-E'; (iii) E"⊆E-E'. (We assume c(u,v)=1 for any {u,v}∈V×V in UW-3-ECA in the paper.) Let k-ECA(M) denote k-ECA such that G' may have multiple edges and creation of multiple edges in G" is allowed.

Concerning UW-k-ECA, only UW-k-ECA(M) has been discussed so far, while, as for W-k-ECA, only W-k-ECA(S) has been considered. Some known results on W-k-ECA are summarized in the following. For UW-k-ECA(M), See [1,3,5,13,16,18,22] for UW-k-ECA(M) [21,23] for UW-3-ECA(S) and UW-3-ECA(M). For W-2-ECA(S), [4] showed that it remains NP-complete even if G' is restricted to a tree with edge costs from {1,2}. An $O(|V|^2)$ approximation algorithm based on minimum-cost arborescence is presented and it is shown that worst approximation is no more than twice (three times, respectively) the optimum if ec(G')=1 (if ec(G')=0). For W-3-ECA, [19] proved that the problem remains NP-complete even if G' is restricted to a 2-vertex-connected graph with edge costs chosen from {1,2}, and an $O(|V|^3)$ approximation algorithm FS that utilizes a minimum-cost arborescence was presented. Four more approximation algorithms with both theoretical and experimental evaluation are given in [20]. [18] mentioned that both W-k-ECA and W-k-VCA with k≥2 is NP-complete if G' is an empty graph.

The subject of the paper is W-k-ECA(M). For simplicity, k-ECA means W-k-ECA(M) unless otherwise stated. The problem has application to designing robust networks: a k-edge-connected network can survive k-1 communication lines' failure. Let λ=ec(G') and k=λ+δ. (λ+1)-ECA, and general (λ+δ)-ECA is solved by repeatedly finding solutions to (λ+1)-ECA. If λ=0 and δ=1 then the problem is optimally solved by using an algorithm for finding a minimum-cost spanning tree. We assume that λ≥1 in the paper. It is proved in [11,17] that the recognition version of (λ+1)-ECA(S) is NP-complete even if G' is λ-vertex-connected and edge costs are either 1 or 2. This proof technique can be used in proving the NP-completeness of similarly restricted W-k-ECA(M).

Four approximation algorithms FSAM, FSMM, SMCM and HBDM are proposed for (λ+1)-ECA(M). FSAM is based on a minimum-cost arborescence algorithm, and its time complexity is $O(|V|^3)$. FSMM, based on a maximum-cost matching algorithm, runs in $O(|V|^3\log|V|)$ time. SMCM is an alternative to FSMM and is a greedy algorithm finding an edge set of small total cost in $O(|V|^3)$ time. HBDM is a combination of FSMM and SMCM and runs in $O(|V|^3\log|V|)$ time. It is shown theoretically that all of FSAM, FSMM and HBDM produce worst approximation no greater than twice the optimum if all costs are equal. (Note that there is an algorithm that optimally solves UW-k-ECA(M).) As experimental evaluation of W-k-ECA(M), it is shown that

cost(FSMM)<cost(HBDM)<cost(SMCM)<cost(FSAM)

and

time(SMCM)<time(FSAM)<time(HBDM)<time(FSMM),

where cost(FSMM) and time(FSMM) are the total cost of an approximate solution by FSMM and its computation time, and similarly for others. For each of FSMM and SMCM there are cases such that worst approximations cannot be bounded by constants. However, it is experimentally shown that FSMM produces good solutions in cases for which SMCM gives unbounded solutions, and vice versa. Also observed is that HBDM generates sharp approximation in each of these unbounded cases.

For general (λ+δ)-ECA, approximate solutions can be obtained by repeatedly using any one of the four algorithms. Experimental results for the case with 1≤δ≤4 are given. It is also observed that FSMM gives best solutions among the four algorithms.

2. Preliminaries

An (*undirected*) *graph* G=(V(G),E(G)), or simply denoted as G=(V,E), consists of a set V(G) of *vertices* and a set E(G) of *undirected edges*, where an edge between u and v is denotes as (u,v). A *directed graph* $\overrightarrow{G}$=(V($\overrightarrow{G}$),A($\overrightarrow{G}$)), or simply denoted as $\overrightarrow{G}$=(V,A), consists of a set of vertices V($\overrightarrow{G}$) and a set of *directed edges* A($\overrightarrow{G}$). A directed edge from u to v is denoted by <u,v>. The *degree* of a vertex v is denoted as $d_G(v)$, or simply d(v): v is often called a degree-d(v) vertex. We consider a pair of multiple edges as a cycle of length 2.

A *leaf* in a tree is a vertex with only one edge incident on it. An *arborescence* is a directed acyclic graph with one specified vertex, called the *root*, having no entering edges, and all other vertices having exactly one entering edge. A minimum-cost arborescence is an arborescence of minimum total cost. A *cactus* is an undirected connected graph in which any pair of cycles share at most one vertex: each shared vertex is a cutpoint. A leaf of a cactus is a vertex v with d(v)=1 or v' included in a cycle with d(v')=2.

Let S⊆V∪E be any minimal set such that G-S (a graph obtained by deleting all element of S from G) is disconnected. S

is called a *separator* of G, or in particular a (u,v)-separator if u and v are disconnected in G-S. A *minimum separator* S of G is a separator of minimum cardinality among those of G, and |S| is the *edge-connectivity* (denoted by ec(G)) of G in case $S\subseteq E$; particularly such $S\subseteq E$ is called a *minimum* cut (of G). (For a minimum separator S, |S| is the vertex-connectivity vc(G) if $S\subseteq V$.) If |S|=1 then the element of S is called a *cutpoint* in case $S\subseteq V$ or a *bridge* in case $S\subseteq E$. A minimum cut $S\subseteq E$ is often denoted as (X,Y), where $X\cup Y$ is a partition of V such that $S=\{(u,v)\in E|u\in X, v\in Y\}$. G is k-edge-connected (k-vertex-connected) if $ec(G)\geq k$ ($vc(G)\geq k$). For two vertices u, v of G, let $\lambda(u,v;G)$, or simply $\lambda(u,v)$, denote the maximum number of pairwise edge-disjoint paths between u and v. A *k-edge-connected component* (k-ecc for short) of G is a subset $S\subseteq V$ satisfying the following (a) and (b): (a) $\lambda(u,v;G)\geq k$ for any pair $u,v\in S$; (b) S is a maximal set satisfying (a). A 1-ecc is called a *component*.

A *structural graph* F(G) of a given graph G=(V,E) with edge-connectivity $ec(G)=\lambda$ is a representation of all minimum cuts of G. F(G) is an edge-weighted cactus of $O(|V|)$ nodes and edges such that each tree edge has weight λ and each cycle edge has weight $\lambda/2$. Particularly if λ is odd then F(G) is a weighted tree. It is shown that F(G) can be constructed in $O(|V||E|)$ time [9] or $O(|E|+\lambda^2|V|\log(|V|/\lambda))$ time [5]. Each vertex in G maps to exactly one vertex in F(G), and F(G) may have some other vertices, called *empty vertices*, to which no vertices of G are mapped. Let $\varepsilon(G)\subseteq V(F(G))$ denote the set of all empty vertices of F(G). Let $\rho:V(G)\rightarrow V(F(G))-\varepsilon(G)$ denote this mapping, and we represent as $\rho(X)=\{\rho(v)|v\in X\}$ for $X\subseteq V$ and $\rho^{-1}(Y)=\{u\in V|\rho(u)=v, v\in Y\}$ for $Y\subseteq V(F(G))$. F(G) has the following properties: each minimum cut (X,Y) in F(G), with $Y=V(F(G))-X$, corresponds to a minimum one $(\rho^{-1}(X-\varepsilon(G)),\rho^{-1}(Y-\varepsilon(G)))$ in G, and conversely, for each minimum cut (X,Y) in G, there exists at least one partition of $\varepsilon(G)$ into two sets ε_1 and ε_2 such that $(\rho(X)\cup\varepsilon_1,\rho(Y)\cup\varepsilon_2)$ is a minimum cut of F(G). Note that if λ is even then replacing each tree edge by a pair of multiple edges preserves the properties of structural graphs and makes their handling easy because the resulting graphs have no bridges. This graph and a tree in the case where λ is odd are called *modified cactuses*. In the following F(G) denotes a modified cactus unless otherwise stated. Fig.1 shows an example of graph G_1 with $\lambda=2$ and its structural graph $F(G_1)$ as a weighted modified cactus with $\varepsilon(G_1)=\varnothing$ and $(\lambda+1)$-eccs

$\rho^{-1}(a)=\{1\}$, $\rho^{-1}(b)=\{2,3\}$, $\rho^{-1}(c)=\{4\}$, $\rho^{-1}(d)=\{5,7,11\}$, $\rho^{-1}(e)=\{6\}$, $\rho^{-1}(f)=\{8,10\}$, $\rho^{-1}(g)=\{9\}$, $\rho^{-1}(h)=\{12\}$, $\rho^{-1}(i)=\{13\}$, $\rho^{-1}(j)=\{14\}$, $\rho^{-1}(k)=\{15\}$, $\rho^{-1}(l)=\{16\}$, $\rho^{-1}(m)=\{17\}$.

Suppose that $\lambda\geq 2$, and let Z be a set of edges connecting vertices of F(G) and such that $Z\cap E(F(G))=\varnothing$. Neglect edge-weights of F(G), and suppose that $ec(F(G)+Z)\geq 2$ if λ is odd (that is, F(G) is a tree) or $ec(F(G)+Z)\geq 3$ if λ is even (that is, F(G) is a union of some cycles). Then Z is called a *solution* to F(G). For each $(u,v)\in Z$, the set $A(u,v)=\{u,v\}\cup\{w\in V(F(G))-\{u,v\}|w$ is a cutpoint separating u from v$\}$, is called a (u,v)-*augmenting set* of G.

A pair of edges without sharing vertices in G are said to be *independent*. An edge set in which any pair are independent in G is called a *matching* of G, and a matching of maximum cardinality is called a *maximum matching*. A *maximum-cost matching* of a weighted undirected graph is a matching whose total cost is maximum. A maximum cost matching of a given graph G is obtained in $O(|V||E|)$ time [11].

Let $\lceil x\rceil$ ($\lfloor x\rfloor$, respectively) denote the minimum integer not smaller (the maximum one not greater) than x.

3. Approximation Algorithms for W-k-ECA(M)

We first describe a general scheme as an algorithm ECM.

ALGORITHM ECM:

Input: a graph G'=(V,E') with $ec(G')=\lambda$, a cost function $c:V\times V\rightarrow Z^+$.

Output: A set E" of edges, each connecting distinct vertices of V, such that $ec(G'+E")\geq\lambda+\delta$.

1. $H'\leftarrow G'$.
2. Construct a structural graph $G_S'=(V_S,E_S')(=F(H'))$ of H' as in [5] or [10], and $\theta\leftarrow ec(H')$.
3. if $\theta\geq k$ then stop.
4. Define a new cost function $c':V_S\times V_S\rightarrow Z^+$ by $c'(u,v)=\min\{c(x,y)|(x,y)\in V\times V, x\in\rho^{-1}(u)$ and $y\in\rho^{-1}(v)\}$, where $\rho^{-1}(w)$ is a $(\theta+1)$-ecc of H' and is represented as $w\in V_S-\varepsilon(H')$. Define a backpointer $b:V_S\times V_S\rightarrow V\times V$ by $b(u,v)\leftarrow(x,y)$ if $c'(u,v)=c(x,y)$ or $b(u,v)\leftarrow(u,v)$ if $c'(u,v)=\infty$.
5. Find an edge set $E_S"$ of small total cost $c'(E_S")$ such that $E_S"$ is a solution to G_S'.
6. Construct a solution $\Gamma=\{b(u,v)\in V\times V|(u,v)\in E_S"\}$ (with multiplicity deleted) such that $ec(H'+\Gamma)=\theta+1$.
7. $H'\leftarrow H'+\Gamma$, $E"\leftarrow E"\cup\Gamma$, $c(x,y)\leftarrow 0$ for any $(x,y)\in E"$, and goto Step 2. ♦

Constructing G_S', c' and b can be done in $O(\min\{|V||E'|,\Delta\}+|V|^2\log|V|)$ time, where $\Delta=|E'|+\lambda^2|V|\log(|V|/\lambda)$. If $E_S"$ of Step 4 can be found in $O(\xi)$ time then ECM runs in $O(\delta(\xi+\min\{|V||E'|,\Delta\}))$ time.

For a set E" ($E_S"$,respectively) of edges, each connecting distinct vertices of V (V_S), let $c(E")=\sum_{(u,v)\in E"}c(u,v)$, $c'(E_S")=\sum_{(u,v)\in E_S"}c'(u,v)$. The following lemma shows that it suffices to consider W-$(\lambda+1)$-ECA(M) for a cactus $G_S'=(V_S,E_S')$.

Lemma 3.1. If $ec(G'+E")\geq\lambda+1$ for some E" then there is $E_S"$ with $c'(E_S")\leq c(E")$ such that $E_S"$ is a solution to G_S'. For a set $E_S"$, let $b(E_S")=\{b(u,v)|(u,v)\in E_S"\}$ with multiplicity deleted. If $E_S"$ is a solution to G_S' then $ec(G'+b(E_S"))\geq\lambda+1$ and $c'(E_S")\geq c(b(E_S"))$. ♦

Thus the remaining task is to devise efficient algorithms producing good approximate solutions E_S'' to G_S' in Step 5 of ECM for the case where $ec(H')<k$. Four procedures FSAM* (Finding Solution by Arborescence), FSMM* (Finding Solution by Matching), SMCM* (finding solution by Selecting Minimum-Cost edges) and HBDM* (combining FSMM* and SMCM*) are to be proposed for Step 5 of ECM. ECM with FSAM* is denoted as FSAM, and similarly for other procedures. Let L denote the set of all leaves of G_S'.

3.1 Procedure FSAM* Based on Minimum-Cost Arborescence

FSAM* is based on a minimum-cost arborescence algorithm [6]. The algorithm is outlined as follows. First, G_S' is changed into a simple graph by deleting multiplicity, and then a tree G_b' will be constructed as follows: for each cycle C that is remaining in this simple graph, a new vertex v_C is added. For every cycle C, each vertex on C and v_C are connected by an edge. Then all edges of C are deleted. Let G_b' denote the resulting tree. Next, we choose a degree-1 vertex r of G_b' as the root, and direct every edge of G_b' toward r. Let $\overline{G}_b'$ denote the resulting directed graph. Some modification of costs will be done. Let G_b and d' be the complete directed graph on V_b and a final cost function, respectively. We find a minimum-cost arborescence $T=(V_b,A_b'')$ of $\overline{G}_b$ with respect to d'. Finally an approximate solution is obtained by means of backpointers.

We first present procedure REMAKEM that changes a modified cactus G_S' into a spanning tree G_b' in the case where λ is even.

PROCEDURE REMAKEM;
/* *Input*: $G_S'=(V_S,E_S')$. *Output*: $G_b'=(V_b,E_b')$. */
1. If λ is odd then $G_b'\leftarrow G_S'$ and stop.
2. Delete multiplicity of edges in G_S', making it simple. Then find all cycles (equal to 2-eccs of G_S') by a depth-first-search.
3. For each cycle C, add a dummy vertex w_C, connect w_C to every vertex on C and delete all edges (u,v) of C. Let $G_b'=(V_b,E_b')$ be the resulting graph. Extend the domains c' and b to $V_b\times V_b$ as follows: for all dummy vertices w_C and all vertices u of V_b, $c'(w_C,u)\leftarrow\infty$ and $b(w_C,u)\leftarrow\varnothing$. ♦

We use a distance function $d:E_b\rightarrow Z^+\cup\{\infty\}$. This function was introduced in [4] in order to avoid poor choice of edges in finding a minimum-cost arborescence. In this paper

$$d(u,v)=\min\{c'(x,y)\mid u \text{ and } v \text{ are on a path from } x \text{ to } y \text{ in } G_b'\}.$$

PROCEDURE DISTM;
/* *Input*: a graph $G_b'=(V_b,E_b')$ of G_b, a cost function $c':V_b\times V_b\rightarrow Z^+\cup\{\infty\}$ and a backpointer b. */
/* *Output*: A distance function $d:V_b\times V_b\rightarrow Z^+\cup\{\infty\}$ and a backpointer $b':V_b\times V_b\rightarrow V_b\times V_b$ such that $c'(b'(u,v))=d(u,v)$. */
1. For each pair of vertices u and v, compute the number a(u,v) of edges on the path between u and v in G_b', find the vertex s(u,v) adjacent to v (s(v,u) adjacent to u, respectively) on this path, $d(u,v)\leftarrow c'(u,v)$ and $b'(u,v)\leftarrow(u,v)$.
2. Bucketsort the pairs (u,v) of $V_b\times V_b$ into nonincreasing order of a(u,v). For each pair (u,v) in $V_b\times V_b$ do the following in its sorted order:
$d(u,s(u,v))\leftarrow d(u,v)$ and $b'(u,s(u,v))\leftarrow b'(u,v)$ if $d(u,v)<d(u,s(u,v))$;
$d(s(v,u),v)\leftarrow d(u,v)$ and $b'(s(v,u),v)\leftarrow b'(u,v)$ if $d(u,v)<d(s(v,u),v)$. ♦

We can show that DISTM correctly computes the distance function d and the backpointer b' in $O(|V|^2)$ time. For two different edges $(u,v),(u',v')\in V_b\times V_b$, it may happen that $b'(u,v)=b'(u',v')$. Hence, in general, $b'(Z)=\{b'(u,v)\mid(u,v)\in Z\}$ may be a multiset for a set $Z\subseteq V_b\times V_b$. However, we assume that b'(Z) denotes the one with multiplicity deleted unless otherwise stated, for notational simplicity. Similar notation will be used for other backpointers to be defined later. Procedure FSAM* is stated as follows.

*ROCEDURE FSAM**;
/* *Input*: a structural graph $G_S'=(V_S,E_S')$ with $ec(G')=\lambda$, a cost function $c':V_S\times V_S\rightarrow Z^+\cup\{\infty\}$, and a backpointer $b:V_S\times V_S\rightarrow V\times V$. */
/* *Output*: A set E_S'' of edges, each connecting distinct vertices of V_S, such that E_S'' is a solution to G_S'. */
1. Construct a complete graph G_b from G_S, and a tree G_b' from G_S' by using *REMAKEM*.
2. Compute $d:V_b\times V_b\rightarrow Z^+$ and $b':V_b\times V_b\rightarrow V_b\times V_b$ by using *DISTM*.
3. $A_b\leftarrow\varnothing$. Insert <u,v> and <v,u> into A_b for each pair $\{u,v\}\in V_b\times V_b$, constructing a complete directed graph $\overline{G}_b=(V_b,A_b)$.
4. Choose any degree-1 vertex of G_b' as the root r. Let A_b' be the set of directed edges generated by directing each edge in E_b' toward r. Denote the resulting graph by $\overline{G}_b'=(V_b,A_b')$.
5. $d'<u,v>\leftarrow d(u,v)$ and $d'<v,u>\leftarrow d(u,v)$ for all $\{u,v\}\in V_b\times V_b$.
6. For $<u,v>\in A_b$, if $u\notin V_S$ or $v\notin V_S$ then $d'<u,v>\leftarrow\infty$; if $<u,v>\in A_b'$ and $\{u,v\}\subseteq V_S$ then $d'<u,v>\leftarrow 0$ and $d<v,u>\leftarrow\infty$; if v=r then $d'<u,r>\leftarrow\infty$.
7. Find a minimum-cost arborescence $T=(V_b,A_b'')$ with the root r of $\overline{G}_b$ with respect to d'.
8. For each edge <u,v> in A_b'' with $0<d'<u,v><\infty$, insert the corresponding edge b'(u,v) into E_S'' (with multiplicity deleted). ♦

Remark 3.1. In Step 5 of FSAM* we may define d' and b'' as in Step 5 of FSMM* (to be given in 3.2) by finding shortest paths. We implemented this version and applied it to 120 data, but no improvement has been observed so far. Hence in this paper Step 5 of FSAM* is left as it is. ♦

The following lemma can be proved.

Lemma 3.2. FSAM* generates a set of directed edges A_b'' such that $(V_b,A_b'\cup A_b'')$ is strongly connected. ♦

We obtain the next theorem.

Theorem 3.1. FSAM generates E" with $ec(G'+E'')\geq\lambda+1$ in $O(|V|^2+\min\{|V||E'|,\Delta\})$ time.
♦

3.2 Procedure FSMM* Based on Maximum-Cost Matchings

FSMM is based on a maximum-cost matching algorithm. The idea is very simple. As a set of $\lceil|L|/2\rceil$ edges, each connecting a pair of leaves in a cactus G_S', the one E''_0 of minimum total cost is selected by using a maximum-cost matching algorithm in $O(|V_S|^3)$ time [15]. If E_0'' is not a solution to G_S' then similar process will be repeated for $G_S'+E_0''$: repetition is at most $O(\log|V_S|)$ time. The description of FSMM* is given as follows, where we denote $G_i'=(V_i, E_i')$ and $H_i'=(W_i, F_i')$.

*PROCEDURE FSMM**:
/* *Input*: a graph $G_S'=(V_S,E_S')$ with $ec(G_S')=\lambda$, a cost function $c':V_S\times V_S\rightarrow Z^+\cup\{\infty\}$, and a backpointer $b:V_S\times V_S\rightarrow V\times V$. */
/* *Output*: A set E_S'' of edges, each connecting distinct vertices of V_S, such that E_S'' is a solution to G_S'. */
1. $G_0'\leftarrow G_S'$, $c_0'\leftarrow c'$, $i\leftarrow 0$.
2. Construct a tree G_b' from G_i' by using REMAKEM.
3. $H_i'=(W_i, F_i')\leftarrow G_b'$.
4. Find a distance function $d_i':W_i\times W_i\rightarrow Z^+\cup\{\infty\}$ with a back pointer $b_i':W_i\times W_i\rightarrow V_i\times V_i$ by using DISTM. For each dummy vertex w_C added within the cycle C of G_S', $d_i'(w_C,v)\leftarrow\infty$ for any vertex v on C.
5. Compute $d_i'':W_i\times W_i\rightarrow Z^+\cup\{\infty\}$ and $b_i'':W_i\times W_i\rightarrow W_i\times W_i$ by finding a shortest path P(u,v) in a complete graph on V_i for each pair u,v of degree-1 vertices of H_i' as follows: for the edge set E_P of P(u,v), $d_i''(u,v)\leftarrow d_i(E_P)$ and $b_i''(u,v)\leftarrow E_P$ (actually $b_i''(u,v)$ is a pointer to the list maintaining E_P).
6. Construct a complete subgraph S_i, on the set of all degree-1 vertices of H_i'.
7. For each cost $d_i'(u,v)$ with $(u,v)\in E(S_i)$, $d_i''(u,v)\leftarrow MAX+1-d_i'(u,v)$, where MAX is the maximum edge-cost of S_i. Find a maximum-cost matching $M_i\subseteq E(S_i)$ of S_i with respect to d_i''.
8. For each edge $(u,v)\in M_i$, insert the corresponding set of edges $b_i'(b_i''(u,v))$ into E_i'' (and then any multiplicity is deleted).
9. If E_i'' is not a solution to G_i' then $\Omega\leftarrow E_i''$ execute (i)-(iii):
(i) $G_{i+1}'\leftarrow G_i'$;
(ii) while $\Omega\neq\emptyset$ repeat (a) and (b):
(a) choose $(u,v)\in\Omega$ and $\Omega\leftarrow\Omega-\{(u,v)\}$;
(b) $G_{i+1}'\leftarrow$ the graph obtained by shrinking the (u,v)-augmenting set A(u,v) of G_{i+1}'; (Note that the resulting G_{i+1}' is also a modified cactus.)
(iii) define a new cost function $c_{i+1}:V_{i+1}\times V_{i+1}\rightarrow Z^+\cup\{\infty\}$ by $c_{i+1}'(u,v)=\min\{c_i'(x,y)|(x,y)\in V_i\times V_i,\ x\in S_u,\ y\in S_v\}$.
where S_w denotes the set of vertices of $G_i'+\Omega$ that are shrunk into $w\in V_{i+1}$ and the edge (x,y) is referenced by a backpointer $b_{i+1}(u,v)=(x,y)$, where $b_{i+1}:V_{i+1}\times V_{i+1}\rightarrow V_i\times V_i$.
(iv) $i\leftarrow i+1$ and goto step 2.
10. If E_i'' is a solution to G_i' then do the following (i) and (ii):
(i) $E_S''\leftarrow E_0''$;
(ii) if $i\geq 1$ then for each j (j=1, ..., i) repeat $E_S''\leftarrow E_S''\cup b_1(...(b_j(E_j''))...)$. ♦

Theorem 3.2. FSMM generates E" with $ec(G'+E'')\geq\lambda+1$ in $O(|V|^3\log|V|+\min\{|V||E'|,\Delta\})$ time. ♦

3.3 Procedure SMCM* Based on Minimum-Cost Edges

SMCM* is a greedy algorithm that finds approximate solutions without using a maximum-cost matching algorithm. The description of SMCM* is almost the same as that of FSMM*. The only difference is that SMCM* finds a solution to G_S' by choosing, at each leaf v, a minimum-cost edge e_v among those incident upon v. The description of SMCM* is almost the same as that of FSMM*. The only difference is that SMCM* finds a solution to G_S' by choosing, at each leaf v of H_i', a minimum-cost edge e_v among those incident upon v. Delete Step 5, replace Step 7 by the following statement in FSMM* and rewrite $b_i'(b_i''(u,v))$ as $b_i'(u,v)$ in Step 8:

Step 7. For each degree-1 vertex v of H_i', let e_v denote an edge connecting a pair $(u,v)\in V_i\times V_i$ with
$d_i'(u,v)=\min\{d_i'(u',v)|(u',v)\in V_i\times V_i\}$, and
$M_i\leftarrow\{e_v|v$ is a degree-1 vertex of $H_i'\}$
(with multiplicity deleted).

Theorem 3.3 SMCM generates E" with $ec(G'+E'')\geq\lambda+1$ in $O(|V|^3+\min\{|V||E'|,\Delta\})$ time. ♦

3.4. Procedure HBDM* Combination of FSMM* and SMCM*

HBDM* is a combination of FSMM* and SMCM*, and is almost the same as that of FSMM*. The only difference is that a maximum-cost matching algorithm, to be repeatedly used in finding a solution to G_S', is applied to the set $E(K_i)=\{e_v|v$ is a degree-1 vertex of $H_i'\subseteq E(S_i)\}$ (with multiplicity deleted) instead of $E(S_i)$, where $E(K_i)$ will be constructed similarly to Step 7 of SMC, but is slightly different from it. If we replace Step 7 of FSMM* by the following statement then the description of HBDM* is obtained.

Step 7. For each degree-1 vertex v of H_i', let f_v denote an edge $(u,v)\in E(S_i)$ with $d_i'(u,v)=\min\{d_i'(u',v)|(u',v)\in E(S_i)\}$, and $E(K_i)\leftarrow\{f_v|v$ is a degree-1 vertex of $H_i'\}$ (with multiplicity deleted). Let K_i denote the subgraph $(V(S_i),E(K_i))$ of S_i. For each cost $d_i'(u,v)$ with $(u,v)\in E(K_i)$, $d_i''(u,v)\leftarrow MAX+1-d_i'(u,v)$, where MAX is the maximum edge-cost of K_i. Find a maximum-cost matching $M_i\subseteq E(K_i)$ of K_i with respect to d_i''.

Theorem 3.4 HBDM generates E" with $ec(G'+E'')\geq\lambda+1$ in $O(|V|^3\log|V|+\min\{|V||E'|,\Delta\})$ time. ♦

4. Evaluation of Worst Approximation

Worst approximation by proposed algorithms is evaluated

theoretically for unweighted cases and experimentally for weighted cases. (Since UW-k-EC'A(M) can be solved optimally in polynomial time, evaluation in unweighted cases is easy.) Let OPT or APP denote total cost of an optimum or approximate solution, respectively. The ratio APP/OPT is going to be evaluated concerning $(\lambda+\delta)$-ECA (mainly for the case with $\delta=1$).

4.1. Theoretical Evaluation of Worst Approximation for $(\lambda+1)$-ECA

For UW-$(\lambda+1)$-ECAM it is known that $OPT=\lceil q/2 \rceil$ [13,18], where $q=|L|$. Theoretical evaluation for FSAM, FSMM and HBDM are given. It suffices to consider solutions to G_S'.

4.1.1. FSAM. Since a minimum arborescence to be found contains exactly q-1 edges, we have APP=q-1 and, therefore, $APP/OPT \le (q-1)/(q/2)=2-2/q<2$.

4.1.2. FSMM and HBDM. We can prove that $lim_{q\to\infty}(APP/OPT)\le 2$. Let G_0 denote a complete graph on $n_0(=q)$ degree-2 vertices of G_S'. Let M_0 be a maximum matching of G_0, and $e_0=|M_0|=\lfloor n_0/2 \rfloor \le q/2$. $G_1=G_0+M_0$ has at most $n_1=\lceil n_0/2 \rceil \le q/2+1/2$ degree-2 vertices. In general, for $i=1,...m=\lceil log q \rceil -1$, $G_i=G_{i-1}+M_{i-1}$ has at most $n_i=\lceil n_{i-1}/2 \rceil$ degree-2 vertices, where *log*q is abbreviation of $log_2 q$ and

$$n_i=\lceil n_{i-1}/2 \rceil \le q/(2^i)+1/(2^i)+...+1/2.$$

If M_i is a maximum matching of G_i then its cardinality $e_i=|M_i|=\lfloor n_i/2 \rfloor$, where

$$\lfloor n_i/2 \rfloor \le q/(2^{i+1})+1/(2^{i+1})+...+1/(2^2).$$

Hence $G_{i+1}=G_i+M_i$ has at most $n_{i+1}=\lceil n_i/2 \rceil$ degree-2 vertices. At the final stage,

$$n_m=\lceil n_{m-1}/2 \rceil \le q/(2^m)+1/(2^m)+...+1/2=3-2/n<3$$

and $n_m=2$. Hence $e_m=1$ and $n_{m+1}=0$. Thus the total number of edges added is

$$e_0+e_1+...+e_{m-1}+e_m$$
$$\le (q/2)+\{q/(2^2)+1/(2^2)\}+...+\{q/(2^m)+1/(2^m)+...+1/(2^2)\}+1$$
$$=q-5/2+(1/2)log q+2/q.$$

We can prove that

$$APP/OPT \le (q-5/2+(1/2)log q+2/q)/(q/2)$$
$$=2-5/q+(log q)/q+4/(q^2)$$

and

$$lim_{q\to\infty}(APP/OPT)\le 2.$$

It seems that this is slightly overestimated, since only APP/OPT<2 has been observed so far in our experimentation by FSMM and HBDM.

4.2. Experimental Evaluation

4.2.1. Input data. We explain how input data, G' with ec(G')=k and c, are constructed as in Steps 1-3.

1. The number |V| of vertices is given as follows:

$$|V| \in \{10,15,20,40,60,80,100,120,140,160,180,200\}.$$

2. Two types of data are provided: type C and type T constructed as follows.

2.1. (type C)

(1) If $|V|\ge 40$ then $i\leftarrow 4$ and partition $V=\{1,...,|V|\}$ into i sets $L_1,...,L_i$ with $|L_i|=|V|/4$ else (that is, $|V|\le 20$) $i\leftarrow 1$ and $L_i\leftarrow V$. Let $G_i=(L_i,E_i)$ with $E_i=\varnothing$ for each i.

(2) For each i, do the following (a)-(d).

(a) $q\leftarrow 0$.

(b) While $k-q\ge 2$ do the following: construct a new cycle C of $|L_i|$ vertices randomly, $G_i\leftarrow(L_i,E_i\cup E(C))$ and $q\leftarrow q+2$.

(c) If $k-q=1$ then construct a new tree T of $|L_i|$ vertices randomly and $G_i\leftarrow(L_i,E_i\cup E(T))$.

(d) Choose an integer h_i with $1\le h_i\le |L_i|/2$ randomly, and add h_i edges to G_i randomly unless the resulting graph G_i' has $ec(G_i')\ge k+1$.

2.2. (type T)

(1) Partition $V=\{1,...,|V|\}$ into n sets $L_1,..,L_n$, where n is given randomly.

(2) For each i, $1\le i\le n$, partition L_i into two sets W_i with $|W_i|=k$ and $W_i^*=L_i-W_i$, join any pair of W_i by k multiple edges, and repeat the following exactly k times for each $u\in W_i^*$: choose a vertex $v\in W_i$ randomly and add an edge (u,v). Let G_i denote the resulting graph, and $R\leftarrow\{G_1,...,G_n\}$.

(3) Choose any $G_i\in R$, $R\leftarrow R-\{G_i\}$ and $G\leftarrow G_i$. Then repeat the following (i)-(iii) until R becomes empty: (i) choosing $G_j\in R$, (ii) $R\leftarrow R-\{G_j\}$, and (iii) $G\leftarrow$ the graph given by coalescing a pair of vertices, one from G and the other from G_j, where a pair of vertices are chosen randomly.

3. Costs on edges are selected randomly from the set $\{1,...,99\}$ of integers.

4.2.2. Experimental results. We have tried 4320 data so far. A workstation SUN SPARC station is used. Tables 1 through 5 show a part of our experimental results for data $|V|\le 200$ and $|E|\le 2095$ of type C. For 180 data with $|V|=10,15,20$, optimum solutions are sought by exhaustive search. Table 1 shows a part of the results, and error $(APP/OPT-1)\times 100$ (%) is computed for these 180 data as summarized in Table 2. Table 3 shows a part of results for large data. Tables 1 through 3 show results for $\delta=1$ (that is, concerning $(\lambda+1)$-ECA(M)), while Table 5 shows those for $\delta>1$, where $\lambda=1$ and $1\le\delta\le 4$. Table 4 shows the average of the ratio cost(*)/cost(FSMM) over 60 data of type C and of type T for each $\lambda\in\{1,2,3,4,5,6,10,15,20\}$ concerning $(\lambda+1)$-ECA(M), where * denotes any one of FSAM, FSMM, SMCM and HBDM.

Let time(FSMM) and cost(FSMM) denote computation time and total cost for FSMM, respectively, and similarly for others. Experimental results show the following (1)-(4).

(1) In general,

cost(FSMM)<cost(HBDM)<cost(SMCM)<cost(FSAM).

(2) We have 180 data to each of which an optimum solution is found by exhaustive search. For 155 data (86.12%) of them, FSMM generates approximate solutions with errors (APP/OPT-1)×100≤5%.

(3) For data of type C (type T, respectively), each algorithm produces better solutions for λ that is odd (even) rather than even (odd).

(4) In general,

time(SMCM)<time(FSAM)<time(HBDM)<time(FSMM).

4.2.3. Unbounded approximation. We show two examples for which FSAM, FSMM or SMCM generates solutions to W-k-ECA(M) such that APP/OPT fails to be bounded by constants. For the graphs shown by solid lines in Figs. 2 and 3, suppose that they represent G_S' and that all costs except those specified in the figures are very large, where M>5 in Fig. 2. For Fig. 2, an optimum solution is shown in halftone lines and OPT=5. FSAM finds a solution with total cost APP=2M+5, and APP/OPT=2M/5+1. FSMM generates a solution {(a,g),(b,f),(c,e),(d,h)} with total cost APP=2M+2, and APP/OPT=2M/5+2/5.

For Fig. 3, we have OPT=2 (the edge (a,g)), while SMCM generates a solution shown by halftone lines and APP=|V|-1(=6). Hence APP/OPT=(|V|-1)/2 (=3). It is observed in our experimentation that FSMM produces good solutions for data to which SMCM gives unbounded solutions, and vice versa. This is why we propose HBDM: it produces sharp approximation for these unbounded data.

5. Concluding Remarks

This paper proposed four approximation algorithms FSAM, FSMM, SMCM and HBDM for W-k-ECA(M), and both theoretical and experimental evaluation of their approximate solutions are given. The following (1) through (3) are left for future research:

(1) theoretical evaluation of HBDM, which is being continued (our conjecture is APP/OPT≤2);

(2) proposing approximation algorithms of better approximation for W-k-ECA(M);

(3) providing more experimental results on k-ECA with $k=\lambda+\delta$ and $\delta\geq 2$.

References

[1] K.P.Eswaran and R.E.Tarjan, Augmentation problems, SIAM J.Comput, 5, 653-655 (1976).

[2] S.Even, Graph Algorithms, Pitman, London (1979).

[3] A.Frank, Augmenting graphs to meet edge connectivity requirements, SIAM J. Discrete Mathematics, Vol. 5, No. 1, 25-53 (February 1992).

[4] G.N.Fredericson and J.Ja'ja', Approximation algorithms for several graph augmentation problems, SIAM J.Comput., 10, 270-283 (1981).

[5] H.N.Gabow, Applications of a poset representation to edge connectivity and graph rigidity, Proc. 32nd IEEE Symp. Found. Comp. Sci., 812-821 (1991).

[6] H.N.Gabow, Z.Galil, T.Spencer and R.E.Tarjan, Efficient algorithms for finding minimum spanning trees in undirected and directed graphs, Combinatorica, 6(2), 109-122 (1986).

[7] Z.Galil and G.F.Italiano, Reducing edge connectivity to vertex connectivity, SIGACT NEWS, 22, 57-61 (1991).

[8] M.R.Garey and D.S.Johnson, Computers and Intractability: a Guide to the Theory of NP-Completeness, Freeman, San Francisco (1978).

[9] J.E.Hopcroft and R.E.Tarjan, Dividing a graph into triconnected components, SIAM J. Comput., 2, 135-158 (1973).

[10] A.V.Karzanov and E.A.Timofeev, Efficient algorithm for finding all minimal edge cuts of a nonoriented graph, Cybernetics, 156-162, Translated from Kibernetika, No.2, 8-12 (March-April, 1986).

[11] T.Mashima, S.Taoka and T.Watanabe, Approximation Algorithms for the k-Edge-Connectivity Augmentation Problem, Tech. Rep. of IEICE of Japan, Tech. Reserch Rep., COMP92-24, 11-20 (1992).

[12] H.Nagamochi and T.Ibaraki, A linear time algorithm for computing 3-edge-connected components in a multigraph, Japan J. Industrial and Applied Math., Vol.9, No.7(June 1992), 163-180.

[13] D.Naor, D.Gusfield and C.Martel, A fast algorithm for optimally increasing the edge-connectivity, Proc. 31st Annual IEEE Symposium on Foundations of Computer Science, 698-707 (1990).

[14] S.Taoka, T.Watanabe and K.Onaga, A linear time algorithm for computing all 3-edge-connected components of an multigraph, Trans. IEICE, E75-3, 410-424 (1991).

[15] R.E.Tarjan, Data Structures and Network Algorithms, CBMS-NSF Regional Conference Series in Applied Mathematics, SIAM, Philadelphia, PA (1983).

[16] S.Ueno, Y.Kajitani, and H.Wada, The minimum augmentation of trees to k-edge-connected graphs, Networks, 18, 19-25 (1988).

[17] T.Watanabe,T.Mashima, and S.Taoka, The k-edge-connectivity augmentation problem of weighted graphs, Proc. 3rd International Symposium on Algorithms and Computation(ISAAC'92)(Dec. 1992), to appear.

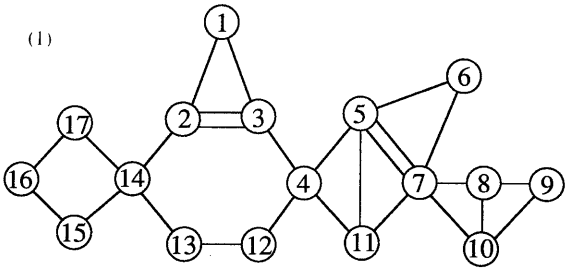
[18] T.Watanabe and A.Nakamura, Edge-connectivity augmentation problems, Journal of Computer and System Sciences, 35, 96-144 (1987).

[19] T.Watanabe, T.Narita and A.Nakamura, 3-Edge-connectivity augmentation problems, Proc. 1989 IEEE ISCAS, 335-338 (1989).

[20] T.Watanabe, S.Taoka and T.Mashima, Approximation algorithms for the 3-edge-connectivity augmentation problem of graphs, Proc. IEEE Asia-Pacific Conference on Circuits and Systems, to appear (Dec. 1992).

[21 T.Watanabe, S.Taoka and T.Mashima, Minimum-cost augmentation to 3-edge-connect all specified vertices in a graph, submitted to 1993 IEEE ISCAS93 (September 1992).

[22] T.Watanabe, M.Yamakado and K.Onaga, A linear-time augmenting algorithm for 3-edge-connectivity augmentation problems, Proc. 1991 IEEE ISCAS, 1168-1171 (1991).
[23] T.Watanbe and M.Yamakado, A linear time algorithm for smallest augmentation to 3-edge-connect a graph, submitted to Trans. IEICE of Japan (September 1992).



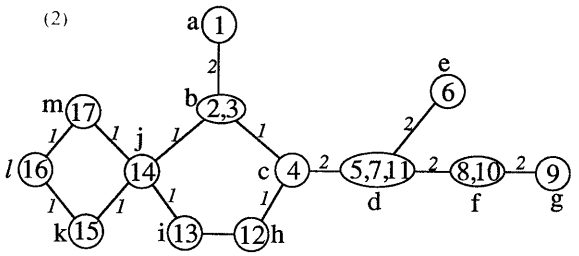


Fig.1. An example of a graph G_1 with $ec(G_1)=2$ and structural graph $F(G_1)$: (1) G_1; (2) $F(G_1)$, where each edge has a weight $\lambda=2$ or $\lambda/2=1$.

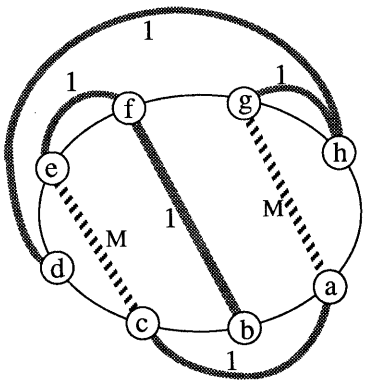


Fig.2. An example of G for which each of FSAM and FSMM generates a solution whose total cost cannot be bounded by a constant.

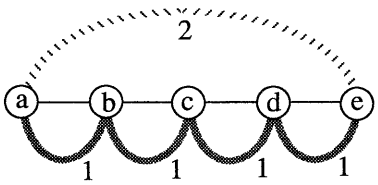


Fig.3. An example of G for which SMCM generates a solution whose total cost cannot be bounded by a constant.

Table 1. A part of our experimental results (1080 data in total) on $(\lambda+1)$-ECA(M) for small data, where $\lambda=ec(G')=4$, $\delta=1$, type C and $\zeta=|V_{sl}|$: The columns "cost", "AP/OP" and "time" show total cost (left), ratio APP/OPT(middle) and CPU time with unit time in 1/60 second (right), respectively.

#	|V|	|E'|	λ	ζ	FSAM			FSMM			SMCM			HBDM			OPT	
					cost	AP/OP	time	cost	AP/OP	time	cost	AP/OP	time	cost	AP/OP	time	cost	time
401	10	23	4	6	67	1	4	67	1	4	67	1	4	67	1	4	67	75
402	10	23	4	7	76	1.3818	5	55	1	4	55	1	3	65	1.1818	5	55	64
403	10	23	4	5	36	1	4	36	1	4	36	1	3	36	1	4	36	51
404	10	23	4	6	44	1.2222	5	36	1	4	44	1.2222	4	44	1.2222	4	36	17
405	10	22	4	7	67	1.2407	4	54	1	5	54	1	4	54	1	6	54	110
406	15	37	4	8	47	1.3056	6	36	1	7	47	1.3056	6	50	1.3889	9	36	149
407	15	32	4	12	63	1	8	63	1	10	63	1	7	63	1	9	63	1116
408	15	36	4	6	17	1	7	17	1	7	17	1	6	17	1	6	17	82
409	15	32	4	16	75	1.0135	13	74	1	13	74	1	10	75	1.0135	13	74	8910
410	15	33	4	11	42	1.2353	9	34	1	9	41	1.2059	8	34	1	9	34	339
411	20	44	4	14	56	1.2444	12	49	1.0889	14	49	1.0889	10	49	1.0889	13	45	2661
412	20	48	4	8	18	1	10	18	1	10	18	1	9	18	1	11	18	215
413	20	48	4	10	34	1.0303	11	33	1	11	33	1	9	37	1.1212	11	33	2187
414	20	42	4	17	47	1.093	16	43	1	16	43	1	13	43	1	15	43	3384
415	20	47	4	9	32	1	11	32	1	10	32	1	9	32	1	11	32	1399

Table 2. Total number of data (left) and its ratio (right), for which each algorithm produces solutions with errors (APP/OPT-1)×100(%) falling into the corresponding intervals. $1 \le \lambda \le 6$, $\delta=1$ and the total number of data is 180, to each of which an optimum solution is found by exhaustive search: (1) type C (90 data); (2) type T (90 data); (3) combining the two types (180 data).

(1)

err.	err.=0%		0%<err.≤5%		5%<err.≤10%		10%<err.≤15%		15%<err.	
FSAM	41	45.56%	8	8.89%	11	12.22%	8	8.89%	22	24.44%
FSMM	73	81.11%	7	7.78%	8	8.89%	1	1.11%	1	1.11%
SMCM	57	63.33%	8	8.89%	10	11.11%	9	10.00%	6	6.67%
HBDM	58	64.44%	8	8.89%	14	15.56%	4	4.44%	6	6.67%

(2)

err.	err.=0%		0%<err.≤5%		5%<err.≤10%		10%<err.≤15%		15%<err.	
FSAM	15	16.67%	12	13.33%	17	18.89%	14	15.56%	32	35.56%
FSMM	63	70.00%	12	13.33%	9	10.00%	2	2.22%	4	4.44%
SMCM	44	48.89%	13	14.44%	15	16.67%	10	11.11%	8	8.89%
HBDM	38	42.22%	13	14.44%	18	20.00%	10	11.11%	11	12.22%

(3)

err.	err.=0%		0%<err.≤5%		5%<err.≤10%		10%<err.≤15%		15%<err.	
FSAM	56	31.11%	20	11.11%	28	15.56%	22	12.22%	54	30.00%
FSMM	136	75.56%	19	10.56%	17	9.44%	3	1.67%	5	2.78%
SMCM	101	56.11%	21	11.67%	25	13.89%	19	10.56%	14	7.78%
HBDM	96	53.33%	21	11.67%	32	17.78%	14	7.78%	17	9.44%

Table 3. A part of our experimental results (1080 data in total) on $(\lambda+1)$-ECA(M) for large data, where columns are similar to those of Table 1.

#	|V|	|E'|	λ	ζ	FSAM		FSMM		SMCM		HBDM	
					cost	time	cost	time	cost	time	cost	time
416	40	91	4	38	92	61	77	60	88	41	79	56
417	40	106	4	17	35	32	33	34	35	31	33	35
418	40	98	4	27	31	44	25	46	29	37	26	42
419	40	95	4	26	64	40	59	42	57	35	60	42
420	40	101	4	22	44	40	40	40	40	33	40	39
421	60	141	4	49	59	114	48	122	53	90	49	109
422	60	146	4	31	58	83	51	90	53	75	51	86
423	60	138	4	41	68	103	62	110	64	84	64	104
424	60	153	4	28	55	79	49	83	49	72	50	84
425	60	152	4	29	42	82	38	85	40	74	38	84
426	80	185	4	56	76	189	65	202	69	157	67	198
427	80	198	4	40	60	152	51	158	52	136	54	155
428	80	195	4	41	62	161	55	165	55	140	56	160
429	80	178	4	63	80	200	69	224	70	162	71	198
430	80	204	4	31	46	141	42	144	43	132	42	142
431	100	236	4	68	72	289	66	331	68	247	67	300
432	100	217	4	85	101	355	85	425	87	282	88	380
433	100	246	4	44	46	236	39	244	40	222	41	247
434	100	235	4	58	92	259	79	303	81	234	81	279
435	100	217	4	82	125	346	99	423	105	276	107	349
436	120	252	4	113	127	610	100	784	107	467	102	604
437	120	257	4	108	136	578	105	764	110	457	108	590
438	120	288	4	60	83	360	66	382	70	332	69	365
439	120	281	4	67	71	383	60	420	62	347	61	403
440	120	275	4	84	92	428	70	502	75	361	73	424
441	140	322	4	86	109	553	89	646	91	487	92	612
442	140	300	4	120	118	740	94	994	100	598	97	776
443	140	302	4	120	130	756	98	1006	103	606	101	780
444	140	323	4	88	112	602	96	714	98	529	97	633
445	140	303	4	124	125	750	96	1021	105	606	101	778
446	160	345	4	132	130	973	94	1283	104	791	99	978
447	160	351	4	120	111	863	89	1073	94	735	91	890
448	160	372	4	91	90	703	76	801	78	637	78	734
449	160	380	4	75	82	675	73	749	77	636	72	721
450	160	363	4	95	109	730	89	860	93	656	90	792
451	180	435	4	82	80	861	67	942	70	812	67	890
452	180	390	4	152	147	1278	114	1847	127	1071	116	1377
453	180	387	4	144	145	1181	103	1572	108	983	105	1237
454	180	391	4	141	142	1186	101	1529	118	994	105	1230
455	180	413	4	105	110	988	89	1153	94	876	90	1043
456	200	469	4	112	105	1148	81	1340	90	1046	83	1187
457	200	474	4	105	98	1149	79	1286	84	1040	80	1188
458	200	492	4	92	88	1078	74	1186	79	1012	74	1100
459	200	444	4	140	133	1304	98	1701	106	1150	102	1392
460	200	448	4	140	134	1371	99	1696	109	1196	104	1454

Table 4. The average of the ratio cost(*)/cost(FSMM) over 60 data (5 data for each $|V| \in \{10,15,20,40,60,80,100,120,140,160,180,200\}$) for each $\lambda \in \{1,2,3,4,5,6,10,15,20\}$ concerning $(\lambda+1)$-ECA(M), where * denotes any one of FSAM, FSMM, SMCM and HBDM: (1) type C; (2) type T; (3) combining the two types.

(1)

	FSAM	FSMM	SMCM	HBDM	#data
λ=1	1.1088	1	1.0338	1.0069	60
λ=2	1.1959	1	1.0461	1.0184	60
λ=3	1.0842	1	1.0278	1.0067	60
λ=4	1.1896	1	1.0543	1.0343	60
λ=5	1.0972	1	1.0216	1.0057	60
λ=6	1.1651	1	1.0428	1.0236	60
λ=10	1.1718	1	1.0451	1.0229	60
λ=15	1.1141	1	1.0199	1.0051	60
λ=20	1.1751	1	1.0461	1.0197	60
total	1.1446	1.0000	1.0375	1.0159	540

(2)

	FSAM	FSMM	SMCM	HBDM	#data
λ=1	1.2306	1	1.0654	1.0394	60
λ=2	1.1883	1	1.0434	1.0249	60
λ=3	1.1752	1	1.0511	1.0266	60
λ=4	1.1874	1	1.0499	1.0191	60
λ=5	1.2029	1	1.0723	1.0262	60
λ=6	1.1720	1	1.0396	1.0244	60
λ=10	1.1686	1	1.0361	1.0239	60
λ=15	1.1681	1	1.0369	1.0195	60
λ=20	1.1756	1	1.0411	1.0246	60
total	1.1854	1.0000	1.0484	1.0254	540

(3)

	FSAM	FSMM	SMCM	HBDM	#data
type C	1.1446	1	1.0375	1.0159	540
type T	1.1854	1	1.0484	1.0254	540
total	1.1650	1.0000	1.0430	1.0207	1080

Table 5. A part of our experimental results (4320 data in total) on $(\lambda+\delta)$-ECA(M) with $\lambda=1$ and $1 \le \delta \le 4$, where columns are similar to those of Table 1.

#	|V|	|E'|	λ	δ	FSAM		FSMM		SMCM		HBDM	
					cost	time	cost	time	cost	time	cost	time
140	120	163	1	1	26	34	25	31	25	16	25	32
				2	118	131	101	172	102	79	102	159
				3	215	519	191	684	200	446	192	600
				4	317	952	309	1348	316	883	303	1210
145	140	200	1	1	43	83	39	47	40	21	39	41
				2	111	114	107	188	108	86	108	204
				3	207	665	193	850	202	581	196	860
				4	314	1337	302	1712	312	1179	306	1814
150	160	204	1	1	49	91	42	91	43	43	42	107
				2	143	295	127	456	133	191	132	471
				3	255	1103	230	1688	244	937	234	1534
				4	366	1946	348	3167	355	1782	352	2896
155	180	201	1	1	90	358	75	433	78	146	75	421
				2	195	571	181	1005	185	354	183	1011
				3	323	1602	292	2414	310	1249	294	2327
				4	459	2770	434	4327	459	2340	434	4111
160	200	232	1	1	77	287	66	323	68	131	66	335
				2	189	652	180	1104	183	394	176	945
				3	313	1973	289	2998	309	1481	290	2571
				4	443	3378	432	5215	447	2762	429	4556