グラフの多重辺付加を許さない4辺連結化問題

高藤 大介　田岡 智志　渡辺 敏正
広島大学 工学部 第二類 回路システム工学講座
(724)東広島市鏡山一丁目 4 － 1
(電話)0824-24-7662 (ファクシミリ)0824-22-7195
(電子メール)watanabe@huis.hiroshima-u.ac.jp

概要：重みなしのk辺連結化問題(UW-4ECAと略記)とは，与えられた無向グラフG=(N,A)に付加すれば得られるグラフG'=(N,A∪A')がk辺連結となるような最小辺集合A'を求める問題である．G，G'共に単純グラフとしたUW-4ECAをUW-4ECA(S,SA)と表し，Gは多重グラフでもよいがG'構成時の多重辺付加は許さない場合をUW-4ECA(*,SA)と表す．本稿ではUW-4ECA(S,SA)に対する$O(|N|^2)$アルゴリズムを提案する．これはUW-4ECA(*,SA)にも応用できる．また，葉グラフと呼ばれるグラフの最大マッチングの辺数も示している．本結果は未解決問題である一般的なUW-kECA(*,SA)の解決に向けての第一歩である．

Simplicity-Preserving Augmentation to 4-Edge-Connect a Graph

Daisuke Takafuji, 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: +81-824-24-7662 (Watanabe) Facsimile: +81-824-22-7195
E-mail: watanabe@huis.hiroshima-u.ac.jp

Key words: connectivity augmentation problems, edge-connectivity, 4-edge-connectedness, maximum matchings, polynomial-time algorithms

Abstract: The unweighted k-edge-connectivity augmentation problem (UW-kECA) is defined by "Given a graph G=(N,A), find an edge set A' of minimum cardinality, with each element connecting distinct vertices of N, such that G'=(N,A∪A') is k-edge-connected." Let UW-4ECA(S,SA) denote UW-kECA such that both G and G' are simple. The paper proposes an $O(|N|^2)$ algorithm for solving UW-4ECA(S,SA). The result can be used in solving UW-4ECA(*,SA) in which G may have multiple edges and creation of new multiple edges in constructing G' is not allowed. Also presented is the cardinality of a maximum matching of a leaf graph that is constructed from G, and the results may be interesting from viewpoint of combinatorial theory. The paper is a first step toward an open problem UW-kECA(*,SA) for general k≥4.

1. Introduction

The unweighted *k-edge-connectivity augmentation problem* (UW-kECA for short) is defined by "Given a graph G=(N,A), find an edge set A' of minimum cardinality, with each element connecting distinct vertices of N, such that G'=(N,A∪A') is k-edge-connected." We often denote G' as G+A', and such an edge set A' is called a *solution* to the problem. Let UW-kECA(*,**) denote UW-kECA with the following restriction (i) and (ii) on G and A', respectively: (i) * is set to S if G is required to be simple, and * means G may be a multiple graph; (ii) ** is set to MA if creation of new multiple edges in constructing G' is allowed, and is set to SA otherwise. Let n=|N| and m=|A|.

The paper proposes an $O(n^2)$ algorithm for solving UW-4ECA(S,SA). The result can be used in solving UW-4ECA(*,SA). Also presented is the cardinality of a maximum matching of a leaf graph, and the results may be interesting from viewpoint of combinatorial theory. The paper is a first step toward general UW-kECA(*,SA) that is an open problem.

Concerning the weighted k-edge-connectivity augmentation problem whose definition can be identified in [12,26,27], see [3,6,12,25,26,31,32,33]. For the vertex-connectivity augmentation problem similarly defined, see [3,6,27,28,29] for example. [24] shows summary of results on (vertex- or edge-) connectivity augmentation problems up to autumn of 1990.

As for UW-kECA, UW-kECA(*,MA) has mainly been discussed so far. Some known results on UW-kECA(*,MA) are summarized in the following. Eswaran and Tarjan proposed in [3] an $O(n+m)$ algortihm for UW-2ECA(*,MA), and Watanabe and Yamakado proposed in [34] an O(n+m) algorithm for UW-3ECA(*,MA). Watanabe and Nakamura first proposed in [27] an $O(kLn^4(kn+m))$ algorithm for UW-kECA(*,MA), where L=min{k,n}. Ueno, Kajitani and Wada discussed in [21] UW-kECA(*MA) with G restricted to a tree, where exact time complexity was not given. For UW-kECA(*,MA), Frank proposed in [4] an $O(n^5)$ algorithm, which can handle directed connectivity augmentation problems. Naor, Gusfield and Martel showed in [17] an $O(\delta^2 nm+n\cdot F(n,m))$ algorithm by utilizing structural graphs introduced in [11], where δ=k-ec(G) and F(n,m) is the time to perform one maximum flow on a graph with n vertices and m edges. It should be mentioned that these time complexities are given for simple graphs: in multigraphs m=|A| should appear in time complexity of [4], and [17] stated that $F(n)\le O(\min\{n^{2/3}m, m^{3/2}\})$ (see [2] for example) which holds only for simple graphs; for multigraphs, we have $F(n,m)\le O(\min\{n,m^{1/2}\}\cdot m)$. Therefore time complexity of [17] may be rewritten as $O(\delta^2 nm+\min\{n,m^{1/2}\}\cdot nm)$ for multigraphs. Some improvement is reported by Gabow in [7].

The problem that we are interested in is UW-kECA(*,SA). It has been made clear from our research process that UW-kECA(S,SA) is the main problem and UW-kECA(*,SA) can be solved by extending the results of UW-kECA(S,SA). As known results on UW-kECA(*,SA), UW-kECA(S,SA) was first discussed in [5] which is a pioneering paper on connectivity augmentation problems: it is shown that an unweighted local edge-connectivity augmentation problem starting with graphs without edges is polynomially solvable, where the problem is more general than UW-kECA(S,SA) and exact time complexity is not given. An $O(n+m)$ algorithm for UW-2ECA(S,SA) can be obtained by slightly modifying the one given in [3] for UW-2ECA(*,MA), and we can show that if n≥3 then this algorithm finds an optimum solution to UW-2ECA(*,MA) such that it is also an optimum one to UW-2ECA(*,SA), and vice versa. As for UW-3ECA(*,SA), [34] proposed an $O(n+m)$ algorithm for UW-3ECA(*,MA) based on the results by [8,15,18], and showed that if n≥4 then the algorithm finds an optimum solution to UW-3ECA(*,MA) such that it is also an optimum one to UW-3ECA(*,SA), and vice versa.

The subject of the paper is UW-4ECA(S,SA) for 3-edge-connected graph G, which will be a first step toward UW-kECA(*,SA). A central concept in solving UW-kECA is a *t-component* of G: a maximal set of vertices such that G has at least t edge-disjoint paths between any pair of vertices in the set. A t-component whose degree is λ (=ec(G)) is called a *leaf*. Although UW-4ECA(S,SA) can be solved almost similarly to UW-kECA(*,MA), the only difference is that the augmenting step has to choose a pair of leaves, each containing a vertex such that these vertices are not adjacent in G. Unfortunately this requirement makes structural graphs useless, where structural graphs introduced in [11] is used in reducing time complexity in solving UW-kECA(*,MA) in [7,17]. Instead of structural graphs, the paper adopts the operation, called *edge-interchange*, in finding a solution, where it was introduced in [22,23] in order to reduce time complexity of [27]. Such a pair of leaves as mentioned above is called a *nonadjacent pair* of leaves, and two nonadjacent pairs of leaves is called *D-combination* if they are disjoint. The main point of the augmenting step used in solving UW-4ECA(S,SA) in this paper is to repeat both choosing a nonadjacent pair of leaves and enlarging a (λ+1)-component by means of edge-interchnge. Hence obtaining an optimum solution requires finding maximum number of nonadjacent pairs such that any two pairs is a D-combination, and it is reduced to finding a maximum matching of a certain graph R(G), called a *leaf-graph*, constructed from G. We can avoid obtaining a maximum matching, which is a time-consuming process, except the case where the number of leaves is small. Based on these observations, an $O(n^2)$ algorithm for solving UW-4ECA(S,SA) is proposed. Some basic definitions are given in Section 2, Section 3 discusses maximum matchings of R(G), Section 4 explains edge-exchange, and solving UW-4ECA(S,SA) is considered in Section 5.

Almost all proofs of propositions are omitted due to shortage of space: they will be given elsewhere.

2. Preliminaries

2.1. Basic Definitions

Technical terms not specified here can be identified in [1,2,9,10,19]. An *undirected graph* G=(V(G),E(G)) consists of a finite and nonempty set of vertices V(G) and a finite set of undirected edges E(G); an edge e incident upon two vertices u,v in G is denoted by (u,v); u and v are called the *endvertices* of an edge e, where e is called a *loop* if u=v. V(G) and E(G) are often denoted as V and E, respectively. If there are two edges both of which have the same pair of endvertices then G is called a *multigraph*. Such edges are called *mutiple edges*. G is called a *simple graph* if G has no loops or no multiple edges. In this paper, the term "a graph" means an undirected multigraph without loops unless otherwise stated. For disjoint sets S, $S' \subset V(G)$, we denote $C(S,S';G)=\{(u,v)\in E(G) | u\in S$ and $v\in S'\}$, where it is often written as C(S,S') if G is clear from the context. If S'=V(G)-S then it is written as C(S;G), and we denote $d_G(S)=|C(S;G)|$. This is called the *degree* of S (in G). If S={v} then $d_G(\{v\})$ is denoted simply as $d_G(v)$ and is the total number of edges (v,v'), $v'\neq v$, incident upon v.

A *path* between u and v, or a *(u,v)-path*, is an alternating sequence of vertices and edges $u=v_0,e_1,v_1,...,v_{n-1},e_n,v_n=v$ ($n\geq 0$) such that $v_0,...,v_n$ are all distinct and if $n\geq 1$ then $e_i=(v_{i-1},v_i)$ for each i, $1\leq i\leq n$. The *length* of this path is n. A *cycle* is a (v_0,v_n)-path together with an edge (v_0,v_n). The length of this cycle is n+1. A pair of multiple edges are considered as a cycle of length two. A component of G is a maximal set of vertices such that any pair u, v in the set is connected by a path of G. Two paths P, P' are said to be *edge-disjoint* if $E(P)\cap E(P')=\emptyset$. For two vertices u, v of G, let M(u,v;G), or simply M(u,v), denote the maximum number of pairwise edge-disjoint paths between u and v.

For a set $R\subseteq V(G)$, let G[R] denote the subgraph having R as its vertex set and $\{(u,v)\in E(G) | u,v\in R\}$ as its edge set. G[R] is called the *subgraph* of G *induced* by R (or the *induced subgraph* of G by R). *Deletion* of $R\subseteq V(G)$ from G is to construct G[V(G)-R], which is often denoted as G-R. If R={v} then we often denote G-v for simplicity. Deletion of $Q\subseteq E(G)$ from G defines a spanning subgraph of G, denoted by G-Q, having E(G)-Q as its edge set. If Q={e} then we denote G-e. For a set E' of edges such that $E'\cap E(G)=\emptyset$, let G+E' denote the graph $(V(G),E(G)\cup E')$. If E'={e} then we denote G+e. Shrinking $R\subseteq V(G)$ to a vertex r_0 is to construct a graph, denoted by $G<R,r_0>$, which has $(V(G)-R)\cup\{r_0\}$ as its vertex set and $E(G-S)\cup\{(r_0,v) | (u,v)\in E(G),\ u\in R,\ v\in V(G)-R\}$ as its edge set, where any loop created is deleted.

Let $S\subseteq V(G)\cup E(G)$ be any minimal set such that G-S has more components than G. S is called a *separator* of G, or in particular a (X,Y)-separator if any vertex of X and any one of Y are disconnected in G-S. If X={u} or Y={v} then it is denoted as a (u,Y)-separator or a (X,v)-separator. A *minimum* (X,Y)-*separator* S of G is a (X,Y)-separator of minimum cardinality. Such S is often calld a (X,Y)-cut if $S\subseteq E(G)$. It is known that a (u,v)-cut S has |S|=M(u,v;G). A *minimum separator* S of G is a separator of minimum cardinality among all separators of G, and if $S\subseteq E(G)$ then |S| is called the *edge-connectivity* (denoted by ec(G)) of G; particularly we call such $S\subseteq E(G)$ a *minimum* cut (of G). A *minimum separator* S of G with |S|=1 is called a *cutvertex* if $S\subseteq V(G)$ or a *bridge* if $S\subseteq E(G)$. G is said to be *k-edge-connected* if $ec(G)\geq k$. A *k-edge-connected component* (k-component, for short) of G is a subset $S\subseteq V(G)$ satisfying the following (a) and (b): (a) $M(u,v;G)\geq k$ for any pair $u,v\in S$; (b) S is a maximal set that satisfies (a). Let $\Gamma(k)$ denote the set of all k-components of G. In a graph G with $ec(G)=\lambda$, a $(\lambda+1)$-component S with $d_G(S)=\lambda+1$ is called a *leaf* $(\lambda+1)$-*component* of G. If H is a graph obtained by shrinking each $(\lambda+1)$-component S of G into an individual vertex v_S then a vertex u with $d_H(u)=\lambda$ is called a *leaf* of H. Any leaf $(\lambda+1)$-component of G corresponds to a leaf of H, and vice versa. For simplicity, we often call a leaf $(\lambda+1)$-component of G as a leaf unless any confusion arises. It is known that $ec(G)\geq k$ if and only if V(G) is a k-component. Note that distinct k-components are disjoint sets. Each 1-component is often called a *component*.

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 matching of a given graph G is obtained in $O(|V(G)|^{1/2}|E(G)|)$ time [13]. A maximum matching M of G with |M|=|V(G)|/2 is called a *complete matching*, where |V(G)| is necessarily even. G is called a *bipartite gaph* if V(G) is partitioned into two sets X, Y(=V(G)-X) such that E(G)=C(X,Y;G). A *complete bipartite graph* is a bipartite one such that any pair $x\in X$ and $y\in Y$ are connected. Let $\lceil x\rceil$ ($\lfloor x\rfloor$, respectively) denote the minimum integer not smaller (the maximum one not greater) than x.

2.2. λ-Components and Leaf-Graphs

Let G=(N,A) be a simple graph with $ec(G)=\lambda$, and let S(G)=(V,E) denote the graph obtained by shrinking each $(\lambda+1)$-component S of G into individual vertex v_S. R(G) or S is often denoted by H or $L(v_S)$, respectively. Let u denote a representative of L(v), where $u\in L(v)$, and we choose u from L(v) whenever necessary. Let $L(v_1)$, $L(v_2)$ be distinct $(\lambda+1)$-components of G. The pair $\{v_1,v_2\}$ and even the pair $\{L(v_1),L(v_2)\}$ are called an *adjacent pair* (denoted as $v_1\chi v_2$) if any two vertices $w\in L(v_1)$ and $w'\in L(v_2)$ are adjacent in G or called a *nonadjacent pair* (denoted as $v_1\bar{\chi}v_2$) otherwise. Let R(G)=(V',E') be defined by

$V'=\{v | v$ is a leaf of S(G)$\}$ and

$E'=\{(v,v') | v,v'\in V'$ and $v\bar{\chi}v'\}$,

and is called the *leaf-graph* of S(G) (or of G). Let v_i, i=1,2,3,4, be distinct leaves of S(G). Two nonadjacent pairs $\{v_1,v_2\},\{v_3,v_4\}$ are called a D-*combination*. In general, for 2t distinct leaves v_i, i=1,...,2t with $t\geq 2$, of S(G), t (≥ 2) nonadjacent pairs $\{v_1,v_2\},...,\{v_{2t-1},v_{2t}\}$ are called a D-*set* of R(G). Let $v_1\chi\{v_2,v_3\}$ denote that both $v_1\chi v_2$ and $v_1\chi v_3$ hold. A D-combination $\{v_1,v_2\},\{v_3,v_4\}$ is called an I-*combination* if $v_1\chi\{v_3,v_4\}$ or $v_2\chi\{v_3,v_4\}$ holds, and is denoted as $\{v_1,v_2\}\angle\{v_3,v_4\}$. We first show some basic results on leaves of G.

Proposition 2.1. Either |L(v)|=1 or $|L(v)|\geq\lambda+2$ holds for $\forall v\in V'$. ♦

Proposition 2.2. If $\{v_1,v_2\}\subseteq V'$ is an adjacent pair then $|L(v_1)|=|L(v_2)|=1$. ♦

Concerning R(G) we obtain the following two propositions.

Proposition 2.3. $d_{R(G)}(v)\geq\max\{|V'|-(\lambda+1),0\}$ for any leaf $v\in V'$. ♦

Proposition 2.4. R(G) is connected if $|V'|\geq 2\lambda+1$ and $\lambda\geq 3$. ♦

Proposition 2.5. Let $Y=\{v_1,v_2,v_3,v_4\}\subseteq V'$, where all elements are distinct. Then (1) and (2) hold.

(1) Y contains at least one nonadjacent pair.

(2) If $\{v_1,v_2\}$ is a nonadjacent pair and we have a pair $\{v_5,v_6\}\subseteq V'-Y$ then there is a nonadjacent pair $\{w_1,w_2\}\subseteq\{v_3,v_4,v_5,v_6\}$. ♦

2.3. Examples

Let G=(N,A) with $|N|\geq 5$ and ec(G)=3 be any given simple graph. Let OPT(M) or OPT(S) denote the cardinality of an optimum solution to UW-4ECA(*,MA) or to UW-4ECA(S,SA) for G, respectively. We give two examples such that OPT(S)=OPT(M)+1. Fig. 3.1 shows a graph G with $|V'|=4$. S(G) and R(G) are shown in Figs. 3.2 and 3.3, respectivel. $\{(u_1,u_3),(u_2,u_4)\}$ is a solution to UW-4ECA(*,MA), while $A'=\{(u_1,u_6),(u_2,u_5),(u_3,u_4)\}$ is a solution to UW-4ECA(S,SA) and OPT(S)=3=OPT(M)+1. Fig. 3.4 shows another graph G with $|V'|=6$. In this example, S(G)=G, and R(G) is shown in Fig. 3.5. $\{(u_1,u_4),(u_2,u_5),(u_3,u_6)\}$ is a solution to UW-4ECA(*,MA), while $A'=\{(u_1,u_4),(u_2,u_5),(u_3,u_4),(u_5,u_6)\}$ is a solution to UW-4ECA(S,SA) and OPT(S)=4=OPT(M)+1. ♦

3. Maximum Matchings of Leaf-Graphs

One of requirements in finding an optimum solution to UW-4ECA is to obtain a largest D-set. Hence, in this section, the cardinality of a maximum D-set is investigated by considering a maximum matching M of R(G). For a set $U\subseteq N$ of G=(N,A), let T(U) denote the total number of odd components of G-U, where an *odd component* is a component having odd number of vertices. A famous theorem proved by Tutte in [20] on complete matchings is very useful in the following discussion, and is stated as follows.

Theorem 3.1 [20]. A graph G=(N,A) has a complete matching if and only if $T(U)-|U|\leq 0$ for $\forall U\subseteq N$. ♦

Let M denote a maximum matching of R(G) in the following discussion unless otherwise stated, where we assume that $ec(G)=\lambda$ with $\lambda\geq 3$.

Proposition 3.1. $|M|=|V'|/2$ if R(G) is connected, $|V'|\geq 2\lambda$ and $|V'|$ is even. ♦

Note that Proposition 2.4 shows that R(G) is connected if $|V'|\geq 2\lambda+1$ with $\lambda\geq 3$. Hence we have the following corollary.

Corollary 3.1. $|M|=|V'|/2$ if $|V'|\geq 2\lambda+1$ and $|V'|$ is even. ♦

We can easily prove the following corollary by applying Proposition 3.1 to the graph obtained by adding to R(G) a new vertex v_0 and new edges (v_0,v) for $\forall v\in V'$.

Corollary 3.2. $|M|=\lfloor |V'|/2\rfloor$ if $|V'|\geq 2\lambda-1$ and $|V'|$ is odd. ♦

Proposition 3.2. Let R(G) be disconnected and $|V'|=2\lambda$. Then (1) or (2) holds:

(1) If λ is even then $|M|=\lfloor |V'|/2\rfloor$ $(=\lambda)$.

(2) If λ is odd then $|M|=\lfloor |V'|/2\rfloor-1$ $(=\lambda-1)$ and G=(N,A) is a complete bipartite graph with $N=X\cup Y$, $X\cap Y=\emptyset$ and $|X|=|Y|=\lambda$.♦

Combining Corollaries 3.1, 3.2 and Proposition 3.2, we obtain the following proposition.

Proposition 3.3. Let $|V'|\geq 2\lambda-1$. Then (1) or (2) holds:

(1) $|M|=\lfloor |V'|/2\rfloor-1$ $(=\lambda-1)$ and G=(N,A) is a complete bipartite graph with $N=X\cup Y$, $X\cap Y=\emptyset$ and $|X|=|Y|=\lambda$, if R(G) is disconnected, $|V'|=2\lambda$ and λ is odd.

(2) $|M|=\lfloor |V'|/2\rfloor$ $(=\lambda)$ otherwise. ♦

It is easy to see that the next proposition holds for M.

Proposition 3.4. For each edge $(v_1,v_2)\in M$, V'-V(M) contains no pair of vertices v,v', $v\neq v'$, such that $\{(v,v_1),(v',v_2)\}\subseteq E'-M$. ♦

Proposition 3.5. $|V'|-\lambda\leq|M|\leq\lfloor |V'|/2\rfloor$ if $\lambda+1\leq|V'|\leq 2\lambda-2$. ♦

Proposition 3.6. $1\leq|M|\leq\lfloor |V'|/2\rfloor$ if $|V'|=\lambda$. ♦

4. Augmentation by Edge-Interchange

Let $\lambda=ec(G)<k$, $k\geq 2$, and Z be a solution to UW-kECA. Z has a partition $Z=Z(\lambda+1)\cup\cdots\cup Z(k)$ such that

$ec(G_i)=ec(G_{i-1})+1$ and $ec(G_i-e)=ec(G_{i-1})$ $(\forall e\in Z(i))$,

where $G_\lambda=G$ and $G_i=G_{i-1}+Z(i)$, $i=\lambda+1,\ldots,k$. In this section we introduce an operation called edge-interchange such that we can construct Z(j) by repeating this operation for each j, $j=\lambda+1,\ldots,k$. In the following we consider only $Z(\lambda+1)$.

4.1. Attachments

Let H denote S(G) for notational simplicity. In H, we have $d_H(v)\geq\lambda$ and $M_H(v,v')=\lambda$ for $\forall v,v'\in V(H)$, $v\neq v'$. Let

$Y=\{y_1,\ldots,y_q\}$, $q\geq 2$,

denote the set of all leaves (vertices $v\in V(H)$ with $d_H(v)=\lambda$) of H, and let $r=\lceil q/2\rceil$. We denote $V(e)=\{u,v\}$ for an edge e=(v,v') and $V(F)=\cup_{e\in F}V(e)$ for an edge set F. We call F an *attachment* (for H) if and only if the following (1) through (4) hold:

(1) $V(F)\subseteq Y$,

(2) $F\cap E(H)=\emptyset$,

(3) $V(e)\neq V(e')$ $(\forall e,e'\in F,\ e\neq e')$, and

(4) F has at most one pair f,f' such that $|V(f)\cap V(f')|=1$.

Let F be any attachment for H. For each $e=(u,v)\in F$, H+F has a new $(\lambda+1)$-component, denoted by A(e,H+F), containing V(e). We can easily prove the next proposition.

Proposition 4.1. If there is an attachment F for H such that $V(F)=Y\subseteq A$ for some $A\in\Gamma_{H+F}(\lambda+1)$ then A=V(H). ♦

Finding $Z(\lambda+1)$ consists of two steps. The first step is to find a minimum attachment $Z_H=\{e_1,\ldots,e_r\}$ such that $ec(H+Z_H)=\lambda+1$. Although there are two cases: r=1 and $r\geq 2$, we discuss only the latter case in the following. (Note that if r=1 then we immediately obtain the desired attachment F.) The second step is to determine $Z(\lambda+1)$ from Z_H. This step is a routine work and, therefore, only the first step is explained.

4.2. Finding a minimum attachment Z_H

Suppose that there are an attachment F for H and vertices $v_{ij}\in Y-V(F)$, $1\leq i,j\leq 2$, where v_{11},v_{12},v_{21} are distinct, and if v_{22} is equal to one of the other three then we assume that $v_{22}=v_{21}$ (see Fig. 4.1). We use the following notations:

$$L=H+F,\ e=(v_{11},v_{12}),\ e'=\begin{cases}(v_{21},v_{22}) & \text{if } v_{21}\neq v_{22},\\ (v_{12},v_{21}) & \text{if } v_{21}=v_{22},\end{cases}$$

$A(e)=A(e,L+\{e,e'\})$, $A(e')=A(e',L+\{e,e'\})$

$f_1=(v_{11},v_{21})$, $f_2=(v_{12},v_{22})$, $f_3=(v_{11},v_{22})$,
$f_4=(v_{12},v_{21})$,
(where we set $f_1=f_3$ and $e'=f_2=f_4$ if $v_{21}=v_{22}$.)

$$A(f_i)=\begin{cases} A(f_i,L+\{f_1,f_2\}) & \text{if } 1\le i\le 2,\\ A(f_i,L+\{f_3,f_4\}) & \text{if } 3\le i\le 4.\end{cases}$$

(Note that e, e', $f_i\notin E(L)$, $1\le i\le 4$.) We have two cases

Case I: $A(e)\cap A(e')=\varnothing$, and

Case II: $A(e)\cap A(e')\ne\varnothing$ (that is, $A(e)=A(e')$).

In Case I, we will show that there are two edges f,f' with $V(f)\cup V(f')=V(e)\cup V(e')$ such that

$V(e)\cup V(e')\subseteq A(f,L+\{f,f'\})=A(f',L+\{f,f'\})$.

That is, we can add two edges so that the resulting $(\lambda+1)$-component contains $V(e)\cup V(e')$. Finding and adding such a pair of edges f, f' is called *edge-interchange* (with respect to $V(e)\cup V(e')$). Case II considers the situation in which there are distinct vertices $v'',w''\in Y-(V(F)\cup V(e)\cup V(e'))$ such that

$A(e',L'+\{e',e''\})\cap A(e'',L'+\{e',e''\})=\varnothing$

for a new edge $e''=(v'',w'')$, where $L'=L+e$. We will show that there are edges f',f'' such that

$V(f')\cup V(f'')=V(e')\cup V(e'')$ and

$A(e,L')\subseteq A(f',L'+\{f',f''\})=A(f'',L'+\{f',f''\})$.

This means that we can add two edges so that the resulting $(\lambda+1)$-component always contains not only the two endvertices of the added edges but also the $(\lambda+1)$-component constructed in Case I.

Case I: $A(e)\cap A(e')=\varnothing$. Note that $v_{21}\ne v_{22}$ in this case. Let K be any fixed $(A(e),A(e'))$-cut of $L+\{e,e'\}$, and let B_i, $1\le i\le 2$, denote the two sets of $L+\{e,e'\}$ such that $B_1\cup B_2=V$, $B_2=V-B_1$, $K=C(B_1;L+\{e,e'\})$, $A(e)\subseteq B_1$ and $A(e')\subseteq B_2$. $|K|=\lambda=M_{L''}(v_1,v_2)$ for $\forall v_i\in B_i$, $1\le i\le 2$, where L'' denotes L, $L+e$, $L+e'$ or $L+\{e,e'\}$. K is a (v_1,v_2)-cut of L. Suppose that f and f' satisfy either (i) or (ii):

(i) $f=f_1$, $f'=f_2$, or (ii) $f=f_3$, $f'=f_4$,

where $\{f,f'\}\cap E(L)=\varnothing$. Note that, for any $b\in\{v_{21},v_{22}\}$, we can consider that $(L+e)<B_2,b>=(L+\{e,e'\})<B_2,b>$ (see Fig. 4.2). The next proposition follows from Theorem 3.1 of [27], and it is used explicitly or implicitly in the following discussion.

Proposition 4.2. Suppose that $A(e)\cap A(e')=\varnothing$. Then the following holds for $I=(L+e)<B_2,b>$ and I' that denotes either $L+e$ or $L+\{e,e'\}$: $M_{I'}(v,v')=M_I(v,v')$ and $M_{I'}(v,b)=M_I(v,b)$ for $\forall v,v'\in B_1$. ♦

For Case I we obtain the following proposition.

Proposition 4.3. If $A(e)\cap A(e')=A(f_1)\cap A(f_2)=\varnothing$ then $A(f_3)\cap A(f_4)\ne\varnothing$, that is, $A(f_3)=A(f_4)$. ♦

Corollary 4.1. Let f_3 and f_4 be the two edges of Proposition 4.3, $L'=L+\{f_3,f_4\}$ anf f be either f_3 or f_4. Then L'-f has no λ-cut separating $V(f_3)$ from $V(f_4)$. That is, if L'-f has a λ-cut K separating a vertex of $V(f_3)$ from another one of $V(f_4)$ then K separates $\{u\}$ from $\{v\}\cup V(f')$ and $V(f')$ is not separated by K, where $V(f)=\{u,v\}$ and $\{f'\}=\{f_3,f_4\}-\{f\}$. ♦

Case II: $A(e)=A(e')$. Put

$e=(v,w)$, $e'=(v',w')$, $L'=L+e$,

and suppose that there are distinct vertices $v'',w''\in Y-(V(F)\cup V(e)\cup V(e'))$ such that

$A(e',L'+\{e',e''\})\cap A(e'',L'+\{e',e''\})=\varnothing$,

where $e''=(v'',w'')\notin E(L'+e')$. By Proposition 4.3, there are edges f', f'' such that

$A(f',L'+\{f',f''\})=A(f'',L'+\{f',f''\})$,

$V(f')\cup V(f'')=V(e')\cup V(e'')$ and $V(f')\cap V(f'')=\varnothing$.

We assume that $f'=(v',w'')$ and $f''=(v'',w')$ (see Fig. 4.3). The next proposition follows from Corollary 4.1.

Proposition 4. 4. $A(e,L'+e')\subseteq A(f',L'+\{f',f''\})$. ♦

Propositions 4.3 and 4.4 show that if $\lambda>0$ then repeating edge-interchange finds a sequence of edges $e_1,\dots,e_r$ ($r=\lceil q/2\rceil\ge 1$) such that

$A(e_i,H_i)\subseteq A(e_{i+1},H_{i+1})$, $1\le i\le r-1$,

$V(e_{j-1})\cap V(e_j)=\varnothing$, $2\le j\le r-1$, and

$$V(e_{r-1})\cap V(e_r)=\begin{cases}\varnothing & \text{if } q \text{ is even,}\\ \{y_q\} & \text{if } q \text{ is odd,}\end{cases}$$

where $H_0=H$, and $H_{i+1}=H_i+e_{i+1}$, $0\le i\le r-1$. Since $A(e_r,H_r)=V(H)$ by Proposition 4.1, we obtain the following proposition.

Proposition 4.5. $Z_H=\{e_1,\dots,e_r\}$ is a minimum attachment such that $ec(H+Z_H)=\lambda+1$. ♦

Another important property of edge-interchange is given as follows.

Proposition 4.5. Suppose that $q\ge 3$. Then $A(e_i,H_i)$ is a leaf of H_i if and only if q is odd and $i=r-1$. ♦

Remark 4.1. Let f,f' be the two new edges such that

$V(f)\cap V(f')=\varnothing$, and $V(f)\cup V(f')=\{v_{11},v_{12},v_{21},v_{22}\}$

as in Proposition 4.3. Suppose that we are going to check whether $A(f,H_i+\{f,f'\})\cap A(f',H_i+\{f,f'\})=\varnothing$ or not. A maximum flow algorithm can be used. Note that we have only to apply the algorithm to $H+\{f,f'\}$ (not to $H_i+\{f,f'\}$) or to $G+\{g,g'\}$, where

$V(g)\cap V(g')=\varnothing$, and $V(g)\cup V(g')=\{u_{11},u_{12},u_{21},u_{22}\}$

with $u_{ij}\in L(v_{ij})$, $i,j=1,2$. Thus this can be done in $O(F(n',m'+2))$ time, where n' and m' are the number of vertices and of edges of H. [14] introduced a sparse graph $G''=(N,E'')$ for a given graph $G=(N,A)$ such that the following (1) through (3) hold for any $u,v\in N$:

(1) $M(u,v;G'')\ge\lambda$ if $M(u,v;G)\ge\lambda$,

(2) $M(u,v;G'')=M(u,v;G)$ if $M(u,v;G)<\lambda$,

(3) $E''\subseteq A$ and $|E''|\le\lambda|N|$.

[14] showed that G'' can be obtained in $O(|N|+|A|)$ time. By utilizing the results in [2], above checking operation can be done in $O(\lambda|N|)$ time. ♦

5. Solving UW-4ECA(S,SA)

It is shown in the preceding section that repeating edge-interchange constructs Z_H from which we obtain $Z(\lambda+1)$. Adding $Z(\lambda+1)$ to G, however, may create multiple edges. Hence we have to choose nonadjacent leaves of H during edge-interchange in order to solve UW-4ECA(S,SA). This is done in this section by combining the results given in Sections 3 and 4.

Let $G=(N,A)$ with $|N|\geq 5$ and $ec(G)=3$ be any given graph. Let $V'=\{v_1,...,v_q\}$ ($q=|V'|$), $I=\{1,...,q\}$ and $L_I=\{L(v_i)|\ i\in I\}$. Let #D denote the number of nonadjacent pairs of a maximum D-set of R(G). An edge set A' of minimum cardinality such that G+A' is a simple graph with ec(G+A')=4 is called a solution.

Proposition 5.1. If q=2 then the following (1) or (2) holds.

(1) If $v_1\bar{\chi}v_2$ then $|M|=1$, $A'=\{(u_1,u_2)\}$ is a solution, and OPT(S)=OPT(M)=1.

(2) If $v_1\chi v_2$ then $|M|=0$, there is a vertex $x\in N$ such that $A'=\{(u_1,x),(u_2,x)\}$ is a solution, and OPT(S)=2=OPT(M)+1. ♦

Proposition 5.2. If q=3 then there are distinct edges e_1,e_2 such that $A'=\{e_1,e_2\}$ is a solution, and OPT(S)=OPT(M)=2.

Proposition 5.3. Suppose that $q\geq 4$, and let $J=\{1,2,3,4\}$, $V_J=\{v_1,v_2,v_3,v_4\}$ and $L_J=\{L(v_j)|\ j\in J\}$.

(1) If V_J consists of a D-combination then V_J can be partitioned into a D-combination $\{v'_1,v'_2\},\{v'_3,v'_4\}$ such that $G'=G+\{(u'_1,u'_2),(u'_3,u'_4)\}$ is a simple graph having a 4-component S that contains every $L(v_j)\in L_J$, where $u'_j\in L(v'_j)$, $j\in J$.

(2) If V_J does not consist of a D-combination then there are three distinct edges e_1,e_2,e_3, with $\{u_1,u_2,u_3,u_4\}\subseteq V(\{e_1,e_2,e_3\})$, such that $G'=G+\{e_1,e_2,e_3\}$ is a simple graph having a 4-component S that contains every $L(v_j)\in L_J$, where $u_j\in L(v_j)$, $j\in J$. ♦

Corolary 5.1. If q=4 then the following (1) or (2) holds.

(1) If V_J consists of a D-combination then OPT(S)=OPT(M)=2.

(2) If V_J does not consist of a D-combination then OPT(S)=3=OPT(M)+1. ♦

Based on these results, we propose an $O(|N|^2)$ algorithm for solving UW-4ECA(S,SA).

Algorithm GS;

/* Input: a simple graph G=(N,A) with ec(G)=3 */
/* Output: a solution A' */
begin
1. Compute all 4-components of G; Constructs S(G); Find all leaves of S(G);
2. Construct R(G)=(V',E'); $V''\leftarrow V'$;
3. $q\leftarrow 1$; $\Delta\leftarrow\emptyset$; $A'\leftarrow\emptyset$; $A''\leftarrow\emptyset$;
4. **if** $|V''|\geq 7$ **then**
 begin
  if q=1 **then** choose a nonadjacent pair $D_1=\{v_1,v_2\}\subseteq V''$;
  goto Step 9
 end;
5. **if** $2\leq|V''|\leq 6$ **then** /* V'=V" */
  find a maximum matching M of R(G)
 else /* $V''=\emptyset$ */ **goto** Step 14;
6. **if** $|M|\geq 2$ **then**
 begin
  choose a nonadjacent pair $\{v_3,v_4\}\subseteq V''-D_1$;
  $D_2\leftarrow\{v_3,v_4\}$; $q\leftarrow q+1$; **goto** Step 10
 end;
7. **if** |M|=1 **then**
 begin /* $2\leq|V''|\leq 4$ by Propositions 3.3, 3.5 and 3.6, and $v_1\bar{\chi}v_2$ */
  if $|V''|=2$ and $V''=\{v_1,v_2\}$ **then** $A''\leftarrow\{(u_1,u_2)\}$;
  if $|V''|=3$, $V''=\{v_1,v_2,v_3\}$ and $D_1=\{v_1,v_2\}$ **then**
   if $v_3\bar{\chi}v_j$ for some $v_j\in\{v_1,v_2\}$ **then**
    $A''\leftarrow\{(u_1,u_2),(u_j,u_3)\}$
   else /* $L(v_i)=\{u_i\}$, i=1,2,3 */
    begin
     choose $u\in N-\{u_1,u_2,u_3\}$ that is not adjacent to u_3 in G;
     $A''\leftarrow\{(u_1,u_2),(u,u_3)\}$
    end;
  if $|V''|=4$, $V''=\{v_1,v_2,v_3,v_4\}$ and $D_1=\{v_1,v_2\}$ **then**
  /* use Corollary 5.1(2) and OPT(S)=OPT(M)+1 */
   begin
    choose three distinct edges $e_1=(u_1,u_2),e_2,e_3$, with $\{u_1,u_2,u_3,u_4\}\subseteq V(\{e_1,e_2,e_3\})$, such that $G'=G+\{e_1,e_2,e_3\}$ is a simple graph having a 4-component S with $L(v_j)\subseteq S$, j=1,2,3,4;
    $A''\leftarrow\{e_1,e_2,e_3\}$
   end;
  $V''\leftarrow\emptyset$; **goto** Step 12
 end;
8. **If** |M|=0 **then**
  /* $|V''|=2$ by Propositions 3.3, 3.5 and 3.6, and $L(v_i)=\{u_i\}$, i=1,2, and OPT(S)=OPT(M)+1 */
 begin
  choose a vertex $u\in N-\{u_1,u_2\}$ such that u is not adjacent to u_i, i=1,2, in G;
  $A''\leftarrow\{(u,u_1),(u,u_2)\}$; $V''\leftarrow\emptyset$; **goto** Step 12
 end;
9. Choose a nonadjacent pair $D_2=\{v_3,v_4\}\subseteq V''-D_1$; $q\leftarrow q+1$;
 /* edge-interchange */
10. Find a D-combination $\{v'_1,v'_2\},\{v'_3,v'_4\}$, whose union is $D_1\cup D_2$, such that two edges $f_1=(u'_1,u'_2)$, $f_2=(u'_3,u'_4)$ with $u'_j\in L(v'_j)$, j=1,2,3,4, satisfies that $G+\{f_1,f_2\}$ is a simple graph having a 4-component S with $L(v'_i)\subseteq S$, i=1,2,3,4.
11. $A''\leftarrow\{f_2=(u'_3,u'_4)\}$; $\Delta\leftarrow\{v'_3,v'_4\}$; $V''\leftarrow V''-\Delta$; $D_1\leftarrow\{v'_1,v'_2\}$;
12. **if** $|V''|>0$ **then** $R(G)\leftarrow R(G)-\Delta$;
13. $A'\leftarrow A'\cup A''$; **goto** Step 4;
14. **halt;**
 end.

We show the correctness of the algorithm and its time complexity.

Theorem 5.1. The algorithm GS correctly finds a solution A' to UW-4ECA(S,SA) for any given G in $O(|N|^2)$ time.

Proof. First we prove the correctness. The algorithm eventually termintes, existence of $\{v'_3,v'_4\}$ in Step 6 is guaranteed by Proposition 2.5, and Step 12 maintains R(G) by Proposition 4.5. Construction of A" and Propositions 4.3 and 4.4 guaratee that if either $|V''|\geq 7$ or $2\leq|V''|\leq 6$ with $|M|\geq 2$ holds then the set A' at Step 13 of the algorithm satisfies that G+A' is a simple graph having a 4-component S with $L(v_i)\subseteq S$, $\forall v_i\in V'-V''$. Hence it suffices to consider the case with $2\leq|V''|\leq 6$ and $|M|\leq 1$. Propositions 3.3, 3,5 and 3.6 show that if |M|=1 then

$2 \le |V''| \le 4$ and if $|M|=0$ then $|V''|=2$. We use Proposition 5.1(1) if $|M|=1$ and $|V''|=2$, and Proposition 5.2 if $|M|=1$ and $|V''|=3$, Corollary 5.1(2) if $|M|=1$ and $|V''|=4$, and Proposition 5.1(2) if $|M|=0$ and $|V''|=2$. Construction of A" in Steps 7 and 8 show that, for A' obtained after the termination, G+A' is a simple graph with ec(G)=4, where $|A'|=OPT(S)=OPT(M)+1$ if the case ($|M|=1$ and $|V''|=4$) or ($|M|=0$ and $|V''|=2$) occurrs, and $|A'|=OPT(S)=OPT(M)$ otherwise. Hence the algorithm is correct.

Next we show its time complexity. Step 1 can be done in $O(|N|^2)$ time by using the algorithm in [16]. Step 2 can be done in $O(|N|^2)$ time, since we have only to check in $O(|E|)$ time if v_i is not adjacent to v_j in S(G) for any pair $v_i, v_j \in V'$ with $L(v_i)=\{v_i\}$ and $L(v_j)=\{v_j\}$. $O(1)$ time is spent in Steps 3,4,6,9,11,13. Step 5 can be done in $O(|V''|^{2.5})$ time by using the algorithm in [13], and this is $O(1)$ time since $|V''| \le 6$. Step 10 checks whether $M(u'_1, u'_2; G+\{f_1, f_2\}) \ge 4$ or not. Remark 4.1 shows that Step 10 can be done in O(|N|) time. Step 12 can be done in O(|N|) time. Steps 4 through 13 are repeated |A'| times, where $|A'| \le \lceil |V'|/2 \rceil + 1$. Hence total time complexity is $O(|N|^2)$. Q.E.D.

6. Concluding Remarks

An $O(|N|^2)$ algorithm for solving UW-4ECA(S,SA) is proposed. The result can be used in solving UW-4ECA(*,SA), where handling of multiplicity of G that may have $2 \le |N| \le 4$ is required. We have also presented cardinality of a maximum matching of a leaf graph, and the results may be interesting from viewpoint of combinatorial theory. The paper is a first step toward an open problem UW-kECA(*,SA), which is left for future research.

Acknowledgements

The third author, T. Watanabe, would like to thank Prof. H. Nagamochi of Kyoto University, Kyoto, Japan, for his suggestion concerning UW-kECA(*,SA). The research of T. Watanabe was partly supported by the Telecommunications Advancement Foundation (TAF), Tokyo, Japan; and is partly supported by The Okawa Institute of Information and Telecommunication, Tokyo, Japan.

References

[1] A. V. Aho, J. E. Hopcroft and J. D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, MA, 1974.

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

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

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

[5] H. Frank and W. Chou, Connectivity considerations in the design of survivable networks, IEEE Trans. Circuit Theory, CT-17, 486-490, 1970.

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

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

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

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

[10] T. C. Hu, Integer Programming and Network Flows, (Addison-Wesley, Reading, Mass., 1969).

[11] 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, pp.8-12 (March-April, 1986).

[12] S.Khuller and R.Thurimella, Approximation Algorithms for Graph Augmentation, Proc. 19th Internatioanl Colloquium on Automata, Languages and Programming (Lecture Notes in Computer Sciece 623, Springer-Verlag) (July 1992), pp.330-341.

[13] S.Micali and V. V. Vazirani, An $O(|V|^{1/2}|E|)$ algorithm for finding maximum matching in general graphs, Proc.21st IEEE Syposium on Foundations of Computer Science, pp.17-27,1986.

[14] H.Nagamochi and T.Ibaraki, A linear time algorithm for finding a sparse k-connected spanning subgraph of a k-connected graph, Algorithmica., Vol.7(1992), pp.583-596.

[15] 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), pp.163-180.

[16] H. Nagamochi and T.Watanabe, Computing k-Edge-Connected Components of a Multigraphs, Trans.IEICE of Japan, Vol.E76-A, No.4 (1993), pp.513-517.

[17] 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, pp.698-707 (1990).

[18] 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, pp.410-424 (1991).

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

[20] W.T.Tutte, The factorization of linear graphs,J ournal of the London Mathematical Society, No.22, pp.107-111, 1947.

[21] S. Ueno, Y. Kajitani and H. Wada, Minimum augmentation of a tree to a k-edge-connected graph, Networks, 18(1988), 19-25.

[22] T. Watanabe, An efficient way for edge-connectivity augmentation, Technical Report No. ACT-76-UILU-ENG-87-2221, Coordinated Science Lab., University of Illinois at Urbana, Urbana, IL 61801, April, 1987. Also presented at Eighteenth Southeastern International Conference on Combinatorics, Graph Theory, Computing, No.15, Boca Raton, FL, U.S.A., February 1990.

[23] T. Watanabe, A simple improvement on edge-connectivity augmentation, IEICE(Japan) Technical Reports, CAS87-203, 43-48, 1987.

[24] T. Watanabe, Graph Augmentation Problems, Trans. IEICE of Japan, Vol.E74, No.4, pp.632-643, 1991.

[25] T.Watanabe,T.Mashima, and S.Taoka, The k-edge-connectivity augmentation problem of weighted graphs, Lecture Notes in Computer Science 650, Algorithms and Computation (T. Ibaraki, Y. Inagaki, K. Iwama, T. Nishizeki and M. Yamashita(Eds.)), Springer-Verlag, Berlin, pp.31-40, December 1992.

[26] T.Watanabe, T.Mashima and S.Taoka, Approximation algorithms for minimum-cost augmentation to k-edge-connect a multigraph, Proc. 1993 IEEE International Symposium on Circuits and Systems (May 1993), pp.2556-2559.

[27] T.Watanabe and A.Nakamura, Edge-connectivity augmentation problems, Journal of Computer and System Sciences, Vol.35, No.1,pp.96-144 (1987).

[28] T. Watanabe and A. Nakamura, 3-Connectivity augmentation problems, Proc. of 1988 IEEE International Symposium on Circuits and Systems(1988), 1847-1850.

[29] T.Watanabe and A.Nakamura, A smallest 3-connectivity augmentation of a graph, Discrete Applied Mathematics Vol.28, 183-186, 1990.

[30] T. Watanabe and A. Nakamura, A Minimum 3-Connectivity

Augmentation of a Graph, Journal of Comput. System Sci., Vol.46, No.1, pp.91-128, 1993.
[31] T.Watanabe, T.Narita and A.Nakamura, 3-Edge-connectivity augmentation problems, Proc. 1989 IEEE International Symposium on Circuits and systems, pp.335-338, 1989.
[32] 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, pp.424-429, Dec. 1992.
[33] T.Watanabe, S.Taoka and T.Mashima, Minimum-cost augmentation to 3-edge-connect all specified vertices in a graph, Proc. 1993 IEEE International Symposium on Circuits and systems (May 1993), pp. 2311-2314.
[34] T.Watanbe and M.Yamakado, A linear time algorithm for smallest augmentation to 3-edge-connect a graph, Trans. IEICE of Japan, Vol.E76-A, No.4 (1993), pp.518-531.

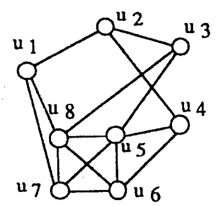

Fig. 3.1. A simple graph G with ec(G)=3 and |V'|=4. {(u_1,u_3),(u_2,u_4)} is a solution to UW-4ECA(*,MA), while {(u_1,u_6),(u_2,u_5),(u_3,u_4)} is a solution to UW-4ECA(S,SA) and OPT(S)=3=OPT(M)+1.

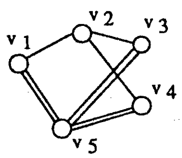

Fig. 3.2. The graph S(G) of G in Fig. .3.1.

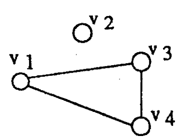

Fig. 3.3. The leaf-graph R(G) of G in Fig. 3.1.

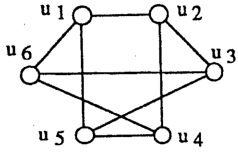

Fig. 3.4. Another graph G with ec(G)=3 and |V'|=6. {(u_1,u_4),(u_2,u_5),(u_3,u_6)} is a solution to UW-4ECA(*,MA), while {(u_1,u_4),(u_2,u_5),(u_3,u_4),(u_5,u_6)} is a solution to UW-4ECA(S,SA) and OPT(S)=4 =OPT(M)+1. In this case S(G)=G.

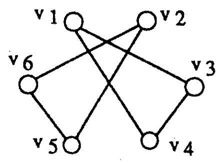

Fig. 3.5. The leaf-graph R(G) of G in Fig. 3.4.

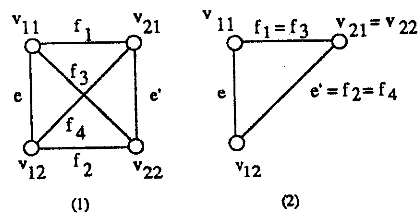

Fig. 4.1. The edges e, e' and f_i, 1≤i≤4. (1) v_{21}≠v_{22}; (2) v_{21}=v_{22}.

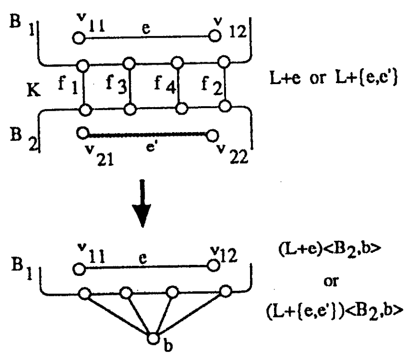

Fig. 4.2. The graph (L+e)<B_2,b> or (L+{e,e'})<B_2,b> constructed from L+e or L+{e,e'}, respectively.

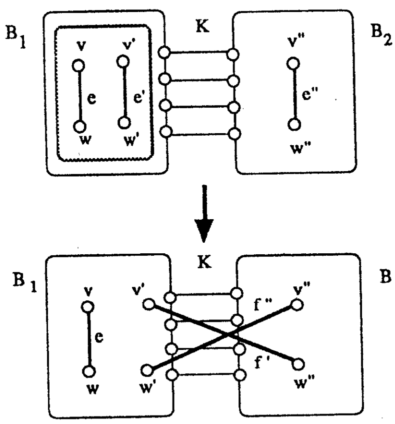

Fig. 4.3. A situation for edges e, e', e", f' and f" in the case where f'=(v',w") and f"=(v",w').