On Mixed Connectivity Certificates

(Extended Abstract)

Shimon Even[1*], Gene Itkis[1**], Sergio Rajsbaum[2***]

[1] Department of Computer Science, Technion, Haifa, Israel 32000
[2] Instituto de Matemáticas, U.N.A.M., Ciudad Universitaria, D.F. 04510, México

Abstract. Vertex and edge connectivity are special cases of mixed connectivity, in which all edges and a specified set of vertices play a similar role. Certificates of k-connectivity for a graph are obtained by removing a subset of its edges, while preserving its connectivity up to k.
We unify the previous work on connectivity certificates and extend it to handle mixed connectivity and multigraphs. Our treatment contributes a new insight of the pertinent structures, yielding more general results and simpler proofs. Also, we present a communication-optimal distributed algorithm for finding mixed connectivity certificates.

1 Introduction

1.1 Basic concepts

Let $G=(V,E)$ be a finite undirected graph with no self-loops, and $x,y\in V$ be a pair of distinct vertices of G. The *edge connectivity* of x and y in G is the maximum number of edge-disjoint paths connecting x and y. Similarly, their *vertex connectivity* is the maximum number of vertex-disjoint paths connecting x and y.

[*] Part of this research was conducted while visiting the Instituto de Matemáticas, U.N.A.M. Supported by the Fund for the Promotion of Research at the Technion and DGAPA Projects, U.N.A.M. E-mail: even@cs.technion.ac.il

[**] Supported by the Israeli Council for Higher Education and NSF grant CCR-90 15276. Part of this research was conducted in Boston University. E-mail: itkis@cs.technion.ac.il

[***] Part of this research was conducted while visiting the MIT Laboratory for Computer Science, and CRL Digital Equipment Corporation. Partly supported by DGAPA Projects, U.N.A.M. E-mail: rajsbaum@theory.lcs.mit.edu

Following [FIN93], we consider a generalization of these two particular types of connectivity. Let $S \subseteq V$. We say that a family of paths connecting vertices x, y is *S-independent* if the paths are edge-disjoint and every element of S appears as an inner vertex in at most one of these paths. The *S-mixed connectivity* $\lambda_S(x, y; G)$ of x and y in G is the maximum number of S-independent paths connecting x and y in G. The cases of $S = \emptyset$ and $S = V$ correspond to edge and vertex connectivity, respectively. For brevity, if G is clear from the context, we omit it. Say that x and y are *S-mixed k-connected* if $\lambda_S(x, y) \geq k$. This is also referred to as *local* connectivity, as opposed to global: the *global connectivity* of graph G is $\lambda_S(G) \overset{\text{def}}{=} \min_{x,y \in V} \lambda_S(x, y; G)$. G is *S-mixed k-connected* if $\lambda_S(G) \geq k$.

For each type of connectivity, a certificate of k-connectivity for G is a subgraph preserving the connectivity up to k. Namely, $G' = (V, E')$, $E' \subseteq E$, is a *certificate of local S-mixed k-connectivity* for G if for any two vertices x and y, $\lambda_S(x, y; G') \geq \min\{k, \lambda_S(x, y; G)\}$. Similarly, G' is a *certificate of global S-mixed k-connectivity* for G if $\lambda_S(G') \geq \min\{k, \lambda_S(G)\}$. The *size* of G' is $|E'|$.

Clearly, certificates of local k-connectivity are also certificates of global k-connectivity; the opposite is generally not true. Unless stated otherwise, we will speak about certificates of local connectivity.

For a k-connected G there is a trivial lower bound of $k|V|/2$ on the size of a certificate of k-connectivity, because the degree of every vertex in a k-connected graph is at least k. Results of Mader ([M72], [M73], [M79]) imply that every edge k-connected graph G contains an edge k-connected subgraph with $O(k|V|)$ edges. Also, Mader's results imply a similar result for vertex connectivity of simple graphs. Namely, an edge k-connected graph has a certificate of global edge k-connectivity of size $O(k|V|)$, and a simple graph, which is vertex k-connected, has a certificate of global vertex k-connectivity of size $O(k|V|)$. However, as is shown below, if one following Zykov's point of view [Z69], as we do, if parallel edges are allowed, the statement does not hold for vertex connectivity.

Example. Let $G = (V, E)$ be a complete graph, each pair of vertices connected by p parallel edges. Then $|E| = p|V|(|V| - 1)/2$ and $\lambda_V(G) = p + |V| - 2$ (since any two vertices have $p + |V| - 2$ vertex-disjoint paths between them). At the same time, the removal of any edge reduces the graph's global connectivity. In this example, for $p \geq |V|$, Mader's $O(p|V|)$ bound is off by a factor of $|V|$.

1.2 Applications

Certificates with fewer edges than the original graph are useful for improving the efficiency of a number of graph algorithms. One may perform a preprocessing step to find a sparse certificate, and then run the algorithms on the certificate. For example, this method has been used to improve the sequential time complexity of testing simple undirected graphs for k-connectivity [NI92, CKT93, G91]; to improve the running time for finding three independent spanning trees [CM88]; to design efficient fault tolerant protocols for distributed computer networks [IR88], dynamic and distributed algorithms [EGIN92].

Sparse certificates are of special utility for the *distributed model* of computation. This model consists of a graph G with the vertices representing processors and the edges representing bidirectional communication links. There is no common memory and all communication is through messages sent along the links. The messages take a finite but arbitrary time to traverse a link. The *communication complexity* of the algorithm is the number of messages sent by the algorithm, assuming each message contains $O(\log |E|)$ bits. For the sake of discussing the *time-complexity* of asynchronous distributed algorithms, it is customary to assume that each message is transmitted and processed within one unit of time.

The vertex (edge) connectivity of the graph is related to the number of processors (links) failures that can be tolerated by the distributed system before the network is disconnected. S-mixed connectivity allows to deal with networks where any link can fail but only processors in S are subject to failure. The number of messages sent by the distributed algorithm often depends critically on $|E|$. In such cases, if the algorithm is executed on a sparse certificate then the message complexity is reduced while preserving the number of faults that can be tolerated.

1.3 Previous work

Given a k-(vertex or edge) connected graph, the problem of finding a k-connected spanning subgraph with the minimum number of edges is NP-hard for any fixed $k \geq 2$; ([GJ79] cites personal communication with F. Chung and R. Graham). Indeed, for $k=2$ the reduction from the Hamiltonian Cycle problem is trivial.

However, finding good approximations is possible for all k: Nagamochi and Ibaraki [NI92] find, in time $O(|E|)$, edge and vertex connectivity certificates of size $< k|V|$ (which is within a factor of 2 from the trivial lower bound described in

Section 1.1). Their algorithm, as well as others, consists of finding a sequence of forests $F_1, F_2 \ldots$ in the graph. Each forest F_j is maximal in the remaining graph $G - \cup_{i=1}^{j-1} F_i$ (e.g. each connected component of the remaining graph is spanned by a tree of F_j). The maximality alone suffices to show that $G_k \overset{\mathrm{def}}{=} \cup_{i=1}^{k} F_i$ is an edge k-connectivity certificate of size $< k|V|$. Moreover, if G is simple and the forests are grown according to a certain rule (see Section 4), then G_k is a vertex k-connectivity certificate as well.

Graph G is called *S-simple* if it has no parallel edges incident to vertices of S. Frank, Ibaraki and Nagamochi [FIN93] show that for all $S \subseteq V$ the algorithm of [NI92] applied to S-simple graphs produces certificates of S-mixed connectivity.

Cheriyan, Kao and Thurimella [CKT93] introduce a more flexible way of constructing certificates of vertex connectivity of size $< k|V|$, and use it in their distributed and parallel algorithms. They show that for simple graphs, constructing F_i in a scan-first-search manner (see Section 4.3) is sufficient to produce certificates of vertex k-connectivity. Their distributed algorithm uses the synchronizers of [AP90] and runs in time $O(k|V|\log^3|V|)$ using $O(k|E| + k|V|\log^3|V|)$ messages. Every certificate obtained by the algorithm of [NI92] can be obtained using the scan-first-search; but as we show in the sequel, the converse does not hold. Thus, the results of [FIN93] do not apply to the certificates of [CKT93].

A new distributed algorithm by Thurimella [T95] for computing $O(k|V|)$ size certificates has time complexity $O(k(D + |V|^{0.614}))$, where D is the diameter of the network. The communication complexity of the algorithm was not analyzed.

In all previous results quoted above, the graphs are assumed to be S-simple; $S = \emptyset$ and $S = V$ for edge and vertex connectivity, respectively.

1.4 Our results

We present a general scheme for generating S-mixed k-connectivity certificates. It consists of an optimum reduction from general graphs, allowing parallel edges, to S-simple graphs, and a scheme to generate certificates of size $< k|V|$ for S-simple graphs.[3] This scheme includes as special cases the results of [CKT93, NI92, FIN93] and has a simpler correctness proof. We believe that the generality of this scheme as well as the simplicity of the proof contribute towards a better understanding of connectivity certificates.

[3] Note that an optimum reduction may not produce an optimum certificate, even if one is given an optimum certificate for the S-simple graph.

For the distributive model, we present a communication-optimal algorithm which is an implementation of the scheme above. For simple graphs it generates certificates of size $< k|V|$ in time $(2k+2)|V|$ using $\leq 4|E|$ messages. In addition to the improved complexity, our algorithm is simpler and works in a more restricted single server model. A *single server* algorithm has the following property. At any time there is exactly one vertex which is active, and all activities in the network, while this vertex is in charge, are restricted to its immediate neighborhood. In the course of the computation the server travels in the network along its edges.

In this extended abstract, some of the proofs are omitted.

2 Notation and Basic Notions

Let $V(E')$ be the set of vertices which are the endpoints of the edges of $E' \subseteq E$. For disjoint sets $X, Y \subseteq V$, let $E(X,Y) \subseteq E$ denote the set of edges of G with one endpoint in X and the other in Y; $E(X)$ is the set of edges with an endpoint in X. We write $E(x,Y)$ and $E(x)$ if $X = \{x\}$. A path *linking* sets X and Y is a path with one endpoint in X and the other endpoint in Y; X and Y are then said to be *linked.* $\Gamma_G(v) \overset{\text{def}}{=} V(E(v)) - \{v\}$ is the set of neighbors of v in G.

Following the definitions in [FIN93], let $\{Z, A, B\}$ be a partition of V such that $Z \subseteq S$, $A \neq \emptyset$ and $B \neq \emptyset$. We say that the pair $C = (Z, E(A,B))$ is an *S-mixed cut.* If $\emptyset \neq A' \subseteq A$, $\emptyset \neq B' \subseteq B$ then the cut $C = (Z, E(A,B))$ *separates* A' and B'. The *size* of C is defined to be $|C| \overset{\text{def}}{=} |Z| + |E(A,B)|$. As was observed in [FIN93], following [DF56], using the max-flow min-cut theorem, it is easy to derive the following version of Menger's theorem.

Theorem 1 *For any $a, b \in V$, the minimum size of an S-mixed cut separating a, b is $\lambda_S(a,b;G)$.*

Henceforth, unless otherwise specified, we discuss S-mixed cuts and connectivity. So, for brevity, we omit the "S-mixed".

3 Reduction to S-simple graphs

$\widetilde{G}$ is the *S-simplification* of G, $\widetilde{G} = simple_S(G)$, if $\widetilde{G}$ is the maximal S-simple subgraph of G. To obtain $\widetilde{G}$ from G, for each $\{a,b\} \cap S \neq \emptyset$ replace all parallel edges between a and b, if there are any in G, by a single edge a—b in $\widetilde{G}$.

We will reduce the construction of a global connectivity certificate for a general graph G (with arbitrary parallel edges) to finding a local connectivity certificate for $simple_S(G)$. The following lemma will be useful:

Lemma 1 *Let G' be a certificate of local k-connectivity for G. Then any cut C' in G' is either a cut in G or $|C'| \geq k$.*

(Proof omitted.)

3.1 Global connectivity

Define the *S-degree of a vertex v*, $d_S(v; G)$, to be $|V|-2+\min_{w\neq v}|E(v,w)|$ if $\Gamma_G(v) = V-\{v\} \subseteq S$ and $|E(v, V-S)| + |\Gamma_G(v) \cap S|$ otherwise. The *S-degree of graph G* is $d_S(G) \stackrel{\text{def}}{=} \min_{v\in V}\{d_S(v; G)\}$. The traditional definitions of degree coincide with $d_\emptyset$. It is easy to show that $d_S(G) \geq \lambda_S(G)$.

Let $m = \min\{k, d_S(G)\}$, and $G' = (V, E' \subseteq E)$. The following procedure *Incr_Deg*(G', G, m) increases $d_S(G')$ to be $\geq m$ by adding edges of G to G':

```
Procedure Incr_Deg(G', G, m):
1. for every $v \in V$, starting with those $\in V-S$, do
2.   while $d_S(v; G') < m$ do
3.     if $\Gamma_G(v)-\Gamma_{G'}(v) \neq \emptyset$ then add some $e \in E(v, \Gamma_G(v)-\Gamma_{G'}(v))$ to $E'$
4.     else if $\Gamma_{G'}(v) = V-\{v\} \subseteq S$ then
5.       for every $u \in V-\{v\}$ such that $|E'(v,u)| = \min_{w\neq v}|E'(v,w)|$
6.         add some $e \in E(v,u)-E'$ to $E'$
7.     else add some $e \in E(v, V-S)-E'$ to $E'$
```

Lemma 2 *After executing Incr_Deg(G', G, m), $d_S(G') \geq m$.*

(Proof omitted.)

Reduction: To obtain a certificate of global S-mixed k-connectivity for G first obtain a certificate G' of local S-mixed m-connectivity for $simple_S(G)$, and then apply *Incr_Deg*(G', G, m) to turn G' into the desired certificate.

First, $\lambda_S(G) \leq d_S(G)$ implies $k \geq m \geq \min\{k, \lambda_S(G)\}$, so there is no difference between certificates for global k- and m-connectivity. Now, the following theorem, together with Lemma 2, implies the correctness of the above reduction.

Theorem 2 *Let $G' = (V, E' \subseteq E)$, $d_S(G') \geq m$, and let G' contain as a subgraph a certificate of local m-connectivity for $simple_S(G)$. Then $\lambda_S(G') \geq \min\{m, \lambda_S(G)\}$.*

(Proof omitted.)

If $\lambda_S(G') \geq k$ then $d_S(G') \geq k$. So, at least for k-connected graphs, it is necessary to increase the S-degree of the simplification's certificate (as done by *Incr_Deg*), in order to turn it into a certificate for G. As we showed above, surprisingly, it turns out to be sufficient as well.

In general, $Incr\_Deg(G', G, m)$ adds minimal but not necessarily minimum number of edges to achieve $d_S(G') \geq m$. But when applied to an m-connectivity certificate of $simple_S(G)$, as in our particular reduction, this number is indeed minimum (i.e. adding any smaller set of edges from G to G' than that added by $Incr\_Deg(G', G, m)$, leaves $d_S(G') < m$). This implies the reduction is optimal.

Lemma 3 (Reduction Optimality) *Let $H = (V, E_H)$ be a certificate of local m-connectivity for $simple_S(G)$, and let $G' = (V, E')$ be obtained from H by applying $Incr\_Deg(H, G, m)$, and $G'' = (V, E'')$ be an arbitrary m-connectivity certificate for G such that G'' contains H as a subgraph. Then, $|E'| \leq |E''|$.*

(Proof omitted.)

Incr_Deg can be executed at each vertex almost independently (it should be executed at the vertices of $V - S$ before the vertices of S). So, say, in the distributed model, it can be implemented to work in 3 steps. For the sequential model its time complexity is $O(|V| + |E|)$, and it can even be adjusted to work in $O(|V| + |\widetilde{E}|)$, where $(V, \widetilde{E}) = simple_S(G)$.

Finally, we observe that it is natural that even for the global connectivity the reduction is to the local one. Indeed, the simplification of a k-connected G may happen to be just 1-connected. Then any spanning tree of the simplification is a satisfactory certificate for it. In fact, it is possible to show that (similarly to the reduction above) given a certificate G' of global k-connectivity for G, a certificate of global k-connectivity for $simple_S(G)$ can be obtained by adding edges from $simple_S(G)$ to $simple_S(G')$ to increase its degree to $d_\emptyset(simple_S(G')) \geq \min\{k, d_\emptyset(simple_S(G))\}$.

3.2 Local connectivity

In general the reduction of sec. 3.1 may not produce certificates of *local* connectivity. However, in some specific cases, obtaining certificates of local connectivity is easy. If $\{x,y\}\cap S=\emptyset$ then $\lambda_S(x,y;G)=\lambda_S(x,y;simple_S(G))$. Therefore, if a certificate of local connectivity between vertices of $V-S$ is required, then a certificate of local connectivity of $simple_S(G)$ can be used. A connectivity certificate for a specific pair s,t can be constructed by defining $S'=S-\{s,t\}$ and obtaining a certificate of local connectivity for $simple_{S'}(G)$.

For general graphs, a certificate of local connectivity for G can be constructed as follows. First, find a certificate of local connectivity for $simple_S(G)$. Next, flesh out each edge $x\overset{e}{-}y$ of the certificate to have $\max\{k,|E(x,y)|\}$ parallel edges between x and y. This could increase the certificate size, unnecessarily, by factor of $k-1$ above the minimum. The problem of efficiently reducing the task of finding sparse certificates of local connectivity to finding sparse certificates for S-simple graphs remains open.

4 Certificates for S-simple graphs

Let $G=(V,E)$ be an S-simple graph. Let $\{F_i\}$ be a sequence of mutually disjoint non-empty sets of edges partitioning E, and define $E_k \overset{\text{def}}{=} \bigcup_{1\le i\le k} F_i$, $\overline{E_k} \overset{\text{def}}{=} E-E_k$, $G_k \overset{\text{def}}{=} (V,E_k)$. Note that $\overline{E_0}=E$. For every $i\geq 1$, let F_i be a maximal forest in $(V,\overline{E_{i-1}})$. Then each forest F_i consists of a set of spanning trees, one for each connected component of $(V,\overline{E_{i-1}})$. The next lemma follows directly from the maximality of the forests [NI92].

Lemma 4 *If u and v are connected in F_j, then for each i, $1\le i<j$, u and v are connected in F_i.*

Lemma 4 is sufficient to prove that G_k is a certificate of local edge (i.e. $\emptyset$-mixed) k-connectivity of size $<k|V|$. We skip the proof since it is a special case of the mixed connectivity results which follow. If $S\neq\emptyset$, G_k may not be a certificate of k-connectivity, even in the global sense.

Next, we define S-greedy forests (which are also maximal) and show that the S-greedy forests yield certificates of local S-mixed connectivity.

4.1 Greedy Forests

The following search procedure produces a maximal forest F of G. Initially $F = \emptyset$. The vertices of G will be visited as specified shortly. When a vertex is visited some of its incident edges may be added to F. The first vertex to be visited can be chosen arbitrarily. Whenever the visitation of a vertex terminates, the next vertex to be visited can be chosen to be any other vertex of $V(F)$, or any vertex of a component of G which has no vertices in $V(F)$. The process ends when all vertices of G have been visited at least once. Upon termination F is a maximal forest of G. The edges are added to F (one at a time) as follows. While visiting a vertex $v \in S$, for *every* neighbor x of v, $x \notin V(F)$, add the edge v—$x \in E$ to F. S-simplicity implies that there is only one edge joining v and x. When visiting $v \notin S$, if $v \overset{e}{—} x \in E$ and if $x \notin V(F)$, one is allowed to add e to F. (Clearly, if e is added to F then none of its parallel edges, if any, may be added to F neither in the same nor in any other visitation.) So, no edges incident to $v \in S$ may be added after its first visitation. If $v \notin S$, edges incident to v may be added during several of its visitations.

The forests produced by the above procedure are called S-*greedy*. Next, we define them without referring to any algorithm (as a static counterpart to the above algorithmic construction). Let F be a maximal forest in G, and let $t : V \to \{1, 2, \ldots, |V|\}$ be a 1-1 *numbering* of the vertices. The numbering t induces orientation on edges: $\vec{F}(t) \overset{\mathsf{def}}{=} \{u \to v : u—v \in F \wedge t(u) < t(v)\}$. If $\vec{T}$ is a tree rooted at r, directed from the root towards the leaves, then for each $v \neq r$, $\mathsf{parent}(v)$ is the unique u such that $u \to v \in \vec{T}$; also $\mathsf{parent}(r) \overset{\mathsf{def}}{=} r$.

Definition 1 *Maximal F is S-*greedy *in G if there exists a numbering t of the vertices such that*

(1) For every tree T of F, $\vec{T}(t)$ is a rooted tree.

(2) If $w—v \in E$ and $w \in S$ then $t(\mathsf{parent}(v)) \leq t(w)$.

Intuitively, Definition 1 reflects the above algorithmic construction of greedy forests as follows. The order in which the vertices of G are visited for the first time is specified by t. If $\mathsf{parent}(v) = u \neq v$, then v has been added to $V(F)$ when an edge $u—v$ has been added to F, while visiting u. If $\mathsf{parent}(v) = v$ then v is the first vertex in its component to be visited. The second item in the above definition reflects the requirement that on the first visitation of $v \in S$ all its non-forest neighbors join the forest. Obviously, an S-greedy forest F in G is also S'-greedy for all $S' \subseteq S$.

4.2 Certificates

Next, we show that if for each $i \leq k$ the forest F_i is S-greedy in $(V, \overline{E}_{i-1})$, then G_k is a k-connectivity certificate (note: $|E_k| < k|V|$).

Theorem 3 below is a generalization of the main theorem of [FIN93], where it has been stated for a specific subclass of S–greedy forests.

Theorem 3 (Mixed Connectivity Certificate)
$\lambda_S(a,b;G_k) \geq \min\{k, \lambda_S(a,b;G)\}$, for all $a,b \in V$.

We need a couple of lemmas before proving the theorem. Let $C = (Z, E(A,B))$ be a cut of G. We use the following obvious fact:

Fact 1 *Let $v \in Z$, $E(v,B) = \emptyset$, $C' = (Z - \{v\}, E(A \cup \{v\}, B))$. Then $|C'| < |C|$.*

Say, a cut $C' = (Z', E'(A', B'))$ of $G' = (V, E' \subseteq E)$ *narrows* a cut $C = (Z, E(A,B))$ of G if $|C'| < |C|$ and C' separates A and B in G' ($A \subseteq A'$, $B \subseteq B'$, so $Z' \subseteq Z$). For example, C' narrows C in Fact 1 above.

Lemma 5 *Let $C = (Z, E(A,B))$ be a cut, $|C| > 0$, and forest F be S-greedy in G. There is a cut $C' = (Z', \overline{F}(A', B'))$ in $(V, \overline{F})$ which narrows C.*

Proof: If A and B are not linked in G, then a zero size cut narrows C. Thus, assume there is a path in G which connects some vertex $a \in A$ with some vertex $b \in B$. By the maximality of F, such a path exists in F as well. If $F \cap E(A,B) \neq \emptyset$ then $C' = (Z, \overline{F}(A,B))$ narrows C, since $|\overline{F}(A,B)| < |E(A,B)|$.

Now, suppose $F \cap E(A,B) = \emptyset$. Let t be a numbering of F as in Definition 1. Clearly, there is some tree $T \subseteq F$, in which both a and b appear. Thus, $V(T) \cap Z \neq \emptyset$. Let w be the least vertex (w.r.t. t) in $V(T) \cap Z$. Let r be the root of $\vec{T}(t)$; wlog assume $r \notin B$.

If $\overline{F}(w,B) = \emptyset$, then by Fact 1, there is a cut in $\overline{F}$, narrowing $(Z, \overline{F}(A,B))$. Therefore, this cut narrows C as well, and the Lemma follows.

Suppose $\overline{F}(w,B) \neq \emptyset$ and let $w \overset{e}{—} v \in \overline{F}(w,B)$. Since there is an edge in $\overline{F}$ connecting w and v, by the maximality of F, $v \in V(T)$ as well. Let $u = \mathsf{parent}(v)$

in $\vec{T}(t)$. By Definition 1 item (2), $t(u) \leq t(w)$. Let $\vec{P}$ be the directed path in $\vec{T}(t)$ from r (through u) to v. For every vertex x on $\vec{P}$, from r to u, $t(x) \leq t(u)$. $\vec{P}$ must have a vertex $z \in Z$, since $r \notin B$ and $F \cap E(A, B) = \emptyset$. By $t(z) \leq t(u) \leq t(w)$ and the minimality of $t(w)$, it follows that $z = u = w$. Thus, $w = \mathsf{parent}(v)$ in $\vec{T}(t)$. By the S-simplicity, $e \in T$, in contradiction to $w \overset{e}{—} v \in \overline{F}(w, B)$. □

Lemma 6 *Let A and B be linked in F_k and separated by a cut C in G. Then $|C| \geq k$.*

Proof: A and B are linked in F_i for all $1 \leq i \leq k$ (by Lemma 4). Use Lemma 5 to construct a sequence of $k+1$ cuts $C_j = (Z_j, \overline{E_j}(A_j, B_j))$, $0 \leq j \leq k$, with $C_k = C$ and C_{j-1} narrowing C_j. $|C_i| > |C_{i-1}| \geq 0$ for all $1 \leq i \leq k$, thus $|C| \geq k$. □

Proof *(of Theorem 3)*:
By Theorem 1 there is a cut $C = (Z, E_k(A, B))$ separating a and b in G_k, such that $\lambda_S(a, b; G_k) = |C|$. If $E_k(A, B) = E(A, B)$ then $|C| = |(Z, E(A, B))| \geq \lambda_S(a, b; G)$. Otherwise, if $E(A, B) - E_k \neq \emptyset$ then by Lemma 4, A and B are linked in F_k, and so by Lemma 6 (applied to G_k), $|C| \geq k$. In either case, $\lambda_S(a, b; G_k) = |C| \geq \min\{k, \lambda_S(a, b; G)\}$. ■

4.3 Sequential Algorithms

A naive use of the greedy search procedure (described in sec. 4.1) to construct k greedy forests one after the other, takes $O(k|E|)$ time.

When $S = V$, this algorithm is called by Cheriyan et al. *scan-first-search* [CKT93]. They prove that the union of these forests constitutes a certificate of vertex k-connectivity (of size $< k|V|$). This is a special case of our Theorem 3.

Nagamochi and Ibaraki [NI92] describe an algorithm, which we call *NI-search*. This algorithm produces a partition of E into (S-greedy) forests $F_1, F_2, \ldots$ in a single search of the graph, thus reducing the complexity to $O(|V| + |E|)$. Frank et al. show that if G is S-simple, then the resulting G_k is a certificate of local S-mixed k-connectivity [FIN93]. As we shall see, each F_i produced by NI-search is V-greedy in $(V, \overline{E}_{i-1})$, and therefore these results are subsumed by our Theorem 3.

However, there are certificates composed of S-greedy forests that cannot be produced by NI-search (e.g. see Figure 1, where $S = V$). Hence, results of

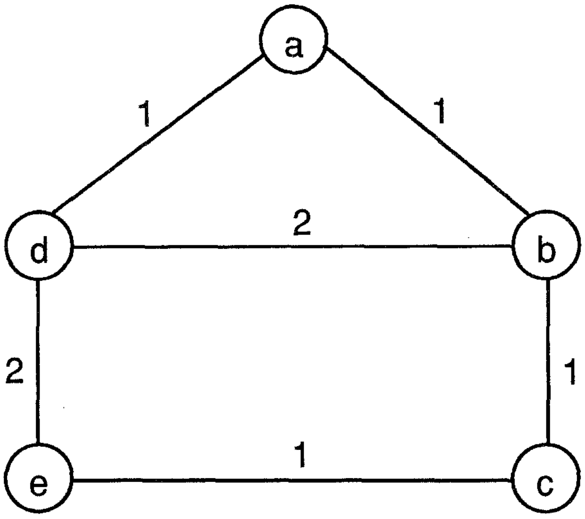

Fig. 1. *A certificate made of V-greedy forests which cannot be obtained by NI-search.*

[FIN93] do not imply that in general S-greedy forests (and in particular scan-first forests) yield mixed connectivity certificates.

NI-search is the only previously published sequential algorithm we know of, to build greedy forests in linear time. Our generic strategy, as presented in sec. 4.1, is more general and provides greater flexibility for other implementations. A case in point is the method used in [CKT93] to design efficient parallel and distributed algorithms that produce vertex connectivity certificates.

We describe NI-search, and prove that it generates S-greedy forests. This is done in detail for self-containment and since our distributed algorithm can be viewed as an implementation of NI-search.

NI-search assigns $\mathsf{rank}(e)>0$ to each edge e. $F_i \overset{\mathsf{def}}{=} \{e\in E|rank(e)=i\}$. Each vertex v keeps $\mathsf{label}(v) \overset{\mathsf{def}}{=} \max_{e\in E(v)}\{\mathsf{rank}(e)\}$. Initially, $\mathsf{rank}(e)=0$ (we say, e is *unranked*) for all e, so $\mathsf{label}(v)=0$ for all v. In each step NI-search visits a yet unvisited v with a highest $\mathsf{label}(v)$ among the unvisited vertices, and assigns $\mathsf{rank}(v\overset{e}{—}w)=\mathsf{label}(w)+1$ to each unranked $e\in E(v)$. NI-search terminates when each vertex has been visited. Notice that if $label(v)=i$ then for every $1\leq j\leq i$, v has at least one incident edge e for which $rank(e)=j$.

Lemma 7 *For each $i>0$, F_i produced by NI-search is a V-greedy forest in $(V,\overline{E_{i-1}})$.*

Proof. First we show that conditions (1) and (2) of Definition 1 hold, and then maximality (also required by the definition) is proved.

Let numbering t of vertices be defined by the order of visitation by NI-search. Let $T \subseteq F_i$ be a connected component of F_i. Let $r \in V(T)$ be the first visited in T: $t(r) = \min_{v \in V(T)}\{t(v)\}$. Obviously, the in-degree of r (in $\vec{T}$) is zero.

For any $v \in V(T)$, its in-degree (in $\vec{T}$) is at most one. Indeed, suppose $v \overset{e}{—} u \in T$, and $t(u) < t(w) < t(v)$. Then $v — w \notin T$: when w was scanned $\mathsf{label}(v)$ already was $\geq i$ (due to $\mathsf{rank}(e) = i$). Therefore, $\vec{T}(t)$ is a rooted tree and F_i is a forest. Thus condition (1) holds.

Condition (2) follows directly from the fact that all incident edges of a vertex are ranked during the first visitation of the vertex.

Finally, the maximality of F_i in $(V, \overline{E_{i-1}})$ is shown by the following sequence of three claims. Tree $T \subseteq \overline{E_{i-1}}$ is called *active* if some of its vertices are unvisited.

Claim 1. *At any time of NI-search, each F_i contains at most one active tree.*

When an edge $v \overset{e}{—} w$ is ranked i, while visiting v either $\mathsf{label}(v) \geq i$ (and the ranking of e creates no new tree in F_i) or $\mathsf{label}(v) = \mathsf{label}(w) = i-1$ (thus creating a new tree in F_i). But in the latter case, there is no unvisited $u \in V$ with $\mathsf{label}(u) \geq i$ (or u would be visited rather than v), and thus there is no other active tree in F_i. □

Claim 2. *If $u \overset{e}{—} v \in F_j$, then for each $0 < i < j$, u and v are connected in F_i.*

Just before e is ranked, $\mathsf{label}(v), \mathsf{label}(u) \geq j-1$, so $v, u \in V(F_i)$ for each $i < j$. By Claim 1, both u and v are in the unique active tree of F_i. □

Claim 3. *Each forest F_i is maximal in $(V, \overline{E}_{i-1})$.*

Let $u \overset{e}{—} v \in \overline{E_i}$. Thus, $e \in F_j$ for some $j > i$. Therefore, by Claim 2, u, v are connected in F_i. □

■

Lemma 7 and Theorem 3 yield the following:

Corollary 1 ([FIN93]) *If G is S-simple then, G_k (produced by NI-search) is a certificate of local S-mixed k-connectivity for G of size $< k|V|$.*

4.4 Distributed Algorithm

In this section we present a new distributed algorithm for finding mixed connectivity certificates of size $< k|V|$, for connected S-simple graphs. The algorithm, described in Figure 2, is executed in a network identified with a graph G: each vertex of V corresponds to a node of the network, and each edge of E corresponds to a communication link. Each node v maintains variables corresponding to the ones in NI-search of sec. 4.3: $\mathsf{label}(v)$ and $\mathsf{rank}(e)$, for each incident edge e (all initially 0). In addition, each node v has a boolean variable *first_time*(v) initially set to *true*, and a list *unvisited* initially including all edges incident to v. The algorithm is initiated by sending a VISIT message to any one node (on a *nil* edge) and terminates with the RETURN message received back (on the same *nil* edge). A node v receives no messages (i.e. v completes its work) after it sends a RETURN message on an edge of $\mathsf{rank} = 1$. When algorithm halts, $\mathsf{rank}(e) > 0$ for all e, $|E_k| < k|V|$ for any $k > 0$, and if G is S-simple then G_k is a certificate of local S-mixed k-connectivity.

The algorithm is of a restricted form. A single center of activity — we call it the *server* — travels from vertex to vertex around the network performing all the work. When the server is in a node v, messages are sent only between v and its neighbors. The messages used by the algorithm are: VISIT, RETURN, RANK_EDGE, and EDGE_RANKED(i), $1 \leq i \leq |E|$. The server is said to be in the vertex that has received VISIT, has not yet sent RETURN, and for each VISIT it has sent, a RETURN message has been received. All the unranked edges incident to a vertex are ranked when the server arrives at it for the first time, and in the same way as in the NI-search. Say a node is *visited* if it has received at least one VISIT message and has completed step (2.2.2). Thus, visited nodes have no unranked edges.

Each (non-*nil*) edge sees the following sequence of messages: RANK_EDGE, EDGE_RANKED($\cdot$), VISIT, RETURN. All of these are constant size except the EDGE_RANKED(i), which has $\lceil \log i \rceil$ bits, where $i < |E|$. Thus the total number of messages sent by the algorithm is $4|E|$. Since in each step a message is sent by the algorithm the time complexity is $O(|E|)$.

Theorem 4 *Algorithm D implements NI-search, runs in time* $4|E|$ *and sends* $4|E|$ *messages.*

(Proof omitted.)

ALGORITHM D (at node v)

```
(1) for RANK_EDGE message arriving at $v$ on edge $e$ do
    (1.1) label, rank($e$) $\leftarrow$ label+1
    (1.2) send EDGE_RANKED(rank($e$)) on edge $e$
(2) for VISIT message arriving at $v$ on edge $e$ do
    (2.1) drop $e$ from unvisited
    (2.2) If first_time then
        (2.2.1) first_time$\leftarrow$false
        (2.2.2) for all edges $e'$, s.t. rank($e'$) $= 0$ do
            (2.2.2.1) send RANK_EDGE on edge $e'$
            (2.2.2.2) wait for EDGE_RANKED($i$) to arrive on edge $e'$
            (2.2.2.3) rank($e'$) $\leftarrow i$
            (2.2.2.4) If label$<i$ then label $\leftarrow i$          /* OPTIONAL */
    (2.3) for each $e' \in$ unvisited with rank($e'$)$\geq$rank($e$) in decreasing order
        of rank($e'$) do
        (2.3.1) drop $e'$ from unvisited
        (2.3.2) send VISIT message on $e'$
        (2.3.3) wait for RETURN message on $e'$
    (2.4) send RETURN message on $e$
```

Fig. 2. Single server algorithm D for S-simple graphs.

Remarks: Line (2.2.2.4) can be omitted without affecting the algorithm correctness, and is included only to guarantee that, analogously to NI-search, if the server is at v then for any unvisited u, $\text{label}(v) \geq \text{label}(u)$.

Also, note that the search for an unvisited vertex of maximum label in the whole graph, as required in the original NI-search, is avoided.

$2|E'| - |V|$ steps of the algorithm can be saved by parallelizing step (2.2.2). It is also possible to modify the algorithm to reduce the time and number of messages to $4|E'|$ for $(V, E') = simple_V(G)$. If k-connectivity is desired only for a fixed k, then the algorithm can be modified to run in time complexity $O(k|V|)$ (and each message size is $\leq \lceil \log k \rceil$).

References

[AP90] B. Awerbuch and D. Peleg, "Network synchronization with polylogarithmic overhead", in *FOCS*, 1990, pp. 514–522.

[CM88] J. Cheriyan and S. N. Maheshwari, "Finding nonseparating induced cycles and independent spanning trees in 3-connected graphs", *J. Algorithms,* 9, 1988, pp. 507–537.

[CKT93] J. Cheriyan, M.-Y. Kao, R. Thurimella, "Scan-first search and sparse certificates: an improved parallel algorithm for k-vertex connectivity", *SIAM J. of Computing,* 22(1), 1993, pp. 157–174.

[DF56] G. B. Dantzig and D. R. Fulkerson, "On the Max-Flow Min-Cut Theorem of networks", *Linear Inequalities and Related Systems, Annals of Math. Study,* 38, Princeton University Press, 1956, pp.215–221.

[EGIN92] D. Eppstein, Z. Galil, G. F. Italiano, A. Nissenzweig, "Sparsification — a technique for speeding up dynamic algorithms", *FOCS*, 1992, pp. 60–69.

[FIN93] A. Frank, T. Ibaraki, H. Nagamochi, "On sparse subgraphs preserving connectivity properties", *J. of Graph Theory,* 17(3), 1993, pp. 275–281.

[G91] H. Gabow, "A matroid approach to finding edge connectivity and packing arborescences", *STOC,* 1991, pp. 112–132.

[GJ79] M. R. Garey and D. S. Johnson, *Computers and intractability, a guide to the theory of NP–completeness*, Freeman, San Francisco, 1979, p. 198.

[IR88] A. Itai and M. Rodeh, "The Multi-Tree Approach to Reliability in Distributed Networks", *Information and Computation*, 79(1), 1988, pp. 3-59.

[M72] W. Mader, "Über minimal n-fach zusammenhängende, unendliche Graphen und ein Extremalproblem," *Arch. Mat.*, Vol. XXIII, 1972, pp. 553–560.

[M73] W. Mader, "Grad und lokaler Zusammenhang in endlichen Graphen," *Math. Ann.,* Vol. 205, 1973, pp. 9–11.

[M79] W. Mader, "Connectivity and edge-connectivity in finite graphs," in *Surveys on Combinatorics,* (B. Bollobas, ed.), London Math. Soc. Lecture Note Series, Vol. 38, 1979, pp. 293–309.

[NI92] H. Nagamochi and T. Ibaraki, "A Linear-time algorithm for finding a sparse k-connected spanning subgraph of a k-conneceted graph," *Algorithmica,* 7, 1992, pp. 583–596.

[T95] R. Thurimella, "Sub-linear Distributed Algorithms for sparse certificates and Biconnected Components," to appear in Proc. of the 14th ACM Symposium on Principles of Distributed Computing, August 1995.

[Z69] A. A. Zykov, *Theory of Finite Graphs,*, (in Russian), Nauka, Novosibirsk, 1969, see pp. 104–105.