Discrete Applied Mathematics 27 (1990) 113-123
North-Holland

SPARSEST CUTS AND BOTTLENECKS IN GRAPHS

David W. MATULA
Department of Computer Science, Southern Methodist University, Dallas, TX 75275, USA

Farhad SHAHROKHI
Department of Computer Science, University of North Texas, Denton, TX 76203, USA

Received 30 May 1989

The problem of determining a sparsest cut in a graph is characterized and its computation shown to be NP-hard. A class of sparsest cuts, termed bottlenecks, is characterized by a dual relation to a particular polynomial time computable multicommodity flow problem. Efficient computational techniques for determining bottlenecks in a broad class of instances are presented.

1. Introduction and summary

A *sparsest cut* in a graph [10] is a cut $(A, \bar{A})$ having the minimum density $|(A, \bar{A})| / |A|\,|\bar{A}|$ of edges taken over all cuts in the graph. More generally, for a network with a distinct demand (for message flow, traffic, etc.) between each vertex pair and a capacity on each edge, the sparsest cut problem is to determine a cut where the ratio of cut capacity to demand across the cut is minimum. Applications of sparsest cuts occur in hierarchical cluster analysis [9, 11], and telecommunications network design [6].

In Section 2 we investigate the complexity of determining a sparsest cut of a graph, termed the sparsest cut problem (SCP). A principal result is that the SCP is NP-hard. Despite the intractability in general, important classes of instances of the SCP have a tractable solution. We show the SCP is solvable in time linear in input size for trees, and in time $O(n^2\sqrt{\log n})$ for those n vertex planar graphs where all demands are between vertices on an outer face.

The SCP is related to the "near dual" maximum concurrent flow problem (MCFP) which has been investigated in [1, 2, 9, 10, 12, 13]. We herein term a sparsest cut which is a tight constraint for the MCFP a bottleneck, and the problem of determining bottlenecks is investigated in Section 3. We first show that the set of all "critical" edges (edges that must be saturated by any solution to the MCFP) can be determined in polynomial bounded time. The principal result of Section 3 is that if deletion of these critical edges yields a graph with at most four components, then a bottleneck (sparsest cut) exists and one can be identified in polynomially bounded time.

We further describe instances where an approximation algorithm for the MCFP

can provide an exact solution to the SCP, and illustrate with computational examples. We close with a discussion of some open problems in Section 4.

2. The sparsest cut problem

Let $G=\langle V,E\rangle$ be a graph which herein shall be assumed "weighted" by capacity and demand functions. Specifically, we shall utilize $C: E \to R^+$ to denote a *capacity* function on the edges of G, and $D: V \times V \to R^+$, with $D(i,i)=0$, $D(i,j)=D(j,i)$, to denote a *demand* function on the set of all vertex pairs of G.

Sparsest cut problem (SCP) [10]. For the partition $A, \bar{A}$ of V, the *cut* $(A,\bar{A})$ denotes all edges with one end vertex in A and the other in $\bar{A}$. We denote by $C(A,\bar{A})$ the capacity of the cut $(A,\bar{A})$, by $D(A,\bar{A})$ the demand across the cut $(A,\bar{A})$, and by $\mathrm{den}(A,\bar{A})$ the *density* of capacity relative to demand across the cut $(A,\bar{A})$. Herein let $\mathrm{cd}(G)$ denote the *minimum cut-density* over all cuts $(A,\bar{A})$ of G. So

$$C(A,\bar{A}) = \sum_{e \in (A,\bar{A})} C(e), \tag{1}$$

$$D(A,\bar{A}) = \sum_{i \in A,\, j \in \bar{A}} D(i,j), \tag{2}$$

$$\mathrm{den}(A,\bar{A}) = C(A,\bar{A})/D(A,\bar{A}) \quad \text{for } D(A,\bar{A})>0, \tag{3}$$

$$\mathrm{cd}(G) = \min_{(A,\bar{A})} \{\mathrm{den}(A,\bar{A}):\ D(A,\bar{A})>0\}. \tag{4}$$

A *sparsest cut* $(A,\bar{A})$ is a cut with $\mathrm{den}(A,\bar{A})=\mathrm{cd}(G)$. Given the capacity and demand functions for a graph, the *sparsest cut problem* (SCP) refers to the determination of the minimum cut-density $\mathrm{cd}(G)$ and some sparsest cut $(A,\bar{A})$.

Note that when the capacity C and demand D are unit functions, then $\mathrm{cd}(G)$ is given by the minimum proportion of edges in any cut, i.e., for an "unweighted" graph,

$$\mathrm{cd}(G) = \min_{(A,\bar{A})} \{|(A,\bar{A})|/|A|\,|\bar{A}|\} \quad \text{for } D,\ C=1. \tag{5}$$

Our principal result of this section is that the SCP is NP-hard.

Theorem 2.1. *The SCP is* NP*-hard.*

Proof. First note for unit demand functions that the problems of minimizing or maximizing the average capacity across a cut, $C(A,\bar{A})/(|A|\,|\bar{A}|)$, are of equivalent complexity. It is then sufficient to show that the max-cut problem [5] reduces to the problem of finding a maximum average capacity cut, to confirm that the SCP is NP-hard.

Given graph G, form the $2n$-vertex graph G^* composed of G and a disjoint iso-

morphic copy G' of G. Assign unit capacity to the edges of G and G' in G^*. Also insert an edge in $E(G^*)$ between $i \in V(G)$ and the corresponding $i' \in V(G')$ for each $i \in V(G)$, assigning capacity M to each of the n such edges. The resulting graph G^* is often termed a prism. For sufficiently large M, a maximum average capacity cut $(A, \bar{A})$ of G^* must have $|A| = |\bar{A}| = n$ with $i \in A$ and $i' \in \bar{A}$, or $i \in \bar{A}$ and $i' \in A$, since only such cuts can have average capacity as large as M/n. A cut $(A, \bar{A})$ of maximum density in G^* is then seen to have density $(nM + 2k)/n^2$ where k is the value of a maximum cut of G, and $(A \cap V(G), \bar{A} \cap V(G))$ must provide a maximum cut of the original graph G. □

For the case where G is a tree, the SCP is routinely solved by evaluating $\mathrm{cd}(G) = \min_e \{C(e)/D(A_e, \bar{A}_e)\}$, where A_e, $\bar{A}_e$ denotes the vertex partition of the tree derived by deleting edge e. By properly organizing the computation it is now shown that the SCP for a tree can be solved in time linear in input size.

Theorem 2.2. *The SCP for a tree with arbitrary capacity can be solved in optimal time*:

(i) $O(|V|)$ *for constant demand* $D: V \times V \to c$,

(ii) $O(|V|^2)$ *for arbitrary demand* $D: V \times V \to R^+$.

Proof. For each vertex $i \in V$ and edge $e \in E$, let $D(i, e)$ be the sum of all $D(i, j)$ where the unique path of the tree from j to i passes through edge e. Note that for each $i \in V$, the computation can be ordered (towards "root i" from the leaves) to yield $D(i, e)$ for all $e \in E$ in time at most proportional to the sum of the vertex degrees, which for a tree is $O(|V|)$.

For the constant demand case (i) let us compute $D(i, e)$ for all $e \in E$ for a single fixed root vertex i. Note then that $D(A_e, \bar{A}_e) = D(i, e) \times (|V| - D(i, e)/c)$ for every $e \in E$, so that $C(e)/D(A_e, \bar{A}_e)$ can be evaluated for all $e \in E$ to determine the minimum in time $O(|V|)$. This time complexity is best possible as the input size for capacity values is $|E| = \Omega(|V|)$.

For the arbitrary demand case (ii) observe that $D(i, e)$ is computable for all $i \in V$, $e \in E$ in $O(|V|^2)$ time. Now for all e, $e' \in E$, let $D(e, e')$ be the sum of all $D(i, j)$ where the unique path of the tree between i and j passes through both edges e and e'. By appropriate order of computation, employing as needed previously computed values of $D(e, e')$ and $D(i, e)$, values of $D(e, e')$ for all e, $e' \in E$ can be computed in $O(|V|^2)$ time. Then $D(A_e, \bar{A}_e) = D(e, e)$. The time complexity is best possible as the input size for demand is $\Omega(|V|^2)$, and the theorem follows. □

Our result for the SCP on a planar graph is limited to the case where all demands are between vertex pairs on an outer face, and is motivated by the results in [8].

Theorem 2.3. *Let* $G = \langle V, E \rangle$ *be a planar triconnected graph with capacity* C *and demand* D, *where* D *is nonzero only between vertices on the outer boundary cycle*

$v_0, v_1, v_2, \ldots, v_m = v_0$ *of an embedding of* G. *Then the SCP for* G, D, C *is solvable in* $O(|V|^2\sqrt{\log|V|})$.

Proof. For $1 \le i < j \le m$, let an ij-cut $(A, \bar{A})$ of G denote a cut where $\langle A\rangle$ and $\langle \bar{A}\rangle$ are each connected, with $v_k \in A$ for $i \le k < j$, and $v_k \in \bar{A}$ for $1 \le k < i$ or $j \le k \le m$. Thus an ij-cut intersects the boundary in the two edges $v_{i-1}v_i$ and $v_{j-1}v_j$.

Observing that $D(A, \bar{A})$ has the same value for any ij-cut, denote that value by D_{ij} for $1 \le i < j \le m$. Further let $C_{ij} = \min\{C(A, \bar{A})$: $(A, \bar{A})$ is an ij-cut$\}$ for $1 \le i < j \le m$. It is straightforward to show that some sparsest cut is an ij-cut, and it follows that $\mathrm{cd}(G) = \min_{i,j}\{C_{ij}/D_{ij}\}$. To complete our proof we need only show that C_{ij} and D_{ij} for all i, j can be computed in time $O(|V|^2\sqrt{\log|V|})$.

First consider the computation of the D_{ij}. By appropriately incrementing sums, the quantities $d_{ij} = \sum_{k=i}^{j-1} D(v_k, v_j)$ can be evaluated for all $1 \le i < j \le m$ in $O(|V|^2)$, and $d_j^* = \sum_{k \ne j} D(v_k, v_j)$ for $1 \le j \le m$ can be evaluated in $O(|V|^2)$. Then using the recurrence $D_{i,j+1} = D_{ij} + d_j^* - 2d_{ij}$, note that D_{ij} for all $1 \le i < j \le m$ can be determined in $O(|V|^2)$.

For determining the C_{ij}, consider the dual graph $G^* = (V^*, E^*)$ of the planar graph G. Let the length of an edge $e^* \in E^*$ equal the capacity of the corresponding edge $e \in E$. Then each C_{ij} can be computed as the length of a shortest path in G^* between the edges dual to $v_{i-1}v_i$ and $v_{j-1}v_j$. The single source shortest path problem for the planar graph G^* is solvable in $O(|V|\sqrt{\log|V|})$ time [4], so all C_{ij} can be determined in $O(|V|^2\sqrt{\log|V|})$, completing the theorem. □

3. Concurrent flows and bottleneck sparsest cuts

The sparsest cut problem has a close "near dual" association with a particular multicommodity flow problem termed the *maximum concurrent flow problem*. Results from the literature on the concurrent flow problem [1, 2, 9, 10, 12, 13] are cited and employed in this section to establish a broad class of graphs for which the SCP is efficiently solvable.

Maximum concurrent flow problem. Let $D: V \times V \to R^+$ and $C: E \to R^+$ denote given demand and capacity functions on the graph $G = \langle V, E\rangle$. Denote by P_e the set of all paths containing the edge e, by P_{ij} the set of all paths between distinct end vertices i, j, and by P the set of all nontrivial paths in G. A *concurrent flow* of *throughput* z in G is a function $f: P \to R^+$ such that:

$$\sum_{p \in P_{ij}} f(p) = zD(i,j), \quad \text{for all distinct } i, j \in V, \tag{6}$$

$$\sum_{p \in P_e} f(p) \le C(e), \qquad \text{for all } e \in E. \tag{7}$$

A *maximum concurrent flow* function f is a concurrent flow having maximum

throughput $\hat{z}$. The *maximum concurrent flow problem* (MCFP) refers to the determination of the maximum throughput $\hat{z}$ and some maximum concurrent flow function f.

By considering the total flow across a cut, one obtains the inquality $zD(A,\bar{A}) \leq C(A,\bar{A})$ for any throughput and any cut. It immediately follows that $\hat{z} \leq \mathrm{cd}(G)$. This leads us here to introduce the following subclass of sparsest cut problems.

Bottleneck sparsest cut problem. For G, D, C define any cut $(A,\bar{A})$ with density equal to the maximum throughput, i.e., with $\hat{z} = \mathrm{den}(A,\bar{A}) = \mathrm{cd}(G)$, to be a *bottleneck sparsest cut* (or simply *bottleneck*), and G to be a *bottleneck graph*. The determination of whether or not G, D, C has a bottleneck, and if so, the determination of a particular bottleneck is termed the *bottleneck sparsest cut problem*.

From the inequality $\hat{z} \leq \mathrm{cd}(G)$ there follows immediately a duality lemma for verifying that a particular cut is a bottleneck.

Lemma 3.1. *For the instance G, D, C of the SCP, suppose some cut $(A,\bar{A})$ of G has a density equal to the throughput of some concurrent flow f on G. Then G is a bottleneck graph with the bottleneck sparsest cut $(A,\bar{A})$ and maximum concurrent flow f.*

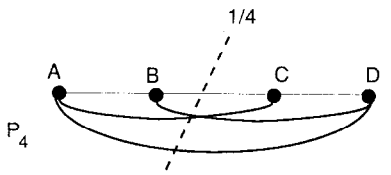
Figure 1 illustrates concurrent flow functions and particular cuts for four graphs assuming unit capacity and unit demand functions. The throughput is seen to equal the density of the indicated cut for the three graphs P_4, $K_1 \times P_4$ and W_5, confirming the cuts to be bottleneck sparsest cuts and the flows to be maximum concurrent flows in these three cases. For the graph $K_{2,3}$ of Fig. 1, it can be shown by other methods [10] that the indicated flow is a maximum concurrent flow with throughput 3/7, and the indicated cut is a sparsest cut with density 1/2. Thus $K_{2,3}$ has no bottleneck. On the other hand, employing unit demand and capacity functions we obtain from Lemma 3.1, as noted in [10], that the following classes of (unweighted) graphs are all bottleneck graphs: cycles, paths, trees, complete graphs and d-dimensional cubes for any d. The example $K_{2,3}$ of Fig. 1 is indicative of the fact that every complete bipartite graph $K_{i,j}$ for $i \geq 2$, $j \geq 3$ has no bottleneck. $K_{2,3}$ also confirms that not all planar graphs have bottlenecks.

Let us now proceed to the main issue of this section, which is how concurrent flow results can be employed and extended to identify instances of the SCP which are efficiently solvable. To identify a bottleneck from the solution of an MCFP we need the following. Let

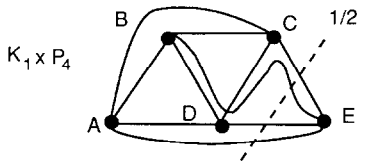
$$E_f = \left\{ e: \sum_{p \in P_e} f(p) = C(e) \right\} \tag{8}$$

denote the edges *saturated* by the maximum concurrent flow f on G, D, C, and

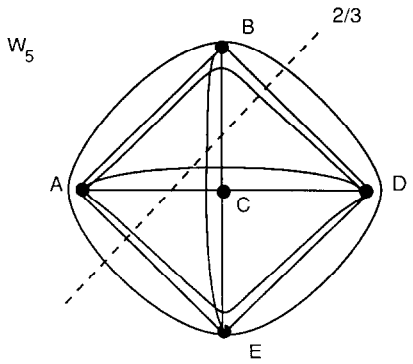
$$E_c = \bigcap \{E_f: f \text{ is a maximum concurrent flow on } G, D, C\} \tag{9}$$



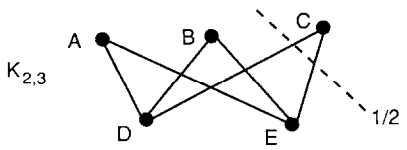
AB, BC, CD, ABC, BCD, ABCD @1/4



AB, AD, BC, BD, CD, CE
DE, ABC, ADE, BDCE @ 1/2



AB, AC, AE, BC, BD, CD, CE, DE @ 2/3
ABD, AED, BAE, BDE @ 1/6
ACD, BCE @ 1/3



AD, AE, BD, BE, CD, CE @ 3/7
ADB, ADC, AEB, AEC, BDC, BEC @ 3/14
DAE, DBE, DCE @ 1/7

Fig. 1. Concurrent flows yielding throughputs of 1/4 for P_4, 1/2 for $K_1 \times P_4$, 2/3 for W_5, and 3/7 for $K_{2,3}$. The sparsest cuts illustrated have densities 1/4 for P_4, 1/2 for $K_1 \times P_4$, 2/3 for W_5, and 1/2 for $K_{2,3}$.

denote the *critical* edges, which are then the edges that must be saturated by every maximum concurrent flow. Now if G, D, C has a bottleneck $(A, \bar{A})$, then $(A, \bar{A}) \subseteq E_c$. We first show that determination of E_c is a problem in $\mathscr{P}$.

Lemma 3.2. *Given G, D, C the set of critical edges E_c can be determined in polynomially bounded time.*

Proof. Employing standard linear programming formulations common to multicommodity flow problems, it follows [13] that the MCFP $\in \mathscr{P}$. Let f, $\hat{z}$ and E_f then be assumed determined for a given G, C, and D. For each saturated edge $e' \in E_f$ form the linear program:

$$\text{Minimize} \quad x = \sum_{p \in P_{e'}} f'(p),$$

$$\text{subject to} \quad \sum_{p \in P_{i,j}} f'(p) = \hat{z}D(i,j) \quad \text{for all } i, j \in V,$$

$$\sum_{p \in P_e} f'(p) \leq C(e) \quad \text{for all } e \in E - e'.$$

Note that $f' = f$ provides a feasible solution with $x = \sum_{p \in P_{e'}} f(p)$. A smaller value of x will be obtained iff e' is not critical. Thus solving at most $|E_f|$ linear programs of the form indicated determines the set of critical edges E_c. Note that restatement of each linear program in its corresponding node-arc [3] multicommodity network linear programming form will assure size polynomially bounded in input size and a solution of each linear program in polynomially bounded time. Thus E_c can be determined in polynomially bounded time. □

As evident from a more detailed inspection of Fig. 1, the critical edge set may contain more than the edges of a single cut. We shall use the notation $(A_1, A_2, \ldots, A_k)$, termed a *k-partite cut*, to denote precisely those edges of E whose end vertices are in distinct parts of the k-part partition $A_1, A_2, \ldots, A_k$ of V. We first note the following.

Lemma 3.3 [10, 13]. *The critical edges E_c for any G, D, C form the k-partite cut $(A_1, A_2, \ldots, A_k)$, where each A_i is the vertex set of a component of $G - E_c$.*

Our major result of this section is then:

Theorem 3.4. *Let G, D, C be an instance of the SCP where the k-partite cut of critical edges $E_c = (A_1, A_2, \ldots, A_k)$ for maximum concurrent flow has $2 \leq k \leq 4$. Then the SCP is solvable in polynomially bounded time with a bottleneck given by one or more of the following at most seven cuts; $(A_i, \bar{A}_i)$ for $1 \leq i \leq k$, and $(A_1 \cup A_i, \overline{A_1 \cup A_i})$, for $2 \leq i \leq 4$ when $k = 4$.*

Proof. By Lemmas 3.2 and 3.3, $E_c = (A_1, A_2, \ldots, A_k)$ can be found in polynomially bounded time. For $k \leq 4$, the cut of minimum density among $(A_i, \bar{A}_i)$ for $1 \leq i \leq k$, and $(A_1 \cup A_i, \overline{A_1 \cup A_i})$ for $2 \leq i \leq k$, can then be determined in polynomially bounded time. It remains to show that one of these at most seven cuts is a sparsest cut for any $2 \leq k \leq 4$. Our polynomial time bounded solution $(f, \hat{z})$ to the MCFP can be assumed [1, 13] to have at most $|V|^2$ paths $p \in P$ with $f(p) > 0$ and $E_f = E_c$. For the case $k = 2$, let $E_c = (A, \bar{A})$. Assume that p with $f(p) > 0$ contains more than one edge of $(A, \bar{A})$. Note then that some flow on p can be rerouted to another path between the same end vertices having at most one edge in $(A, \bar{A})$. This then provides another maximum concurrent flow function f' and a spanning tree T of G in which each edge has residual capacity, contradicting the maximality of the throughput. It follows that each path with $f(p) > 0$ crosses $(A, \bar{A})$ at most once, assuring $\hat{z} = \mathrm{cd}(G)$, and confirming $(A, \bar{A})$ to be a sparsest cut.

For the case $k=3$, let $E_c=(A_1,A_2,A_3)$. Suppose some path carries flow between, say, parts A_1 and A_3 passing through part A_2. Then no path can carry flow between A_1 and A_2 through A_3, since we could reroute flow, preserving throughput, and introduce slack capacity between A_2 and A_3. Similarly then, no path can carry flow between A_2 and A_3 through A_1. Hence $(A_1, A_2 \cup A_3)$ and $(A_1 \cup A_2, A_3)$ are both bottlenecks. Similar arguments yield the balance of the result for $k=4$. □

Theorem 3.4 is sharp in the sense that the graph $K_{2,3}$ of Fig. 1 has $G-E_c=\bar{K}_5$, and so yields the critical 5-partite cut $E_c=(A_1,A_2,A_3,A_4,A_5)=(\{v_1\},\{v_2\},\{v_3\},\{v_4\},\{v_5\})$, and has no bottlenecks.

The effectiveness of an algorithm for determining $E_c=(A_1,A_2,\ldots,A_k)$, and a sparsest cut $(A,\bar{A})$ when $k\leq 4$, based on the linear programming approach employed in the proof of Lemma 3.2, is limited in practice by the large size of the linear programs encountered. In [1,12] we have introduced a relatively efficient approximation algorithm for the MCFP. In [13] we prove the algorithm to provide a *fully polynomial time approximation scheme* [5] on instances G, D, C with arbitrary demand but constant capacity. We here present an extension of Lemma 3.1 that allows us to employ the more efficient approximation algorithm for the MCFP to obtain an (exact) solution of the SCP for many instances having both constant demand and capacity (which without loss of generality may each be taken as unit functions).

Theorem 3.5. *Let G, D, C be an instance of the SCP with unit demand between each vertex pair and unit capacity on each edge. Suppose given a particular concurrent flow f of throughput z and a particular cut $(A,\bar{A})$ with*

$$\operatorname{den}(A,\bar{A})-z<16/|V|^4.$$

Then $(A,\bar{A})$ is a sparsest cut of G.

Proof. Note that for unit capacity and demand functions, with $m=|V|^2/4$, $\operatorname{den}(A,\bar{A})=|(A,\bar{A})|/(|A|\,|\bar{A}|)=i/j\in F_m$, where F_m is the Farey set composed of all irreducible fractions with numerator and denominator at most m. If some cut $(B,\bar{B})$ of G has density less than that of $(A,\bar{A})$, then $z\leq \operatorname{den}(B,\bar{B})=k/l<\operatorname{den}(A,\bar{A})=i/j$, where $k/l, i/j\in F_m$. From the theory of Farey fractions it is known [7] that distinct elements of F_m differ by at least $1/m^2$, so $\operatorname{den}(A,\bar{A})-z\geq i/j-k/l\geq 16/|V|^4$, a contradiction. Thus $(A,\bar{A})$ is a sparsest cut. □

The approximation algorithm for the MCFP was applied to a variety of graphs in [12] including the eight graphs illustrated in Fig. 2. Data from [12] on near optimal throughput is tabulated in Table 1 along with the density of the cuts illustrated in Fig. 2. The fact that the cuts illustrated in graphs 1–6 are sparsest cuts follows immediately from Theorem 3.5.

The cuts illustrated in graphs 7–8 may also be shown to be sparsest cuts by closer

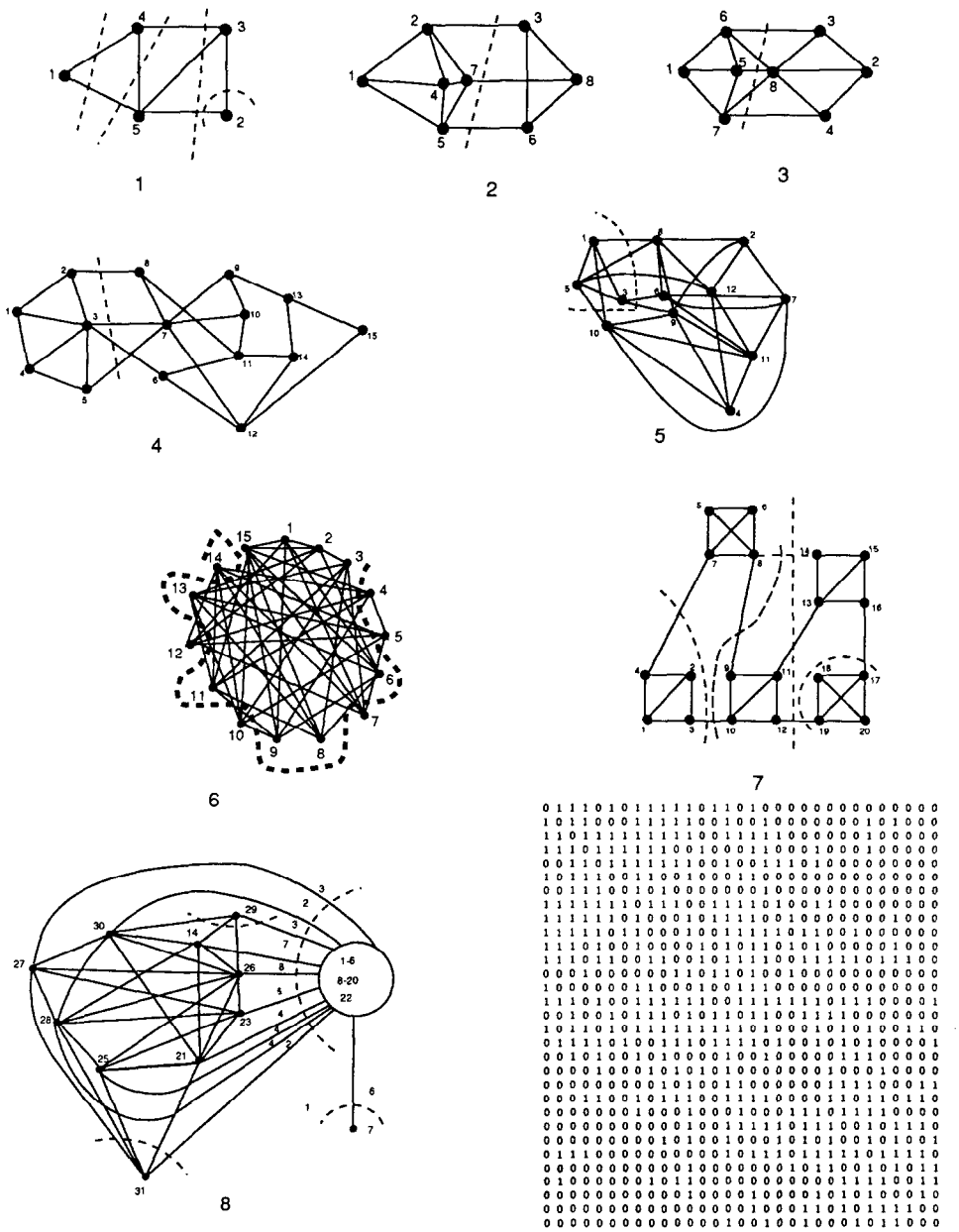


Fig. 2. Graphs 1–8 of Table 1 with all sparsest cuts illustrated.

inspection of the rational values and Farey sets involved. For graph 8, we note that no fraction of F_{240} has a value interior to the interval $[47/236, 1/5]$. Since for the feasible throughput $z = 0.1997$ we have $47/236 < 0.1997 \leq 1/5$, the illustrated cuts

Table 1. An achievable throughput z for concurrent flow from [12] and the density of an illustrated cut for the eight graphs of Fig. 2.

Graph	$\|V\|$	$\|E\|$	z	$\text{den}(A,\bar{A})$
1	5	7	0.4993	0.5000 (=2/4)
2	8	14	0.1999	0.2000 (=3/15)
3	8	15	0.31209	0.31250 (=5/16)
4	15	26	0.07998	0.08000 (=4/50)
5	12	31	0.25906	0.25926 (=7/27)
6	15	49	0.27761	0.27777 (=15/54)
7	20	33	0.0290	0.03125 (=2/64)
8	31	188	0.1997	0.2000 (=6/30)

for graph 8 of density 0.2 are each sparsest cuts. For graph 7 we note that the only fraction of the form $k/(i(20-i))$ interior to the interval $[0.0290, 0.03125]$ is $3/100$. Inspection of graph 7 reveals no cut $(A,\bar{A})$ with $|(A,\bar{A})|=3$ and $|A|=|\bar{A}|=10$, so the four cuts shown of density 0.03125 are each confirmed to be sparsest cuts.

The sparsest cut problem has applications in cluster analysis and in telecommunications routing [6, 11]. In both applications the identification of a sparsest cut with the constraining features characterizing our notion of a bottleneck is of practical significance. The empolyment of concurrent flows to determine such bottleneck sparsest cuts thus has considerable practical, as well as theoretical, value.

4. Open questions

As the bottleneck sparsest cuts we have here characterized exhibit a duality between cuts and flows, further analysis should be possible by a variety of methods. Some questions of interest are:

(1) Is the bottleneck sparsest cut problem in $\mathscr{P}$?

(2) What is the best time complexity one can obtain for identifying if G, D, C has a bottleneck, and if so, for determining one particular sparsest cut?

(3) What further classes of unweighted graphs (assuming unit demand and capacity functions) can be determined to contain exclusively bottleneck graphs?

References

[1] R.P. Bianchini and J.P. Shen, Interprocessor traffic scheduling algorithm for multiple-processor networks, IEEE Trans. Comput. 36 (1987) 396-409.

[2] J. Biswas and D.W. Matula, Two flow rerouting algorithms for the maximum concurrent flow problem, in: Proceedings FJCC (1986) 629-636.

[3] L.R. Ford and D.R. Fulkerson, Flows in Networks (Princeton University Press, Princeton, NJ, 1962).

[4] G.N. Frederickson, Shortest path problems in planar graphs, in: Proceedings 24th IEEE Symposium on Foundation of Computer Science (1983) 242–247.
[5] M.R. Garey and D.S. Johnson, Computers and Intractability (Freeman, New York, 1979).
[6] M. Gerla, H. Frank, W. Chou and J. Eckl, A cut saturization algorithm for topological design of packet switched communication networks, in: Proceedings NTC (1974) 1074.
[7] G.H. Hardy and E.M. Wright, An Introduction to the Theory of Numbers (Oxford University Press, Oxford, 1979).
[8] K. Matsumoto, T. Nishizeki and N. Saito, An efficient algorithm for finding multicommodity flows in planar networks, SIAM J. Comput. 14 (1985) 289–301.
[9] D.W. Matula, Cluster validity by concurrent chaining, in: J. Felsenstein, ed., Numerical Taxonomy, NATO Advanced Science Institutes Series G 1 (Springer, New York, 1983) 156–166.
[10] D.W. Matula, Concurrent flow and concurrent connectivity in graphs, in: Y. Alavi et al., eds., Graph Theory and its Applications to Algorithms and Computer Science (Wiley, New York, 1985) 543–559.
[11] D.W. Matula, Divisive vs agglomerative average linkage hierarchical clustering, in: W. Gaul and M. Schader, eds., Classification as a Tool of Research (North-Holland, Amsterdam, 1986) 289–301.
[12] F. Shahrokhi and D.W. Matula, On solving large maximum concurrent flow problems, in: Proceedings ACM Computer Conference (1987) 205–209.
[13] F. Shahrokhi and D.W. Matula, The maximum concurrent flow problem, J. ACM (to appear).