

The NP-Completeness Column: An Ongoing Guide

DAVID S. JOHNSON

AT&T Bell Laboratories, Murray Hill, New Jersey 07974

This is the fourteenth edition of a quarterly column that provides continuing coverage of new developments in the theory of NP-completeness. The presentation is modeled on that used by M. R. Garey and myself in our book "Computers and Intractability: A Guide to the Theory of NP-Completeness," W. H. Freeman & Co., New York, 1979 (hereinafter referred to as "[G&J]"; previous columns will be referred to by their dates). A background equivalent to that provided by [G&J] is assumed, and, when appropriate, cross-references will be given to that book and the list of problems (NP-complete and harder) presented there. Readers who have results they would like mentioned (NP-hardness, PSPACE-hardness, polynomial-time-solvability, etc.), or open problems they would like publicized, should send them to David S. Johnson, Room 2C-355, AT&T Bell Laboratories, Murray Hill, NJ 07974 (CSNET address: dsj.btl@csnet-relay). Please include details, or at least sketches, of any new proofs (full papers are preferred). If the results are unpublished, please state explicitly that you would like them to be mentioned in the column. Comments and corrections are also welcome. For more details on the nature of the column and the form of desired submissions, see the December 1981 issue of this Journal.

1. INTRODUCTION

The last column surveyed complexity issues arising from the use of communication networks. This time I shall consider the *design* of networks, both those used for communication and those adapted to more arcane ends. Not all the topics labelled as "network design" in [G&J] will be covered; "routing" was covered in [Dec. 1982] and new routing results will be included in a future column on "flow problems." Nevertheless, there is extensive territory to be surveyed. The itinerary, ignoring the occasional detour, is as follows.

Section 2 considers problems like the minimum spanning tree problem, in which the goal can be viewed as building an optimal network from a given collection of edges and vertices. Section 3 deals with the difficulties involved in evaluating the suitability of a network for a given task: Is it sufficiently reliable? How well does it work? Does it work at all? Section 4 then concludes with problems about networks that are conceptual in nature, used primarily for organizing data rather than transmitting it.

2. AN ORCHARD OF SPANNING TREES

Given a connected graph $G = (V, E)$ with weights on the edges, a *spanning tree* for G is a subgraph that is a tree and that contains all the vertices in V . Alternatively, it is a maximal acyclic subgraph. In any case, such trees can easily be found in time linear in the number of edges in G , and since you are reading this column, you probably already know a variety of ways of finding one with minimum total edge weight in time that is not much worse. (“Not much worse” is getting less and less worse all the time, incidentally. As of this writing, the running time for the best minimum weight spanning tree algorithm has dropped to roughly $O(|E| \log \log^* |V|)$ [19]).

Nevertheless, although the basic minimum spanning tree problem may be easy, it does not take much extra to make it difficult; eight NP-complete variants were listed in [G&J], and this was just a start. A much more extensive classification of spanning tree variants has recently been developed by Camerini, Galbiati, and Maffioli. In a series of papers [8,9,10], they have devised a list that now extends to 18 different spanning tree parameters that one might wish to optimize, and determined for each the complexity of finding a spanning tree that minimizes (or maximizes) that measure. In addition, they have classified the doubly constrained problems in which one wishes to satisfy bounds for two of the parameters simultaneously. There is not room here to give a complete summary of their results, but I will present a sample relating to a parameter that has obvious applications to communication networks.

[1] MINIMUM MAX-FLOW SPANNING TREE

INSTANCE: Graph $G = (V, E)$, bound $B > 0$.

QUESTION: Is there a spanning tree $T = (V, E')$ for G such that for all edges $e \in E'$, the “flow” on e , i.e., the number $f(e)$ of pairs of vertices $u, v \in V$ that would be disconnected if e were removed from T , is no more than B ? (The obvious application is to minimizing the maximum traffic over any single link of a communication network, or the maximum service interruption should one link go down.)

Reference. Camerini, Galbiati, and Maffioli [8]. Transformation from 3-DIMENSIONAL MATCHING.

Comment. If instead of asking that the *maximum* of the $f(e)$'s be less than B we ask that their *sum* be less than B , we have the NP-complete SHORTEST TOTAL PATH LENGTH SPANNING TREE problem ([ND3] in [G&J]) and a special case of OPTIMUM COMMUNICATION SPANNING TREE [ND7]. (For new information on subcases and variants of the latter, see [30] for NP-completeness results and [1] for polynomial-time algorithms.) If for each edge

we count only the number of vertices that it separates from a prespecified root v_0 , and we wish that the maximum of these revised flows be bounded by B , then we get an NP-complete subproblem of CAPACITATED SPANNING TREE [ND5]. However, if we replace “max” by “sum” in this variant we obtain polynomial time solvability, since all we are asking for is a shortest path tree [8]. If instead of asking that all flows be B or less, we ask that there exist at least one edge with $f(e)$ greater than B , I’m no longer sure what the “obvious” application is, but we again have an NP-complete problem in the unrooted case, although not in the rooted one. For the rooted case, a spanning tree with such a high-flow edge can be found in polynomial time [9] — unless, of course, one wishes simultaneously to minimize the tree height, minimize the tree diameter, minimize the sum of the distances to the root, or maximize the maximum vertex degree. Each of these doubly-constrained problems is NP-hard, despite the fact that the individual constraints can be handled separately in polynomial time [9].

I have now mentioned 6 of the parameters covered in [8,9,10]; for the other 12, and the classifications they engender, the reader is directed to the papers themselves. There is one recent spanning tree result that will not be found there, however. It involves a 19th parameter, whose omission is perhaps excusable in light of the subtlety of its connection to communication networks.

[2] MINIMUM CYCLE BASIS SPANNING TREE

INSTANCE: Graph $G = (V, E)$, bound $B > 0$.

QUESTION: Is there a spanning tree T of G that yields a “cycle basis” for G of total length B or less? (The cycle basis corresponding to a given spanning tree T consists of all those simple cycles of G that contain exactly one non-tree edge.)

Reference. Deo, Prabhu, and Krishnamoorthy [15]. Transformation from SHORTEST TOTAL PATH LENGTH SPANNING TREE.

Comment. A *cycle basis* for G is a basis for the vector space consisting of all those subsets of the edge set E that are either cycles in G or the symmetric differences of sets of cycles in G . A *minimal cycle basis* is a cycle basis no proper subset of which is also a basis. Although any spanning tree generates a minimal cycle basis in the manner described above, not all minimal bases correspond to spanning trees, and there exist graphs for which the cycle basis of minimum total length does not correspond to any spanning tree (e.g., see [15]). Thus the question remains open of determining, given G and B , whether G has *any* cycle basis of total length B or less.

Let us now turn to problems where the desired subgraph is not *a priori* required to be a tree. In [34], Plesnik considers the problem, given a graph with non-negative lengths and costs on the edges, of determining whether there is a subgraph that simultaneously obeys given bounds on diameter and total edge cost. Although the subgraph need not be a tree, it is easy to see that this problem contains BOUNDED DIAMETER SPANNING TREE [ND4] as a special case (the edge weights can be modified to limit the subgraph to $|V| - 1$ edges). Plesnik's paper shows that a variety of new special cases are NP-complete, and that even guaranteeing a close approximation in these cases is NP-hard. The next problem also concerns diameters of subgraphs, only now we are interested not only in the subgraph we construct but also in the residue left when that subgraph is taken away.

[3] BOUNDED DIAMETER DECOMPOSITION

INSTANCE: Graph $G = (V, E)$, bound $D > 0$.

QUESTION: Is there a subset $E' \subseteq E$ such that both $G' = (V, E')$ and $G'' = (V, E - E')$ have diameter D or less?

Reference. Plesnik [35]. Transformation from HYPERGRAPH 2-COLORABILITY [SP4].

Comment. Remains NP-complete even if $D = 3$ and G is bipartite. Analogous results hold for radii and for directed graphs [35].

In real networks, our desire to minimize diameter is often thwarted because there is a practical bound on the maximum vertex degree. Indeed, an ongoing competition in the literature concerns constructing, for each D and k , graphs with a maximum number of vertices, given that no vertex degree may exceed k and the diameter must be D . For instance, as of September, 1984, the largest published graph with diameter 8 and maximum vertex degree 3 contained 200 vertices. For a bibliography and a table of current bests, see [14].

Note that the two-constraint problem of telling whether a *given* graph G has a subgraph with diameter at most D and maximum vertex degree at most k is clearly NP-complete. Indeed, the degree bound alone is enough to ensure NP-completeness, so long as one desires a *connected* subgraph — see the [G&J] problem DEGREE CONSTRAINED SPANNING TREE [ND1]. On the other hand, if all one wants is a not-necessarily-connected subgraph of maximum edge weight that obeys an upper bound on vertex degrees (but no diameter bound), we have the “*b*-matching” problem of Edmonds, and maximum weight subgraphs can be found in polynomial time even if separate upper bounds are specified for each vertex, and even if lower bounds are also provided (e.g., see [31], page 245). If the upper and lower bounds are equal and G is a tree, one can even compute the number of such subgraphs in polynomial time [43]. There

is still room for an NP-completeness result or two, however.

[4] DEGREE CONSTRAINED SUBGRAPH

INSTANCE: Graph $G = (V, E)$, for each vertex $v \in V$ a set D_v of allowable degrees.

QUESTION: Does G contain a subgraph G' such that if $d(v)$ is the degree of vertex v in G' for all $v \in V$, then $d(v) \in D_v$ for all $v \in V$?

Reference. Plesnik and Wawruch [36]. Transformation from EXACT COVER BY 3-SETS.

Comment. Remains NP-complete even for bipartite planar graphs of maximum degree 3. Solvable in polynomial time if each set D_v is a set of 1 or more consecutive integers (see the remarks above on “ b -matchings”). If, however, we are given just the desired degrees, and are not constrained as to which vertices get which degrees, the problem once more becomes NP-complete [11], and also becomes quite similar to (and much more sensible than) the [G&J] problem ELIMINATION DEGREE SEQUENCE [GT47].

Our next problem may have more to do with social networks than communication networks. Suppose we want to form a new club and to maximize the overall prestige of its membership. The club is to be an exclusive one, so no one will be asked to join unless that person already knows a member. Unfortunately, we are not on close speaking terms with all the social lions we wish to collect, and our only approach to some of them is through mutual acquaintances of somewhat questionable standing. We are then faced with the following.

[5] MAXIMUM WEIGHT CONNECTED SUBGRAPH

INSTANCE: Graph $G = (V, E)$, integral (and possibly negative) weight $w(v)$ for each $v \in V$, bound B .

QUESTION: Is there a connected subgraph of G whose total vertex weight is at least B ?

Reference. Vergis [49]. Transformation from STEINER TREE IN GRAPHS [ND12].

Comment. Remains NP-complete even if G is planar with maximum degree 3 and all weights either $+1$ or -1 . Note that in this problem we may assume that the desired subgraph is a subtree, since additional edges serve no purpose. Also NP-complete is the variant in which the labels are on the edges rather than the vertices, and the desired subgraph is *required* to be a subtree [7].

The Steiner tree problem mentioned above is of course itself a network design problem: Given an edge-weighted graph $G = (V, E)$, a specified subset V' of vertices, and a bound B , does there exist a subtree of G that connects all the vertices of V' (but not necessarily all those of V) and has total edge weight B or less? I mentioned new results for special cases of this problem in both the [Dec. 1982] and the [March 1984] columns. I can now add [March 1985] to this list of citations, by pointing out here that the problem is also solvable in polynomial time for “separable rectangle trees” [16].

Camerini, Galbiati, and Maffioli, mentioned above for their horticultural success with spanning trees, have also undertaken the task of classifying variants on the Steiner tree problem. In [7], they consider much the same range of optimization criteria as they did for spanning trees, determining for each whether it leads to NP-completeness or polynomial-time solvability. Their classification also covers problems in which the desired subtree can contain *only* vertices from the specified subset (but not necessarily all of them) and problems in which the subtree must meet a bound on the number of vertices, rather than contain or be contained in a particular subset. For the details of the classification (and its one remaining open problem) see [7]. For more details on the case where one wants a subtree with precisely k vertices, see [33], which shows that despite NP-completeness for most criteria, there do exist polynomial-time algorithms if the input graphs are restricted to trees and the optimization criterion is “monotone.” For a Steiner variant that managed to escape classification, in which the required set is split into two sets V_1 and V_2 and the goal is a tree that meets individual distance bounds for each pair of one vertex from V_1 and one from V_2 , see [6].

3. DOES THE NETWORK WORK?

Although there are many complexity-theoretic pitfalls involved in problems of network design, there is one universally effective (and quite popular) procedure for avoiding the dangers: Let someone *else* design the network. Unfortunately, this is only a partial solution. Your delegate may promptly return with a network completely designed, but how are you to know whether the design meets your original specification?

For instance, consider the question of *reliability*. Given that individual links in a network may fail (bridges may collapse, etc.), one would like the failure probability for the network as a whole to be as low as possible. Calculating that probability, unfortunately, is nontrivial. More precisely, suppose we are given a graph $G = (V, E)$, a *source* vertex $s \in V$, and an edge failure probability p . The problem of determining the “network reliability,” i.e., the probability that there is a path from s to all other vertices in V , given that each edge fails with independent probability p , is now known to be #P-complete. (Previously, the basic problem was only known to be NP-hard; see [ND20] in [G&J].) #P-

completeness was proven independently in [26,38] for undirected graphs and in [22,26,38] for directed ones. Even approximating the reliability to within a given ϵ is #P-complete [38]. If, however, the graph is directed and acyclic, network reliability can be computed in polynomial time [4].

Similar progress has been made in studying the related “ s - t reliability” problem where we are given a *sink* vertex t in addition to our source s , and asked for the probability that at least one path from s to t survives. Valiant proved this problem to be #P-complete for general directed or undirected graphs [48]; Provan has now shown that it remains #P-complete even when G is a simple planar grid graph, undirected or directed (and even if acyclic in the latter case) [37]. This is true even though many of the cut- and path-counting problems related to s - t reliability can be solved in polynomial time for such graphs [4], and even though acyclicity alone was enough to guarantee polynomial time solvability for “network reliability,” as mentioned above. Special cases of s - t reliability that *are* solvable in polynomial time include series-parallel graphs in the undirected case [42] and acyclic graphs whose underlying graph is series-parallel in the directed case [2].

A different kind of “special case” algorithm is implicit in [39], where Provan and Ball present algorithms for general graphs (directed or undirected) that compute s - t reliability in time polynomial in $|V|$ and the number of minimal s - t cuts in G . That number can be exponentially large, but need not be so in all cases. Analogous algorithms exist for the undirected and directed “network reliability” questions, with “cutsets” replaced by spanning trees [39]. These results are not automatic; for the problem of determining the probability that s is still connected to a prespecified *subset* T of the vertices, no algorithm can run in time polynomial in $|V|$ and the number of minimal s - T cuts (or the number of trees linking s to T) unless $P = NP$ [39]. For more results of this sort, and some open questions, see [39].

Even the few positive results mentioned above go away if we abandon our assumption that all edge failure probabilities are equal and independent. Without this assumption one can still solve certain simple problems, such as finding the spanning tree of G with maximum network reliability, or even the spanning tree with the optimal cost/reliability ratio [12], but determining reliability of anything more complicated than a tree seems hopeless. In practice, the situation is even worse than this, however. Not only do edge failure probabilities differ from edge to edge, but they most likely are not independent and most certainly are not precisely known. We thus confront something like the following problem.

[6] NETWORK RELIABILITY UNDER UNCERTAINTY

INSTANCE: Graph $G = (V, E)$, subset $V' \subseteq V$, for each edge $e \in E$ a lower bound $a(e)$ and an upper bound $b(e)$ on its failure probability, and a bound B .

QUESTION: Is there any assignment of a probability to each subset $E' \subseteq E$ (representing the probability that the edges in E' fail and those in $E - E'$ do not) in such a way that for each edge e the cumulative failure probability for e lies between $a(e)$ and $b(e)$, and in such a way that the probability that all vertices in V' remain connected to each other is greater than B ?

Reference. Zemel [50]. Transformation from STEINER TREE IN GRAPHS.

Comment. NP-hard, not known to be in NP. Surprisingly, one can in polynomial time determine whether there is an assignment that yields a probability *less* than B that V' is still connected. In fact one can determine the best possible such lower bound in polynomial time. The key trick is to notice that both upper and lower bound problems involve solving linear programs with an exponential number of constraints, and in the latter case the techniques of Grötschel, Lovász, and Schrijver [21] apply, whereas they do not in the former. Similar results hold for the “uncertain” version of NETWORK SURVIVABILITY [ND21] and a variety of other problems [50].

Given that edges may fail, one may well wish to devise ways of locating such failures. If the network under consideration were an electronic network, one might test for failures by placing probes at two vertices and seeing if the generated current (a function of the paths in the network between the two vertices) is as expected. If not, then one of those paths contains a failed edge. Problem [MS18] in [G&J] concerns this problem when G is directed and the objects subject to failure are vertices, rather than edges. As stated there, it is NP-hard to find a minimum sized collection of tests that is always sufficient to locate a failed vertex when there is only one such failure in the network. It turns out that the NP-hardness of this vertex-failure problem is shared by our edge-failure problem [24]. However, recent results by Ibaraki, Kameda, and Toida [24,25] offer hope for networks that are (what else?) trees. Here minimum test sets can be constructed in polynomial time in both the directed [24] and undirected [25] case.

While we are on the topic of testing electronic networks, let me mention recent improvements on the NP-completeness result for FAULT DETECTION IN LOGIC CIRCUITS [MS17]. In this problem the vertices are now logical gates, an edge failure results in the edge being “stuck at” a fixed value (0 or 1) and the tests are vectors supplied to the circuit’s inputs to see if the circuit outputs the expected value. The question asked is whether all single faults are detectable by such experiments (in which case the circuit is called *irredundant*). Fujiwara and Toida [17] show that this problem remains NP-complete even if the circuit is “monotone” or “unate,” although there are certain classes of “real life” circuits for which the problem is in P. Krishnamurthy and Akers [28] prove NP-hardness for the following problem and some variants: Given an

irredundant circuit, find a minimum sized set of test vectors sufficient for detecting all single stuck-at faults. A more sophisticated problem concerning the reliability of electronic networks, where now the network consists of a set of nodes interconnected by (MOS) transistor switches, and my knowledge of the technology is sufficiently limited to preclude an exact description, goes roughly as follows.

[7] RACE DETECTION

INSTANCE: A description of a feedback-free (i.e., acyclic) MOS circuit C at the transistor switch level, with initial values for all the nodes, new values for the input nodes, and a specified node t .

QUESTION: If we assume the model where a node's value can be either 0 or 1 and transitions between the two values take place instantaneously, does the new set of input values cause a race condition at t i.e., is there more than one value into which t can settle, depending on the delays incurred along the paths in C from the input nodes to t ?

Reference. Ramachandran [40,41]. Transformation from 3SAT.

Comment. Remains NP-complete as long as C is allowed to have "fanouts" of at least 2. Solvable in linear time if only "unit fanout" is allowed [40,41]. In the model where node values go through an intermediate *undefined* value on their way between 0 and 1, race conditions of this type are more likely, and can be detected in polynomial time even in circuits with feedback, although the detection of more sophisticated types of race conditions appears to remain NP-hard [29].

To conclude this section, we turn from questions about whether a network is reliable to one about whether it works at all.

[8] DEFECTIVE SUPERCONCENTRATOR

INSTANCE: Acyclic directed graph $G = (V, A)$, disjoint sets $I, O \subseteq V$ of "input" and "output" vertices, with $|I| = |O|$.

QUESTION: Is it the case that G is *not* a superconcentrator? (It would be one if, for all sets $S \subseteq I$ and $T \subseteq O$ with $|S| = |T|$, there exist $|S|$ vertex-disjoint paths between S and T .)

Reference. Blum, Karp, Vornberger, Papadimitriou, and Yannakakis [5]. Transformation from CLIQUE (of size $|V|/2$) [GT19].

Comment. That the problem is in NP follows from the min-cut max-flow

theorem. Superconcentrators have proved useful as counter-examples and in the proofs of lower bounds [47]. A potential “real-world” application would be to switching networks, but for superconcentrators to be competitive with existing techniques, they would have to have a small number of edges, say fewer than $10n$, where n is the number of inputs. It was not even known until 1975 that superconcentrators with $O(n)$ edges existed. However, there has been steady progress since then, with the constant of proportionality dropping from 238 [46] to 36 when last I looked [13]. Unfortunately, these bounds are non-constructive; the best constant of proportionality for an explicitly constructed superconcentrator, which started at 271.8 in 1979 [18], has so far dropped only to 157.4 [3]. The non-constructive results, however, do provide methods for “randomly generating” structures that obey the much smaller size bounds and have a high probability of being superconcentrators. This suggests the following procedure for finding such smaller superconcentrators: Use the random procedure to generate structures, testing each for superconcentratorhood until a non-defective one is found. The current result indicates that there are serious drawbacks to such an approach. Also NP-complete are the problems of identifying defective “matchers” and “concentrators” [5]. (A *matcher* is a bipartite graph $G = (V_1, V_2, E)$ with $|V_1| = |V_2| = 2M$, such that for each $S \subseteq V_1$ with $|S| = M$, there is a matching between S and some subset of V_2 . A *concentrator* is an acyclic directed graph $G = (V, A)$ with disjoint sets $I, O \subseteq V$ of input and output vertices such that, for every subset $S \subseteq I$ with $|S| = |O|$, there are $|S|$ vertex-disjoint paths from S to O .) The latter problem remains NP-complete even if G contains no path of length longer than two, but can be solved in polynomial time if it contains no path of length longer than one (all the arcs in G are from vertices in I to ones in O) [5].

4. CONCEPTUAL NETWORKS

In this final section we consider problems of network design for networks whose purpose is more representational than transportational. An example is the “PERT network,” an acyclic directed graph in which the arcs represent tasks to be performed and the length of an arc represents the expected time to perform the corresponding task. Once constructed, such a network can be analyzed to determine bottlenecks in the overall project (i.e., arcs along the longest path from the start vertex to the finish vertex). An alternative representation, more familiar to scheduling theorists, uses *vertices* to represent tasks, with an arc from v_i to v_j if the task associated with v_i must be completed before the task associated with v_j can begin. Problem [ND44] in [G&J] concerns the conversion from the latter representation to the former, which sometimes requires the creation of “dummy tasks.” Minimizing the number of these is NP-hard. I bring this up because of a recent related result: Syslo [44] has developed a polynomial-time algorithm for determining whether dummy tasks can be

avoided, and for constructing a dummy-free PERT network if one exists.

Another type of conceptual network, familiar to computer science majors everywhere, is the *data structure*, which in most cases can be viewed as a directed graph with data items labelling its vertices. Designing optimal data structures can be just as hard as designing other kinds of networks, e.g., see PRUNED TRIE SPACE MINIMIZATION [SR3] in [G&J]. Here is a new data structure complexity result.

[9] MULTIWAY SEARCH TREE SEARCH COST MINIMIZATION

INSTANCE: Sequence K_1, K_2, \dots, K_n of keys, for each key K a positive integer size $s(K)$, an integer value $v(K)$ such that $v(K_1) < v(K_2) < \dots < v(K_n)$, and a probability $p(K)$, together with probabilities $p(-\infty, v(K_1))$, $p(v(K_1), v(K_2))$, ..., $p(v(K_n), \infty)$ for each of the open intervals (*gaps*) bounded by key values ($-\infty$ and $+\infty$ are considered honorary key values), a size limit L , and a cost bound B .

QUESTION: Is there a multiway search tree for the keys such that no vertex is labelled by keys of total size exceeding L and such that the expected search cost is bounded by B ? A multiway search tree is a rooted tree, each vertex of which is labelled by a set of keys, with each key assigned to precisely one vertex. The number of keys in the set labelling u must equal one less than the number of children of u (0 if u is a leaf). Moreover, if u is the k th child of w in the tree, then the subtree rooted at u represents the interval from the $k-1$ st smallest value for a key labelling w to the k th smallest, and the vertices of the subtree rooted at u collectively contain all the keys with values in this (open) interval. (If $k = 1$, then “the $k-1$ st smallest” is replaced by “the largest overall key value less than all the values labelling w and its subtrees.” If w has only $k-1$ keys, then “ k th smallest” is replaced by “the smallest overall key value greater than all values labelling w and its subtrees.”) The search cost for a given key value is the length of the path from the root to the vertex labelled with that key; for a gap it is the length of the path to the unlabelled leaf representing that gap.

Reference. Szwarcfiter [45]. Transformation from PARTITION.

Comment. Remains NP-complete if we require that the tree have all its leaves at the same level, so long as we also prespecify a *lower bound* L' in addition to our upper bound L on the total size of a set of keys that can label a non-leaf vertex. (This corresponds to looking for a “weak B-tree.”) Both problems can be solved in pseudo-polynomial time by dynamic programming. Among the variants solvable in ordinary polynomial time are finding multiway search trees with minimum height or with minimum total space [45]. Another type of data structure with a recently discovered polynomial-time optimization algorithm is the “binary split tree” [32].

For our final problem, we consider the design of graphs to represent systems of sets in a coherent visual form, with applications to VLSI design as well as graphics.

[10] PLANAR VENN DIAGRAM

INSTANCE: Hypergraph $H = (V, C)$. (Recall that in a hypergraph, the set C of “hyperedges” is a collection of arbitrary subsets of V , rather than just 2-element subsets as in an ordinary graph.)

QUESTION: Is there a planar graph $G = (V, E)$ such that for each $c \in C$, the subgraph of G induced by c is connected?

Reference. Johnson and Pollak [27]. Transformation from PLANAR CUBIC HAMILTONIAN PATH.

Comment. The “Venn diagram” in question is obtained by embedding G in the plane and constructing the planar dual G^* . Each vertex of H is then represented by a face of G^* , and each hyperedge by a connected region in the plane. The problem remains NP-hard even if we require these regions to be convex (or rectangular). The variant in which we require G to be a tree and each $c \in C$ to induce a path has applications to telling whether a binary matroid is graphic and can be solved in polynomial time [20]. In the VLSI application of the original problem, the vertices are circuit elements, the hyperedges are subsets of elements that must be wired together in a one-layer circuit, and we are interested in G itself, not G^* [27].

REFERENCES

1. S. AGARWAL, A. K. MITTAL, AND P. SHARMA, Constrained optimum communication trees and sensitivity analysis, *SIAM J. Comput.* **13** (1984), 315-328.
2. A. AGRAWAL AND A. SATYANARAYANA, An $O(|E|)$ time algorithm for computing the reliability of a class of directed networks, *Operations Res.* **32** (1984), 493-515.
3. N. ALON AND V. D. MILLMAN, Eigenvalues, expanders and superconcentrators, in “Proceedings 25th Ann. Symp. on Foundations of Computer Science,” pp. 320-322, IEEE Computer Society, Los Angeles, 1984.
4. M. O. BALL AND J. S. PROVAN, Calculating bounds on reachability and connectedness in stochastic networks, *Networks* **13** (1983), 253-278.
5. M. BLUM, R. M. KARP, O. VORNBERGER, C. H. PAPADIMITRIOU, AND M. YANNAKAKIS, The complexity of testing whether a graph is a superconcentrator, *Inform. Process. Lett.* **13** (1981), 164-167.
6. P. M. CAMERINI AND G. GALBIATI, The bounded path tree problem, *SIAM J. Algebraic and Discrete Methods* **3** (1982), 473-483.
7. P. M. CAMERINI, G. GALBIATI, AND F. MAFFIOLI, On the complexity of Steiner-like problems, in “Proceedings 17th Ann. Allerton Conf. on Communication, Control, and Computing,” pp. 969-977, Department of Electrical Engineering and the Coordinated Science Laboratory, University of Illinois, Urbana, Ill., 1979.

8. P. M. CAMERINI, G. GALBIATI, AND F. MAFFIOLI, Complexity of spanning tree problems: Part I, *European J. Op. Res.* **5** (1980), 346-352.
9. P. M. CAMERINI, G. GALBIATI, AND F. MAFFIOLI, On the complexity of finding multi-constrained spanning trees, *Disc. Applied Math.* **5** (1983), 39-50.
10. P. M. CAMERINI, G. GALBIATI, AND F. MAFFIOLI, The complexity of weighted multi-constrained spanning tree problems, *to appear in* "Proc. Colloq. on the Theory of Algorithms, Pécs 1984," János Bolyai Mathematical Society, Budapest, 1984.
11. P. M. CAMERINI AND F. MAFFIOLI, "Unlabelled partition systems: Optimization and complexity," Report No. LCE83-1, Dipartimento di Elettronica del Politecnico di Milano, Milan, Italy, 1983.
12. R. CHANDRASEKARAN AND A. TAMIR, Polynomial testing of the query 'Is $a^b \geq c^d$?' with application to finding a minimal cost reliability ratio spanning tree, *Disc. Applied Math.* **9** (1984), 117-123.
13. F. R. K. CHUNG, On concentrators, superconcentrators, generalizers, and nonblocking networks, *Bell Syst. Tech. J.* **58** (1979), 1765-1777.
14. C. DELORME AND G. FARHI, Large graphs with given degree and diameter — Part I, *IEEE Trans. Computers* **C-33** (1984), 857-860.
15. N. DEO, G. M. PRABHU, AND M. S. KRISHNAMOORTHY, Algorithms for generating fundamental cycles in a graph, *ACM Trans. Math. Software* **8** (1982), 26-42.
16. A. M. FARLEY, S. T. HEDETNIEMI, AND S. L. MITCHELL, Rectilinear Steiner trees in rectangle trees, *SIAM J. Algebraic and Discrete Methods* **1** (1980), 70-81.
17. H. FUJIWARA AND S. TOIDA, The complexity of fault detection problems for combinational logic circuits, *IEEE Trans. Computers* **C-31** (1982), 555-560.
18. O. GABBER AND Z. GALIL, Explicit construction of linear-sized superconcentrators, *J. Comput. System Sci.* **22** (1981), 407-420.
19. H. N. GABOW, Z. GALIL, AND T. H. SPENCER, Efficient implementation of graph algorithms using contraction, in "Proceedings 25th Ann. Symp. on Foundations of Computer Science," pp. 347-357, IEEE Computer Society, Los Angeles, 1984.
20. F. GAVRIL AND R. TAMARI, An algorithm for constructing edge-trees from hypergraphs, *Networks* **13** (1983), 377-388.
21. M. GRÖTSCHEL, L. LOVÁSZ, AND A. SCHRIJVER, The ellipsoid method and its consequences in combinatorial optimization, *Combinatorica* **1** (1981), 169-197.
22. J. N. HAGSTROM, Computing rooted communication reliability is #P-complete, manuscript (1981).
23. T. IBARAKI, T. KAMEDA, AND S. TOIDA, "Generation of minimal test sets for system diagnosis," Report No. 58-PM-031079, Department of Systems Design, University of Waterloo, Waterloo, Ont., 1979. Unseen paper, cited in [24].
24. T. IBARAKI, T. KAMEDA, AND S. TOIDA, On minimal test sets for locating single link failures in networks, *IEEE Trans. Computers* **C-30** (1981), 182-190.
25. T. IBARAKI, T. KAMEDA, AND S. TOIDA, Minimal test set for diagnosing a tree system, in "Studies on Graphs and Discrete Programming," pp. 215-240, P. Hansen (ed.), North-Holland, Amsterdam, 1981.
26. M. JERRUM, "On the Complexity of Evaluating Multivariate Polynomials," Report No. CST-11-81, Department of Computer Science, University of Edinburgh, Edinburgh, 1981.
27. D. S. JOHNSON AND H. O. POLLAK, Hypergraph planarity and the complexity of drawing Venn diagrams, manuscript (1984).
28. B. KRISHNAMURTHY AND S. B. AKERS, On the complexity of estimating the size of a test set, *IEEE Trans. Computers* **C-33** (1984), 750-753.
29. T. LENGAUER AND S. NÄHER, "Delay-independent switch level simulation of digital MOS circuits," Report No. 03/1984, Fachbereich 10, Universität des Saarlandes, Saarbrücken, West Germany.

30. N. MEGIDDO, On the complexity of the one-terminal network design problem, *Operations Res. Lett.* **1** (1982), 105-107.
31. C. H. PAPADIMITRIOU AND K. STEIGLITZ, "Combinatorial Optimization: Algorithms and Complexity," Prentice-Hall, Englewood Cliffs, N.J., 1982.
32. Y. PERL, Optimum split trees, *J. Algorithms* **5** (1984), 367-374.
33. Y. PERL AND Y. SHILOACH, Efficient optimization of monotonic functions on trees, *SIAM J. Algebraic and Discrete Methods* **4** (1983), 512-516.
34. J. PLESNÍK, The complexity of designing a network with minimum diameter, *Networks* **11** (1981), 77-85.
35. J. PLESNÍK, Complexity of decomposing graphs into factors with given diameters or radii, *Math. Slovaca* **32** (1982), 379-388.
36. J. PLESNÍK AND A. WAWRUCH, On the complexity of finding degree constrained subgraphs, *Acta Math. Universitatis Comenianae* **40** (1982), 215-218.
37. J. S. PROVAN, "The complexity of reliability computations in planar and acyclic graphs," Report No. UNC/ORSA/TR-83/12, Curriculum in Operations Research and Systems Analysis, University of North Carolina, Chapel Hill, N.C., revised version, 1984.
38. J. S. PROVAN AND M. O. BALL, The complexity of counting cuts and computing the probability that a graph is connected, *SIAM J. Comput.* **12** (1983), 777-788.
39. J. S. PROVAN AND M. O. BALL, Computing network reliability in time polynomial in the number of cuts, *Operations Res.* **32** (1984), 516-526.
40. V. RAMACHANDRAN, A linear-time algorithm for race detection in transistor switch-level circuits, in "Proc. IEEE International Conference on Computer Design: VLSI in Computers," pp. 345-348, IEEE Computer Society, Los Angeles, Calif., 1983.
41. V. RAMACHANDRAN, "A linear-time algorithm for switch-level simulation with race detection in a class of MOS VLSI circuits," Report No. ACT-48, Coordinated Science Laboratory, University of Illinois, Urbana, Ill., 1984.
42. A. SATYANARAYANA AND R. K. WOOD, "Polygon-to-chain reductions and network reliability," Report No. ORC 82-4, Operations Research Center, University of California, Berkeley, Calif., 1982. Unseen paper, cited in [2].
43. P. J. SLATER, A linear algorithm for the number of degree constrained subforests of a tree, *Inform. Process. Lett.* **15** (1982), 186-188.
44. M. M. SYSLO, On the computational complexity of the minimum-dummy-activities problem in a PERT network, *Networks* **14** (1984), 37-45.
45. J. L. SZWARCFITER, Optimal multiway search trees for variable size keys, *Acta Inform.* **21** (1984), 47-60.
46. L. G. VALIANT, On nonlinear lower bounds in computational complexity, in "Proceedings 7th Ann. ACM Symp. on Theory of Computing," pp. 45-53, Association for Computing Machinery, New York, 1975.
47. L. G. VALIANT, Graph-theoretic properties in computational complexity, *J. Comput. System Sci.* **13** (1976), 278-285.
48. L. G. VALIANT, The complexity of enumeration and reliability problems, *SIAM J. Comput.* **8** (1979), 410-421.
49. A. VERGIS, manuscript (1983).
50. E. ZEMEL, Polynomial algorithms for estimating network reliability, *Networks* **12** (1982), 439-452.