The NP-Completeness Column: An Ongoing Guide

DAVID S. JOHNSON

Bell Laboratories, Murray Hill, New Jersey 07974

This is the fourth edition of a quarterly column the purpose of which is to provide continuing coverage of new developments in the theory of NP-completeness. The presentation is modeled on that used by by M. R. Garey and myself in our book ''Computers and Intractability: A Guide to the Theory of NP-Completeness,'' W. H. Freeman & Co., San Francisco, 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, Bell Laboratories, Murray Hill, NJ 07974, including details, or at least sketches, of any new proofs (full papers are preferred). In the case of unpublished results, please state explicitly that you would like the results to be mentioned in the column.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

According to the prospectus presented in this column's first [Dec. 1981] edition, one of its functions is to provide ''occasional reports on new theoretical developments concerning NP-completeness and related issues.'' This month seems like an appropriate time to initiate this feature, as exciting developments have occurred recently in two separate areas. I shall devote the first half of the column to covering them.

The column is organized as follows. Section 2 discusses an extension of the work on relativized complexity covered in Chapter 7 of [G&J], which suggests a new approach to proving $P \neq NP$ (but doesn't, according to the latest word, suggest it very loudly). Section 3 surveys some surprising new approximation algo-

rithms for the ''bin packing'' problem (a problem discussed in detail in Chapter 6 of [G&J]) and uses these results to motivate this column's ''open problem of the month.'' We return to our list-making in Section 4, which surveys new results related to problems of permutations and orderings. Section 5 concludes with some brief ''updates.''

The promised discussion of routing problems will have to wait one more issue, by which time I hope to be able to capture the pictorial nature of these problems by using Chris Van Wyk's IDEAL program for typesetting figures [41].

2. P ≠ NP (ALMOST ALWAYS)

In Chapter 7 of [G&J] the concept of a Turing machine (TM) with an ''oracle'' is discussed. An *oracle* for a language A is a ''black box'' that, given a string x as input, will tell in one step whether or not $x \in A$. If we augment ordinary TM's with oracles for language A, we can ask all the standard questions of complexity theory, but now they are ''relativized to A'' and the answers may well depend on A. In particular, in [6] it was shown that there are languages B and C such that $P^B = NP^B$ and $P^C \neq NP^C$. Since all of the known techniques for distinguishing classes of languages yield the same results whether the TM's have oracles or not, this was taken as evidence that the standard techniques would be of little help in resolving the P versus NP question.

However, in a recent paper [7], Bennett and Gill suggest that there might be hope after all. They show that languages like language B above are exceedingly rare, in a rigorous, measure-theoretic sense. Note that there is a one-to-one correspondence between infinite sets of strings in $\{0,1\}^*$ and points in the unit interval [0,1): Lexicographically order the strings in $\{0,1\}^*$ as $x_1, x_2, \ldots$, and let each $p \in [0,1)$ correspond to the set that contains all x_i such that the $i+1st$ bit in the binary representation of p is 1. Thus we can ask what is the measure of the set of languages A for which $P^A = NP^A$, or equivalently, given a ''random'' oracle A, what is the probability that $P^A = NP^A$? Bennett and Gill prove that this probability is 0, in other words, $P^A \neq NP^A$, *almost always*.

Bennett and Gill then argue that this result provides strong evidence that P ≠ NP, since ''random oracles by their very structurelessness appear more benign and less likely to distort the relations among complexity classes than (ones) designed expressly to help or frustrate some class of computations.'' In particular, they propose the following *random oracle hypothesis*:

> Let S^A be an *acceptably relativized* statement (a rather complicated definition of ''acceptably relativized'' is used, one that is designed to avoid certain obvious pitfalls while still allowing all of our standard conjectures to be considered). Then the unrelativized statement S is true if and only if the relativized statement S^A is ''almost always'' true.

Given the results in [7], this hypothesis would imply not only P ≠ NP, but also

that LOGSPACE ≠ P, that NP ≠ co-NP ≠ PSPACE ≠ EXPTIME, and that P is identical to the random polynomial time class R of [1] and a number of its variants.

Thus we have a new way to resolve P versus NP and our other open problems concerning complexity classes: prove the random oracle hypothesis. Initially, this seemed promising, as it opened up the P versus NP question to attack by a whole new range of techniques. Unfortunately, it now appears that proving the hypothesis may well be significantly more difficult than proving P ≠ NP: the random oracle hypothesis, at least as formalized by Bennett and Gill, is false.

Kurtz [25] has provided two counterexamples, one of which we shall describe briefly. Let P^{3SAT} be the set of languages that can be recognized in polynomial time with the aid of an oracle for 3SAT. It is easy to see that NP $\subseteq P^{3SAT}$. However, if we relativize this statement to a random oracle A (the second class thus has *two* oracles), it becomes false with probability 1, even though it qualifies as ''acceptably relativized.'' Although there is some hope (not shared by Kurtz) that the hypothesis may yet be resurrected by an appropriate redefinition of ''acceptable relativization,'' none has yet been proposed. For now, at least, it appears that ''almost always'' is not nearly often enough.

3. Bin Packing and the Ellipsoid Algorithm

Recall that in the *bin packing* problem we are given a set $U = \{u_1, u_2, \ldots, u_n\}$ of *items*, each with a rational *size* $s(u_i) \in [0,1]$, and are asked to partition U into disjoint sets $U_1, U_2, \ldots, U_k$, called *bins*, such that the sum of the item sizes in each U_i is no more than 1, and such that k is as small as possible. This problem is, of course, NP-hard (transformation from 3-PARTITION), and so efforts have concentrated on ''approximation'' algorithms. Such an algorithm A, given an instance I, constructs a packing (partition) with a number $A(I)$ of bins *close to*, but not necessarily equal to, the optimum number OPT(I).

Since in most applications of interest the number of bins is large, the criterion for comparison between algorithms has been, from the first, the *asymptotic worst case ratio* R_A^∞, defined as follows.

$$R_A^\infty = \inf\left\{ r \geq 1 : \begin{array}{l} \textit{for some } N,\ \mathrm{A}(I)/\mathrm{OPT}(I) \leq r \textit{ for all} \\ \textit{instances } I \textit{ satisfying } \mathrm{OPT}(I) \geq N \end{array} \right\}$$

A brief summary of the highlights of the past decade of bin packing goes as follows. A first, easy, observation was that the straightforward, linear time algorithm NEXT FIT (NF) satisfied $R_{NF}^\infty = 2$ [21]. Garey, Graham, and Ullman [13] showed that the slightly more sophisticated $O(n \log n)$ time FIRST FIT (FF) algorithm yielded $R_{FF}^\infty = 1.7$, and proved that the next step in sophistication, FIRST FIT DECREASING, obeyed $R_{FFD}^\infty \leq 1.25$. Johnson (yours truly) then observed

that FFD was also $O(n\log n)$ and spent 100 pages of his thesis [20] (see also [22]) proving the exact result $R^{\infty}_{FFD} = 11/9 = 1.222...$ Progress then stalled from 1973 until 1978, when A. Yao finally succeeded in breaking the 11/9 barrier, devising an $O(n^{10}\log n)$ algorithm A with $R^{\infty}_{A} \leq (11/9) - (1/10{,}000{,}000)$ [44]. Thus inspired, other bin packers returned to the fray, with Friesen and Langston [11] developing an $O(n\log n)$ hybrid algorithm having $R^{\infty}_{A} \leq 6/5 = 1.20$ and Garey and Johnson [14] devising an $O(n\log n)$ modified FFD having $R^{\infty}_{A} = 71/60 = 1.18333...$

This process of incremental improvement in R^{∞}_{A} has now come to a surprisingly abrupt conclusion. Two researchers, Lueker and Fernandez de la Vega, have independently discovered that for every ε there is a linear time algorithm A[ε] with $R^{\infty}_{A[\varepsilon]} \leq 1 + \varepsilon$. Algorithms with this sort of behavior had long been known for problems such as the knapsack problem, where performance is measured by *absolute* (rather than asymptotic) worst case ratios, with the term *approximation scheme* being used to describe the overall collection of algorithms $\{A[\varepsilon]: \varepsilon > 0\}$. Lueker and Fernandez de la Vega discovered that, counter to expectations, techniques from the knapsack approximation scheme *could* be adapted to bin packing, so long as one was prepared to solve a very large, but (for each ε) fixed-size, linear program. The details of this ''asymptotic approximation scheme'' are discussed in a joint paper [10].

A few of these details may limit the applicability of the algorithms: the precise performance guarantee provided is

$$A[\varepsilon](I) \leq (1 + \varepsilon)\cdot OPT(I) + (1/\varepsilon)^2,$$

and the running time, although linear in n, is exponential in $(1/\varepsilon)^2$. For $\varepsilon = 1/10$, the additive constant in the guarantee would be about 100 bins, and the algorithm would take on the order of 10^{20} steps. One can, by a standard trick, move the 10^{20} term out of the running time and into the additive constant, but that would not be much of an improvement.

What one would really like is an asymptotic approximation scheme that simultaneously provides a guarantee of the form $(1 + \varepsilon)\cdot OPT(I) + p_1(1/\varepsilon)$ and a running time of $p_2(n,1/\varepsilon)$, where p_1 and p_2 are both polynomials. One's first thought is to try to speed up the algorithms of Lueker and Fernandez de la Vega by using the now-famous ''ellipsoid algorithm'' for linear programming. There is a bottleneck, however. The linear program P_{ε} that has to be solved in the algorithm A[ε] has a number of variables that is exponential in $1/\varepsilon$. Even the ellipsoid algorithm will have trouble with that. Karp and Karmarkar [23], however, do not. Using an imposing arsenal of techniques from mathematical programming and complexity theory, they have very recently succeeded in constructing the desired ''fully polynomial'' asymptotic approximation scheme.

Their first step is to note that the dual of P_{ε} has only a polynomial number of variables (although an exponential number of constraints). They therefore can make use of the fact that, in the ellipsoid algorithm, one doesn't need to know

all the constraints, but only to be able to find a violated constraint when there is one [17].

This approach is applicable here because, using an old idea due to Gilmore and Gomory [15], a violated constraint can be generated by solving a knapsack problem. The knapsack problem is, of course, NP-complete. However, it can be solved approximately using the above-mentioned knapsack approximation scheme, which turns out to be good enough if one only wants an approximate solution to the dual. An approximate solution to the dual is good enough, if one only wants a lower bound on the optimal solution of P_ε. This is good enough, because with it we can obtain, in polynomial time, an approximate lower bound on OPT(I). This lower-bounding procedure can then be used as a subroutine in an iterative process for building up a near-optimal packing, much as one can use a procedure for computing the *exact* value of OPT(I) as a subroutine in building up an optimal packing.

There are, of course, many more details than this brief summary can convey, and the algorithm is quite a *tour-de-force*. However, although these approximation scheme results are generating much excitement among the packers of theoretical bins, one should stress that, as of yet, none of the schemes has much pretension toward practicality. Nevertheless, as long as we are following the path of better and better guarantees for less and less practical algorithms, there is one more step to be taken. So far we have found that, for every $\varepsilon > 0$, there is a polynomial time algorithm with $R_A^\infty \leq 1 + \varepsilon$. What about $\varepsilon = 0$, i.e., is there a polynomial time algorithm A with $R_A^\infty = 1$?

The answer is yes. Indeed, such algorithms are easily derived from either of the above schemes, merely by making an appropriate choice of ε as a function of the instance at hand (actually, as a function of $\sum_{i=1}^{n} s(u_i)$, a sum that is always within a factor of 2 of OPT(I)). The difference between the two schemes shows up in the additive terms of the guarantees provided. These unfortunately are no longer constants, although they *do* remain $o(\mathrm{OPT}(I))$. For Lueker and Fernandez de la Vega we have

$$\mathrm{A}(I) \leq \mathrm{OPT}(I) + O\left(\frac{\mathrm{OPT}(I)\log\log[\mathrm{OPT}(I)]}{\log[\mathrm{OPT}(I)]}\right)$$

whereas for Karp and Karmarkar we have

$$\mathrm{A}(I) \leq \mathrm{OPT}(I) + O(\mathrm{OPT}(I)^{1-\delta})$$

for some $\delta > 0$. With both algorithms there is a trade-off between running time and the asymptotic growth rate of the additive term, but no way of balancing the two to yield an algorithm that would be useful for instances of less than astronomical size. A much more attractive additive term would be something like $\log[\mathrm{OPT}(I)]$, or better yet, a constant, independent of OPT(I). However, Karp and

Karmarkar seem to have pushed the state of the art about as far as it will go. Thus here is perhaps where one should begin to investigate the possibility of ''impossibility'' results, e.g., the possibility of proving that unless P = NP no polynomial time approximation algorithm A can guarantee that $A(I) - OPT(I)$ is less than some given bound.

To date there has been little success in this direction. Still open is the weakest possible such result, which can hence serve as our ''open problem of the month.'' Prove or disprove: Assuming that P ≠ NP, there can be no polynomial time approximation algorithm A that guarantees $A(I) \leq OPT(I) + 1$.

4. Problems of Orderings and Permutations

The most famous problems involving permutations are probably the TRAVELING SALESMAN problem [ND22] and its special case, the HAMILTONIAN CIRCUIT problem [GT37]. A number of new developments concerning the latter problem have come to my attention since I discussed it in the [Mar. 1982] column (where it revealed its multifaceted nature by masquerading as an *embedding* problem).

Akiyama, Nishizeki, and Saito [4] have continued the process of finding ever-more restricted classes of graphs for which HAMILTONIAN CIRCUIT is NP-complete, showing that NP-completeness continues to hold both for 2-connected cubic bipartite planar graphs and for 3-connected cubic bipartite (not necessarily planar) graphs. Wigderson [43] has shown that NP-completeness also holds for *maximal* planar graphs, a result that implies NP-completeness for the following interesting variant: ''Given a planar graph G, can we add new edges (but not new vertices) so as to render the resulting graph Hamiltonian, while still keeping the graph planar?'' There is a more positive development for the *directed* HAMILTONIAN CIRCUIT problem: Ramanath [38] has shown that this problem (as well as the directed HAMILTONIAN PATH problem) can be solved in polynomial time when restricted to ''reducible flowgraphs'' (as defined, for instance, in [2], pp. 938-941).

There have also been a number of interesting developments related to optimization version of TRAVELING SALESMAN, which we shall refer to as the ''TSP'' in what follows. Consider the version of the TSP in which the cities correspond to points in the plane. In applying the standard approximation algorithms (see [G&J], Chapter 6), one can obtain tours in which the path intersects itself, and hence can clearly be shortened by an appropriate interchange. Van Leeuwen and Schoone have recently shown that all such crossings can be removed in polynomial time [40], a result that is not entirely obvious since removing one crossing may create new ones. A second way of improving the tours found by the heuristics is by optimizing the choice of ''shortcuts'' taken when an Euler tour is reduced to a Hamiltonian one. Here we are not so fortunate. Papadimitriou and Vazirani [33] have shown that, even for the Euler tours gen-

erated by Christofides' algorithm, this optimization problem is NP-hard.

On a more theoretical plane, Papadimitriou [32] has shown that the problem ''Given an instance of the TSP, does it have more than one optimal solution?'' is complete for the class P^{3SAT} mentioned in Section 2 (i.e., the class Δ_2^p in the polynomial hierarchy). A future column will report on work of Papadimitriou and Yannakakis [34] extending earlier results on the complexity of the ''TSP polytope.''

Another famous problem that can be viewed as a permutation problem is GRAPH ISOMORPHISM: is there a permutation of the vertices of G that makes it identical to H? This problem remains open, but some interesting variants have been shown to be NP-complete.

[1] GRAPH ISOMORPHISM WITH RESTRICTIONS

INSTANCE: Two graphs $G = (V,E)$ and $H = (U,F)$, and a set $R \subseteq V \times U$.

QUESTION: Is there an isomorphism f of G to H that, when viewed as a set of ordered pairs, contains no pair from R?

Reference. Lubiw [29]. Transformation from 3SAT.

Comment. Remains NP-complete when $H = G$ (AUTOMORPHISM WITH RESTRICTIONS). In the latter case, remains NP-complete even if $R = \{(v,v): v \in V\}$ (FIXED-POINT-FREE AUTOMORPHISM), and this problem remains NP-complete even if we ask for an order 2 fixed-point-free automorphism. Note, however, that the order 2 fixed-point-free automorphism problem is solvable in polynomial time for groups. AUTOMORPHISM WITH RESTRICTIONS remains NP-complete so long as $|R| \geq \varepsilon n^{1/k}$ for fixed ε and k. However, if $|R| = 1$, this problem becomes Turing-equivalent to the still open GRAPH ISOMORPHISM problem. It is not known whether the ordinary GRAPH AUTOMORPHISM problem ($|R| = 0$) is as hard as GRAPH ISOMORPHISM, although the problem of counting automorphisms is equivalent to that of counting isomorphisms (and both are equivalent to GRAPH ISOMORPHISM itself) [30].

[2] PARTITIONED GRAPH ISOMORPHISM

INSTANCE: Two graphs $G = (V,E)$ and $H = (U,F)$, and a positive integer K.

QUESTION: Are there partitions $E = E_1 \cup \cdots \cup E_K$ and $F = F_1 \cup \cdots \cup F_K$ of E and F into disjoint (and possibly empty) sets such that $G_i = (V,E_i)$ is isomorphic to $H_i = (U,F_i)$, $1 \leq i \leq K$?

Reference. F. Yao [45]. Transformation from EXACT COVER BY 3-SETS

(X3C).

Comment. Remains NP-complete for $K = 2$; equivalent to GRAPH ISOMORPHISM for $K = 1$. The variant in which we are just given G, and are asked whether there is a partition $E = E_1 \cup E_2$ such that the two graphs $G_1 = (V,E_1)$ and $G_2 = (V,E_2)$ are isomorphic, is NP-complete even for trees [16], although for trees it can be solved in polynomial time if E_1 and E_2 are required to be connected [18].

To conclude our discussion of vertex permutation problems, let us consider two related problems that were mentioned in the [Mar. 1982] column: BANDWIDTH [GT40] and MINIMUM CUT LINEAR ARRANGEMENT [GT44]. New special-case algorithms have been found for both of these problems. Assman, Peck, Syslo, and Zak [5] have shown that BANDWIDTH can be solved in time $O(n\log n)$ for ''caterpillars with hairs of length 1 and 2.'' Chung, Makedon, Sudborough, and Turner [8] have shown that MINIMUM CUT LINEAR ARRANGEMENT can be solved in time $O(n(\log n)^{d-1})$ for trees with maximum vertex degree d. This yields polynomial time algorithms for any fixed value of d, in contrast to the case for BANDWIDTH, which is known to be NP-complete even for trees of maximum degree 3 [12]. Some new complexity results for *variants* on BANDWIDTH are collected together in the following entry.

[3] CYCLIC BANDWIDTH

INSTANCE: Graph $G = (V,E)$, positive integer $K \leq |V|$.

QUESTION: Is there a cyclic ordering of V with bandwidth K or less, i.e., is there a one-to-one function $f : V \rightarrow \{1,2,\ldots,|V|\}$ such that, for all $\{u,v\} \in E$, either $|f(u) - f(v)| \leq K$ or $(|V| - |f(u) - f(v)|) \leq K$?

Reference. Leung, Vornberger, and Witthoff [27]. Transformation from 3-PARTITION.

Comment. Unlike ordinary BANDWIDTH, which can be solved in polynomial time for any fixed K [39], this problem remains NP-complete for $K = 2$ (it is trivial for $K = 1$). It *can*, however, be solved in polynomial time for any fixed K if G is required to be connected. The ''separation'' variant, in which both distances above must *exceed* K, is NP-complete for $K = 1$ *and* connected G, as is the corresponding variant of ordinary BANDWIDTH. For DIRECTED BANDWIDTH [GT41], the ''separation'' variant is NP-complete for arbitrary K, but solvable for in-forests or out-forests, for interval orders, and for $K = 1$ [27].

We turn now to a problem involving the ordering of edges, rather than vertices.

[4] SEARCH NUMBER

INSTANCE: Graph $G = (V,E)$, positive integer $K \leq |V|$.

QUESTION: Does G have a *search number* of K or less, i.e., can K *searchers* ''clear'' the graph according to the following rules:

(a) A *move* consists of placing a searcher on a vertex, removing a searcher from a vertex, or moving a searcher along an edge. A vertex is *guarded* if it contains a searcher.

(b) An edge is *clear* if it has been ''cleared'' and there is no unguarded path from it to an unclear edge. Initially, all edges are unclear.

(c) If $\{u,v\}$ is an edge and u is guarded, then $\{u,v\}$ can be cleared by placing a second searcher at u and then moving it along the edge to v. If $\{u,v\}$ is the only unclear edge with u as endpoint, the second searcher is not needed and we may clear the edge by moving the u's guard from u to v along it.

Reference. Megiddo, Hakimi, Garey, Johnson, Papadimitriou [31]. Transformation from MINIMUM CUT INTO EQUAL-SIZED SUBSETS (see comments to [ND17]).

Comment. This problem was first studied by Parsons [35,36]. The fact that the problem is really about edge permutations (and hence is in NP), is due to a recent result of LaPaugh [26], who shows that if the search number of G is K, then G can be cleared by K searchers in such a way that no edge is ever ''recontaminated.'' In other words, no guard is ever removed from a vertex if its absence will open up an unguarded path from an uncleared edge to a cleared one. Thus any search strategy determines, and is determined by, a permutation of the edges, and the existence of the desired strategy can be verified in nondeterministic polynomial time by guessing this permutation. The problem can be solved in *deterministic* polynomial time if $K \leq 3$ or if G is a tree [31]. There is an interesting connection between SEARCH NUMBER and MINIMUM CUT LINEAR ARRANGEMENT: for any graph G the search number is no greater than cutwidth (where ''cutwidth'' is the quantity minimized in MINIMUM CUT LINEAR ARRANGEMENT). The quantities are in fact equal for trees of maximum degree 3 [8], although not in general.

Our final three problems concern the generation of permutations.

[5] MINIMUM LENGTH GENERATOR SEQUENCE

INSTANCE: A set $\{g_i : 1 \leq i \leq k\}$ of generators of a permutation group G, a target permutation $P \in G$, and a positive integer K.

QUESTION: Is there a sequence $g_{i_1}, g_{i_2}, \ldots, g_{i_j}$, $j \leq K$, such that $P = g_{i_1} g_{i_2} \cdots g_{i_j}$?

Reference. Even and Goldreich [9]. Transformation from X3C.

Comment. Also NP-hard is the problem, given a set of generators and an integer K, of determining whether *all* permutations in G can be generated by generator sequences of length K or less.

[6] SHUFFLED STRING

INSTANCE: Finite alphabet Σ, strings w and $w_1, w_2, \ldots, w_n$ in Σ^*.

QUESTION: Is the string w in the *shuffle* of $w_1, w_2, \ldots, w_n$, i.e., is it of the form $w_1[1]w_2[1]...w_n[1]w_1[2]w_2[2]...w_n[2]...w_1[k]w_2[k]...w_n[k]$, where $w_i[1]w_i[2]...w_i[k]$ is a partition of w_i into a sequence of (possibly empty) substrings, $1 \le i \le n$?

Reference. Warmuth and Haussler [42]. Transformation from 3-PARTITION.

Comment. Remains NP-complete even if $|\Sigma| = 2$. If $|\Sigma| = 3$, remains NP-complete even if $w_1 = w_2 = \cdots = w_n$, in which case we are asking whether w is in the *iterated shuffle* of w_1. If we extend the definition of iterated shuffle to languages in the standard way, there exist fixed context free languages L such that membership in the iterated shuffle of L is NP-complete. For any fixed *regular* language L, however, the problem is solvable in polynomial time. Reference [42] also contains a number of other results along this line, asking when NP-complete languages can be generated by combining the shuffle and iterated shuffle operators with the standard ones for generating languages, i.e., $\cup$, $\cdot$, *, etc. For more on theoretical aspects of the shuffle, see [19].

[7] DEFECTIVE SORTING NETWORK

INSTANCE: Sequence $x_1, x_2, \ldots, x_n$ of *input variables* and a *program* P consisting of a sequence of statements of the form ''if $x_i \ge x_j$, then interchange their values,'' where $i < j$. (Note that there is a one-to-one correspondence between such programs and ''comparator networks'' (as described in [24], Section 5.3.4), so long as no comparators are allowed to operate in parallel).

QUESTION: Is there an input to program P which P fails to sort into nondecreasing order, i.e., are there integer input values $x_i = z_i$, $1 \le i \le n$, and an index k, $1 \le k < n$, such that, after program P has been executed, $x_k > x_{k+1}$?

Reference. Rabin [37]. Transformation from 3-DIMENSIONAL MATCHING.

Comment. Can be solved in polynomial time if all statements are of the form ''if $x_i \ge x_{i+1}$, then interchange their values,'' i.e., if all comparators link adjacent lines in the network (see [24], Exercise 5.3.4-36c). Sorting networks are in

vogue again, what with the recent claim by Ajtai, Komlós, and Szemerédi [3] of a network with $O(n\log n)$ comparators and $O(\log n)$ delay time, both bounds offering an improvement by a factor of $\log n$ over bounds that had stood for some 18 years.

5. UPDATES

Since this column is in danger of overflowing its page allotment, I will limit myself to some minor items that can be disposed of quickly. In the last column [Jun. 1982], the ''to appear'' citation on reference [31] should be moved to reference [32], and the first occurrence of the word ''NP-complete'' in the comments to Problem 13 should be replaced by ''NP-hard.'' Also, the claim that 11-colorability of Steiner triple systems is NP-complete turns out to have been based on a misreading of that column's reference [11]. The appropriate conclusion is that 14-colorability is NP-complete. Our final comment concerns the source problem in many of the transformations reported in the last two columns. The proof of the NP-completeness of PLANAR 3-SAT has at last made its formal appearance [28], after four years as an ''unpublished manuscript.''

REFERENCES

1. L. ADLEMAN AND K. MANDERS, Reducibility, randomness, and intractability, *in* ''Proceedings 9th Ann. ACM Symp. on Theory of Computing,'' pp. 151-153, Association for Computing Machinery, New York, 1977.
2. A. V. AHO AND J. D. ULLMAN, ''The Theory of Parsing, Translation, and Compiling, Vol. II: Compiling,'' Prentice-Hall, Englewood Cliffs, N. J., 1972.
3. M. AJTAI, J. KOMLÓS, AND E. SZEMERÉDI, in preparation, (1982).
4. T. AKIYAMA, T. NISHIZEKI, AND N. SAITO, NP-completeness of the Hamiltonian cycle problem for bipartite graphs, *Journal of Information Processing* **3** (1980), 73-76.
5. S. F. ASSMAN, G. W. PECK, M. M. SYSLO, AND J. ZAK, The bandwidth of caterpillars with hairs of length 1 and 2, *SIAM J. Algebraic and Discrete Methods* **2** (1981), 387-393.
6. T. BAKER, J. GILL, AND R. SOLOVAY, Relativizations of the P =? NP question, *SIAM J. Comput.* **4** (1975), 431-442.
7. C. H. BENNETT AND J. GILL, Relative to a random oracle A, $P^A \neq NP^A \neq co-NP^A$ with probability 1, *SIAM J. Comput.* **10** (1981), 96-113.
8. M-J. CHUNG, F MAKEDON, I. H. SUDBOROUGH, AND J. TURNER, Polynomial algorithms for the min cut problem on degree restricted trees, manuscript (1982).
9. S. EVEN AND O. GOLDREICH, The minimum-length generator sequence problem is NP-hard, *J. Algorithms* **2** (1981), 311-313.
10. W. FERNANDEZ DE LA VEGA AND G. S. LUEKER, Bin packing can be solved within $1 + \varepsilon$ in linear time, *Combinatorica* **1** (1981), 349-355.
11. D. K. FRIESEN AND M. A. LANGSTON, Analysis of a compound bin packing algorithm, manuscript (1981).
12. M. R. GAREY, R. L. GRAHAM, D. S. JOHNSON, AND D. E. KNUTH, Complexity results for bandwidth minimization, *SIAM J. Appl. Math.* **34** (1978), 835-859.

13. M. R. GAREY, R. L. GRAHAM, AND J. D. ULLMAN, Worst-case analysis of memory allocation algorithms, *in* ''Proceedings 4th Ann. ACM Symp. on Theory of Computing,'' pp. 143-150, Association for Computing Machinery, New York, 1972.
14. M. R. GAREY AND D. S. JOHNSON, A 71/60 algorithm for one-dimensional bin packing, manuscript (1980).
15. P. C. GILMORE AND R. E. GOMORY, A linear programming approach to the cutting-stock problem, *Operations Res.* **9** (1961), 849-859.
16. R. L. GRAHAM AND R. W. ROBINSON, private communication (1982).
17. M. GRÖTSCHEL, L. LOVÁSZ, AND A. SCHRIJVER, The ellipsoid method and its consequences in combinatorial optimization, *Combinatorica* **1** (1981), 169-198.
18. F. HARARY AND R. W. ROBINSON, Isomorphic factorizations VIII: Bisectable trees, manuscript (1981).
19. M. JANTZEN, ''Eigenschaften von Petrinetzsprachen,'' Report No. IFI-HH-B-64, Fachbereich Informatik, Universität Hamburg, Hamburg, 1979.
20. D. S. JOHNSON, ''Near-optimal bin packing algorithms,'' Report No. MAC TR-109, Project MAC, Massachusetts Institute of Technology, Cambridge, Mass., 1973.
21. D. S. JOHNSON, Fast algorithms for bin packing, *J. Comput. System Sci.* **8** (1974), 272-314.
22. D. S. JOHNSON, A. DEMERS, J. D. ULLMAN, M. R. GAREY, AND R. L. GRAHAM, Worst-case performance bounds for simple one-dimensional packing algorithms, *SIAM J. Comput.* **3** (1974), 299-325.
23. R. M. KARP AND N. KARMARKAR, An efficient approximation scheme for the one-dimensional bin packing problem, *in* ''Proceedings 23rd Ann. Symp. on Foundations of Computer Science,'' IEEE Computer Society, Long Beach, Calif., 1982 (to appear).
24. D. E. KNUTH, ''The Art of Computer Programming, Vol. 3: Sorting and Searching,'' Addison-Wesley Publishing Company, Inc., Reading, Mass., 1973.
25. S. L. KURTZ, On the random oracle hypothesis, *in* ''Proceedings 14th Ann. ACM Symp. on Theory of Computing,'' pp. 224-233, Association for Computing Machinery, New York, 1982.
26. A. S. LAPAUGH, Recontamination does not help, manuscript (1982).
27. J. Y-T. LEUNG, O. VORNBERGER, AND J. D. WITTHOFF, On some variants of the bandwidth minimization problem, manuscript (1982).
28. D. LICHTENSTEIN, Planar formulae and their uses, *SIAM J. Comput.* **11** (1982), 329-343.
29. A. LUBIW, Some NP-complete problems similar to graph isomorphism, *SIAM J. Comput.* **10** (1981), 11-21.
30. R. MATHON, A note on the graph isomorphism counting problem, *Inform. Process. Lett.* **8** (1979), 131-132.
31. N. MEGIDDO, S. L. HAKIMI, M. R. GAREY, D. S. JOHNSON, AND C. H. PAPADIMITRIOU, The complexity of searching a graph (preliminary version), *in* ''Proceedings 22nd Ann. Symp. on Foundations of Computer Science,'' pp. 376-385, IEEE Computer Society, Los Angeles, 1981.
32. C. H. PAPADIMITRIOU, On the complexity of unique solutions, *in* ''Proceedings 23rd Ann. Symp. on Foundations of Computer Science,'' IEEE Computer Society, Long Beach, Calif., 1982 (to appear).
33. C. H. PAPADIMITRIOU AND U. V. VAZIRANI, On two geometric problems related to the travelling salesman problem, manuscript (1981).
34. C. H. PAPADIMITRIOU AND M. YANNAKAKIS, The complexity of facets (and some facets of complexity), *in* ''Proceedings 14th Ann. ACM Symp. on Theory of Computing,'' pp. 255-260, Association for Computing Machinery, New York, 1982.
35. T. D. PARSONS, Pursuit-evasion in a graph, *in* ''Theory and Application of Graphs,'' (Y. Alavi and D. R. Lick, eds.) pp. 426-441, Springer-Verlag, Berlin, 1976.
36. T. D. PARSONS, The search number of a connected graph, *in* ''Proceedings 9th Southeastern Conference on Combinatorics, Graph Theory, and Computing,'' pp. 549-554, Utilitas Mathematica Publishing, Winnipeg, Ont., 1978.

37. M. O. RABIN, private communication (1981).
38. M. V. S. RAMANATH, The Hamiltonian path and cycle problems are P-time solvable for reducible flow graphs, manuscript (1982).
39. J. B. SAXE, Dynamic-programming algorithms for recognizing small-bandwidth graphs in polynomial time, *SIAM J. Algebraic and Discrete Methods* **1** (1980), 363-369.
40. J. VAN LEEUWEN AND A. A. SCHOONE, ''Untangling a traveling salesman tour in the plane,'' Report No. RUU-CS-80-11, Department of Computer Science, University of Utrecht, the Netherlands, 1980.
41. C. J. VAN WYCK, A graphics typesetting language, *SIGPLAN Notices* **16** (1981), 99-107.
42. M. K. WARMUTH AND D. HAUSSLER, ''On the complexity of iterated shuffle,'' Report No. CU-CS-201-81, Department of Computer Science, University of Colorado, Boulder, Colo., 1981.
43. A. WIGDERSON, ''The complexity of the Hamiltonian circuit problem for maximal planar graphs,'' Report No. 298, Electrical Engineering and Computer Science Department, Princeton University, Princeton, N.J., 1982.
44. A. C. YAO, New algorithms for bin packing, *J. Assoc. Comput. Mach.* **27** (1980), 207-227.
45. F. F. YAO, Graph 2-isomorphism is NP-complete, *Inform. Process. Lett.* **9** (1979), 68-72.