

The NP-Completeness Column: An Ongoing Guide

DAVID S. JOHNSON

Bell Laboratories, Murray Hill, New Jersey 07974

This is the sixth 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 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 In the case of unpublished results, please state explicitly that you would like the results to be mentioned in the column.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

This month's column will be a potpourri, inspired in part by the recent 23rd Annual Symposium on Foundations of Computer Science (FOCS-82), held November 3-5 in Chicago. Of the 46 papers presented at that conference, over a third are relevant to the general subject of NP-completeness. These cover topics running the gamut from language theory to robotics, and in doing so reflect the wide variety of fields to which the theory of NP-completeness is currently being applied. To illustrate this variety, I shall draw both on the FOCS-82 Proceedings and on a broader than usual range of other publications.

The column is organized as follows. Section 2 concerns an attempt to use "applied NP-completeness" in the field of cryptography, and what a new result,

presented at FOCS-82, has to say about it. Section 3 is devoted to recent theoretical results concerning the complexity of algorithms for LINEAR PROGRAMMING. There are many interesting avenues to be pursued, even though the “big” question of whether the problem can be solved in polynomial time has now been resolved. We turn to new NP-hard problems in Section 4, presenting an abbreviated run of the above gamut and concluding with a graceful new “Open Problem of the Month.”

2. A CRACKED KNAPSACK

Usually, the remark that a problem is NP-complete is prefaced with an adverb like “unfortunately.” However, there is one area where the existence of a polynomial time algorithm may occasionally deserve that same qualifier: the design of secure cryptographic systems. In recent years, a number of “public key” cryptosystems have staked their claim to security on the fact that polynomial time algorithms do not exist for certain problems (or are at least unlikely). One of the first such systems was proposed in 1978 by Merkle and Hellman [39], and involved the NP-complete KNAPSACK problem (actually the special case known as SUBSET SUM [SP13]).

An instance of the SUBSET SUM problem is a list a_1, a_2, \dots, a_n of positive integers together with a target integer B . It is known to be NP-hard to determine whether there is some subset of the a_i 's that sums exactly to B , and if so, to generate such a subset. In the Merkle-Hellman cryptosystem, the a_i 's constitute a “public key,” available to all. Anyone wishing to send an n -bit message X encodes it by summing up the a_i 's corresponding to the non-zero bits of X , obtaining an integer B_X . In order to decode such a message, i.e., obtain X from B_X , one must solve an instance of the NP-hard SUBSET SUM problem. This suggests that the system is, in a sense, guaranteed to be secure (assuming $P \neq NP$).

Unfortunately, as described so far, this system makes the deciphering job just as hard for the intended recipient as for any would-be eavesdropper. This difficulty is avoided in a public key cryptosystem because the intended recipient is provided with some “trapdoor information,” known only to him, which allows for an easy solution of the decoding problem. In the Merkle-Hellman scheme, this information consists of a pair $w < M$ of integers such that w is relatively prime to M , and such that the integers $b_i \equiv wa_i \pmod{M}$ form a superincreasing sequence (i.e., $b_{j+1} > \sum_{i=1}^j b_i$, $1 \leq j < n$), with $M > \sum_{i=1}^n b_i$. It is an easy exercise to show that, in this case, X will be the unique solution to the SUBSET SUM problem for the b_i 's and the target integer $wB_X \pmod{M}$, and that it can be found quickly using a “greedy” algorithm.

There is still a difficulty, however. Not all lists a_1, \dots, a_n have a w and M that will turn them into the desired superincreasing sequence. Thus the way the Merkle-Hellman scheme works is as follows: Our intended recipient randomly

chooses a superincreasing sequence, along with an appropriate w and M , determines a U such that $Uw \equiv 1 \pmod{M}$ (such a U must exist since w is relatively prime to M), and then publishes (as his public key) the integers $\{Ub_i \pmod{M}\}$.

Note, however, that a potentially crucial qualification has now been placed on our claims of security for the system. Instead of facing the general (NP-hard) SUBSET SUM problem, a would-be eavesdropper now faces only a special case: those instances that can be generated from superincreasing sequences in the specified manner. Although the general SUBSET SUM problem is NP-hard, this special case need not be. Thus the theory of NP-completeness, although often dragged into discussions of this and related systems, really has little to say about its conjectured security. Nevertheless, Merkle and Hellman were still willing to bet their scheme could not be cracked.

As a consequence, Adi Shamir is now \$100 richer. In a paper presented at FOCS-82 [49], he has shown that the above-mentioned special case of SUBSET SUM can be solved in polynomial time "almost always." As one might expect, the basic idea is to look for the pair (w, M) , a task that at first might seem like looking for a needle in a haystack. It turns out, however, that the haystack is usually full of needles: if there is one pair (w, M) , there is a sub-interval of $[0, 1]$ in which *every* rational point w'/M' yields a pair that will serve just as well for decoding purposes as the original. The algorithm works by finding a collection of intervals, one of which must have the desired properties. Although there is a slight chance that the size of this collection may get too large to handle, for any fixed bound $\epsilon > 0$, one can adjust the parameters of the algorithm so that it runs in polynomial time with a probability $1 - \epsilon$ of success.

The algorithm works by solving an integer programming problem, and the most important of the above parameters is the number k of variables in the integer programming instance to be solved. Shamir proposes using H. W. Lenstra, Jr.'s algorithm [33], which has been proved polynomial for any fixed k . Shamir argues that $k = 4$ is sufficient to reduce the probability of failure to $1/250,000$ when $n = 100$. There still may be practical drawbacks, however. In Lenstra's best running time bound for his algorithm [33], the *degree* of the polynomial is an exponential function of k .

Adleman [2] and others have suggested an alternative approach that appears to speed things up considerably. The idea is to use instead an algorithm of Lenstra, Lenstra, and Lovasz [32] for finding a reduced basis in a lattice. This idea has also been used by Kannan [25] to speed up Lenstra's original algorithm, but here it is applied directly to the cryptography problem. It also offers hope of cracking more sophisticated variants on the Merkle-Hellman scheme, such as the multiply-encrypted knapsack, in which the public key is obtained from a super-increasing sequence by a *series* of modular multiplications, rather than just one. (Another variant to which it applies is the "Graham-Shamir" Knapsack [31,38], so in a sense Shamir's efforts in code-breaking have backfired on him.)

Another way of making a knapsack cryptosystem harder to break is simply to use more bits in the original a_i 's. Shamir's result assumes that each a_i has at most $2n$ bits, meaning that one needs to transmit roughly $2n$ bits for an n -bit message. Lagarias [30] has shown that Shamir's approach will work for any fixed bound on the number of bits per a_i ; however, increasing the number of bits has a substantial (negative) effect on the tradeoff between running time and probability of failure. Note, however, that large amounts of computation time may be feasible for would-be code-breakers, since they can begin work as soon as the public key is published, and need not wait for a message to be sent. Moreover, the ability to compromise a small but significant fraction of the private keys is probably enough to render a cryptosystem unusable.

There do remain some as-yet-uncompromised public key cryptosystems, however. The most famous is the "RSA" scheme [45], which is justified, not by the NP-completeness of SUBSET SUM, but instead by the presumed difficulty of factoring integers. The currently best algorithms known for factoring an integer n are probabilistic ones with expected running times of the form $e^{c\sqrt{\log n \log \log n}}$ [14] (and deterministic ones that run within similar time bounds if certain number theoretic conjectures hold [40,42]). Such a bound is better than exponential, but still not practical for, say, 500-bit numbers. The RSA scheme is based on the assumption that an algorithm that can decipher messages in polynomial time, given only the public key K , would imply the existence of a polynomial time algorithm for factoring K . This implication has actually been proved for a pair of more complicated variants on the scheme due to Rabin [43] and Williams [54]. There is a sense, however, in which factoring K is a special case of the general factoring problem, since in all these schemes K is the product of two primes rather than a general composite number. To date this has not been much help, since the special case in question seems to be the *hardest* case for factoring; however, a paper by Adleman and McDonnell at FOCS-82 [3] raises, at least by analogy, the possibility that it just might be the other way around.

As an addendum, one might note the very real progress that has recently been made on a different sort of factoring problem: the factoring of *polynomials* rather than integers. Lenstra, Lenstra, and Lovasz have shown that their abovementioned "basis reduction" algorithm leads to a polynomial time algorithm for decomposing an arbitrary single variable polynomial into its irreducible factors over the integers [32]. Polynomial time algorithms have also been developed to handle the multivariate case, for any fixed number of variables, as reported by Kaltofen at FOCS-82 [24] (see also [23]). These results assume that the polynomial to be factored is represented by the sequence (or array in the multivariate case) of coefficients (including zero coefficients) for all terms up to those of maximum degree, a representation which is perhaps overly verbose when it comes to sparse multivariate polynomials. Thus a word of caution may be necessary: As illustrated in Section A7 of [G&J], many problems that are easy when polynomials are represented in this way become NP-hard when more con-

cise representations are used.

3. NEW ANGLES ON LINEAR PROGRAMMING

When L. G. Khachiyan [26] discovered in 1979 that the Linear Programming problem can be solved in polynomial time by the “ellipsoid method,” he opened as many doors as he closed. An initial flurry of research into the ellipsoid method has now given way to a more detailed analysis of other algorithms for the problem, both old and new. Of particular interest have been attempts to explain why the “simplex method,” which with its variants remains the method of choice for solving linear programming problems, performs so well in practice, despite its exponential worst-case running time. At the recent International Symposium on Mathematical Programming, held August 23-27, 1982 in Bonn, two researchers, Karl-Heinz Borgwardt and Steven Smale, independently announced results that claim to provide some of this “explanation.”

The type of analysis they perform departs in two ways from the complexity analysis used in the theory of NP-completeness. Firstly, as might be expected in an attempt to explain the behavior of algorithms “in practice,” they are interested in *average case*, rather than worst-case behavior. Secondly, and more subtly, they measure complexity in the “real arithmetic model,” where each arithmetic operation has unit cost, and express running times as functions of n and m , the numbers of variables and constraints, instead of in terms of an “input length” that takes into account the number of bits required to represent an instance. (Note that this change in model does not significantly affect the worst-case complexity of the simplex method: it remains exponential. The ellipsoid method, however, suffers greatly. Not only is it no longer polynomial, it is not even bounded. Even when $m = n = 2$, it can produce arbitrarily long running times [52].)

A first result on the expected behavior of simplex-type algorithms was presented in 1980 by Dantzig [12], but this result concerned only a restricted special case of the linear programming problem. Borgwardt and Smale each consider the general problem. Recall that an instance of linear programming can be viewed as an $m \times n$ matrix of rationals A , together with m - and n -tuples b and c , with the goal being to find an n -tuple x of non-negative rationals such that $Ax \leq b$ and cx is maximized. The simplex method works by a series of “pivot steps” that walk from vertex to vertex around a multi-dimensional polyhedron (defined by the instance) until an optimal solution is found (or it is determined that no solution exists). Each pivot step takes $O(nm)$ time in the real arithmetic model, so the number of such steps is the crucial determinant of the algorithm’s running time. Although various versions of the simplex method have different rules for “pivoting,” none have yet been proved to guarantee a polynomial number of pivots, and indeed most have been shown occasionally to require an exponential number (e.g., see [28]).

The results mentioned above indicate that such untoward behavior may be quite rare. Borgwardt [10,11] considers a simplex variant he calls the “Schattenalgorithmus” (shadow-vertices algorithm), in which pivoting is done based on a two-dimensional projection of the polyhedron. Assuming that b is the all-1’s vector (every instance can be normalized so that it has this form), and that c and the rows of A are distributed independently, identically, and symmetrically with respect to rotations about the origin, he shows in [11] that the expected number of pivots is $O(n^4 m)$. In an earlier paper [10], Borgwardt had analyzed what happens in his model when n is fixed and m goes to infinity, and was able to provide a general bound of $O(m^{1/(n-1)})$ on the expected number of pivot steps, with better bounds for certain specific distributions (although all were exponential if n was not fixed). Smale [51] also fixes n , but considers a different simplex variant, the “self-dual parametric algorithm,” and a specific probability measure not included among those covered by Borgwardt’s result. The bound he obtains on the expected number of pivots can be viewed as asymptotically better than Borgwardt’s: $O((1 + \log(m+1))^{n(n+1)})$. (Smale’s actual result has m and n reversed, but duality allows us to interchange them.)

Although the mathematics involved in these results is impressive, there are, of course, limitations to their significance: The results only hold for the specific variants of the simplex algorithm mentioned, and the probability distributions used in both cases do not, on average, yield the kinds of sparse instances (instances with many zero-entries in A) that one encounters in practice. They also are unlikely to yield degenerate instances, even though, as Dantzig points out in his recent lively reminiscence on the early years of linear programming [13], such instances are surprisingly common in practice. The results for fixed n are further compromised by their asymptotic nature. When the number of variables is fixed and the number of randomly generated constraints is allowed to increase, the probability that the instance is infeasible (that there are *no* solutions) rapidly approaches 1, whereas in many practical applications, infeasible instances can never occur.

A result presented by Megiddo at FOCS-82 [36,37] casts further doubt on the explanatory power of the results for fixed n . Using a clever recursive application of a new multi-dimensional search technique, Megiddo has designed algorithms for linear programming that can be implemented with a *worst case* running time $O(m \log^n m)$ when n is fixed, or, even better in the asymptotic sense, $O(2^{2^n} m)$. The previous best worst-case bounds for this problem had been $O(m^{n/2})$ [28]. Megiddo makes no claims for the practicality of his algorithms when n is large. However, it is interesting to note that these worst case bounds are *better* than all the average-case bounds obtained above for fixed n , when those bounds are modified to take the time for performing pivot operations into account (each takes time at least $\Omega(n+m)$). For instance, Smale’s bound becomes roughly $O(m \log^{n(n+1)} m)$, which is worse than the first of Megiddo’s two bounds for all positive m and n . Thus the one average case result above that stands out is

Borgwardt's general $O(n^4 m)$ bound, which is polynomial in *both* n and m , whereas no algorithm is known whose worst-case complexity is polynomial in this sense.

I shall conclude this section with a brief discussion of two other new algorithms for linear programming. For this discussion we shall return to the standard world (from a complexity theorist's point of view) where running times are measured by bit operations, input sizes by bit lengths, and the ellipsoid method is once again polynomial. The excellent survey of work on the ellipsoid method by Bland, Goldfarb, and Todd has now appeared in final form [7], but new developments continue. One was presented at FOCS-82 by Ursic [53]. He considers the problem of controlling round-off errors when taking square roots in the standard versions of the algorithm, and shows that this can be avoided if one considers a variant in which all computations are performed in exact rational arithmetic using continued fractions.

Our other new polynomial time algorithm is actually an "old" one. As pointed out in [7], one of the prime sources of ideas that eventually led to Khachiyan's result was an 1965 algorithm developed independently by Newman [41] and A. Iu. Levin [34], which involved the shrinking of polytopes rather than ellipsoids. This approach was very complicated, and seemed to involve an exponential growth in the number of facets of the polytope. However, it was the inspiration for the work of Iudin and Nemirovsky, who, after studying this approach in [21], decided to make the crucial switch from polytopes to ellipsoids in [22]. The work of Iudin and Nemirovsky, plus work of Shor [50], led finally to Khachiyan's result and a polynomial time algorithm. It now turns out, however, that polytopes are enough. A more insightful analysis of the Levin and Newman algorithm was presented at FOCS-82 by Yamnitsky and L. A. Levin [55] (no relation; this is the Levin whose 1973 paper [35] independently invented the idea of NP-completeness). They show that the polytopes in the 1965 algorithm don't really get as complicated as was feared, and hence this algorithm too can be implemented to run in polynomial time (although its "polynomial" suffers from the same drawbacks in practice as those for the more elegant ellipsoid method). It is perhaps too early to judge the significance of this result, but it certainly has value as historical commentary, and gives rise to a number of intriguing questions of the "what if" variety.

4. AN ABBREVIATED GAMUT

This month's new NP-hard problems do not have a common theme, but instead are intended to reflect the variety of research in the field. As we shall see, however, the first bears some relation to the results discussed in Section 2.

[1] SIMULTANEOUS DIOPHANTINE APPROXIMATION

INSTANCE: A finite vector of rationals $\alpha = (a_1/b_1, \dots, a_d/b_d)$, positive integer N , and a rational $\epsilon > 0$.

QUESTION: Is there a positive integer $Q \leq N$ such that $\{\{Q\alpha\}\} \leq \epsilon$, where $\{\{Q\alpha\}\}$ is defined to be $\text{MAX}_{1 \leq i \leq d} \{\text{MIN}_{n \in \mathbb{Z}} |Qa_i/b_i - n|\}$? In other words, do there exist integers n_1, \dots, n_d such that, for $1 \leq i \leq d$,

$$|\frac{n_i}{Q} - \frac{a_i}{b_i}| \leq \frac{\epsilon}{Q}?$$

Reference. Lagarias at FOCS-82 [29]. Transformation from WEAK PARTITION [June 1982].

Comment. For $d = 1$, a continued fractions approach has long been known to yield an efficient algorithm, and Shamir uses this as one step in his procedure for breaking the Merkle-Hellman public key cryptosystem, discussed in Section 2. There are indications that the higher dimensional problems may also have cryptanalytic implications [30]. If d is fixed but greater than 1, the problem is solvable in polynomial time using Lenstra's fixed-dimension integer programming algorithm [33]. In general, it is known by Dirichlet's Theorem that the desired Q always exists if $N \geq 1/\epsilon^d$. However, the fact that a Q exists for a given α , N , and ϵ does not mean we can find one easily. The best that one can presently guarantee (with an algorithm that runs in polynomial time for unbounded d) is to find a $Q^* \leq f(d)N$ with $\{\{Q^*\alpha\}\} \leq g(d)\epsilon$, where f and g are both exponential functions of d [29] (this is yet another application of the "basis reduction" algorithm of [32]). Further results, concerning "best simultaneous approximation denominators" (Q such that $\{\{Q\alpha\}\} \leq \{\{Q'\alpha\}\}$ for $1 \leq Q' < Q$), are described in [29].

[2] SINGLE-LETTER CONTEXT-FREE LANGUAGE MEMBERSHIP

INSTANCE: Context-free grammar $G = (N, \Sigma, \Pi, S)$, where the set Σ of terminal symbols has one element w , and a positive integer K .

QUESTION: Is the string $x = w^K$ in the language generated by G ?

Reference. Huynh at FOCS-82 [20]. Transformation from SUBSET SUM [SP13]. For those whose formal language theory is a bit rusty, definitions of "context-free" etc. can be found in [19].

Comment. Solvable in pseudo-polynomial time by standard parsing techniques for context-free languages. The name of this problem is a bit misleading, since in normal language membership problems the string x (and not just its length) is given in the input. The surprising result here is that the current problem is in fact in NP. Also surprising is the fact that the inequivalence problem for grammars of this type is in PSPACE (more precisely, it is complete for the

class Σ_2^P in the polynomial hierarchy [20]). Recall that the problem of inequivalence of arbitrary context free grammars is undecidable (e.g., see [19]). Note also that, although every single-letter context-free language is a “regular” language, the current results do not hold for single-letter regular grammars. The membership problem (given K , not x) is solvable in polynomial time for such grammars, and the inequivalence problem is “only” NP-complete (the trick in both proofs is to use matrix multiplication). This difference in complexities is explained by the greater expressive power of context-free grammars. Even though a language is regular, it is still possible that no regular grammar can represent it as succinctly as does a given context-free grammar.

[3] GROUND STATE OF A SPIN GLASS

INSTANCE: Positive integers H , L , and W , the 3-dimensional grid graph $G = (V, E)$ whose vertices are the integer-coordinate points (a, b, c) , $1 \leq a \leq H$, $1 \leq b \leq L$, $1 \leq c \leq W$, and whose edges connect each pair of vertices that are adjacent in one of the three directions, an integer *interaction weight* $J(e)$ for each edge $e \in E$, and an integer K .

QUESTION: Is there an assignment of a *spin* $s(v) \in \{-1, +1\}$ to each vertex $v \in V$ such that the “ground state spin energy” is K or less, i.e., such that

$$- \sum_{\{u,v\} \in E} J(u,v) s(u) s(v) \leq K ?$$

Reference. Barahona [4]. Transformation from SIMPLE MAX CUT [ND16]. In the process, the latter problem is shown to be NP-complete for 3-dimensional grid graphs of depth 2.

Comment. Remains NP-complete even if $H = 2$ and all interaction values are from the set $\{-1, 0, +1\}$. Solvable in polynomial time for arbitrary interaction values if $H = 1$ (and hence G is planar), for then the problem becomes a special case of planar MAX CUT and can be solved in a number of ways [4,5,6]. However, for arbitrary planar graphs, NP-hardness returns when an external magnetic field is introduced (this adds a new term to the spin energy function) [4]. It is not suggested that the complexity of computing a ground state will keep a physical system from finding one. Sooner or later, entropy must have its way (intriguing questions about “how soon” are raised, however). Nevertheless, although NP-completeness is unlikely to tell us anything specific about the physics of spin glasses, physicists have recently proposed that statistical mechanical techniques, developed for analyzing more realistic spin glass problems, may be applicable to a variety of NP-complete optimization problems. In a paper written for *Science* [27] that has already attracted attention in the press [1,48], Kirkpatrick, Gelatt, and Vecchi suggest that such objects as near-optimal Traveling Salesman tours can be found by a randomized algorithm that is analogous to

“melting” and then gradually “cooling” a spin glass until it “freezes.” This “simulated annealing” approach is actually a variant on neighborhood search, with the randomization used to prevent the search from getting stuck prematurely at a poor local optimum. It may well prove useful, especially when dealing with very large instances where more sophisticated techniques would be hopelessly expensive. As yet, however, little has been done in the way of substantive testing or theoretical analysis.

[4] ROBOT ARM REACHABILITY

INSTANCE: Graph $G = (V, E)$ with a two-dimensional integer-coordinate *initial location* $L(v)$ for each $v \in V$, a set of *fixed joints* $F \subseteq V$, a *terminal joint* $v_T \in V$, and an integer-coordinate *destination* (x, y) .

QUESTION: Is there a strategy for continuously moving the vertices of G from their initial locations to a final configuration in which v_T is at (x, y) , subject to the constraint that at all times the fixed joints remain stationary and, if $\{u, v\} \in E$, then the Euclidean distance between u and v remains constant?

Reference. Hopcroft, Joseph, and Whitesides at FOCS-82 [17]. Transformation from LINEAR BOUNDED AUTOMATON ACCEPTANCE [AL3].

Comment. The problem corresponds to that of moving a robot arm around in the plane, and is PSPACE-hard when arbitrary graphs (i.e., linkages) are allowed. It is solvable in polynomial time if G is a path with one endpoint fixed and the other the terminal joint, even if G is constrained to lie within a circle [17]. The question of whether such a simple arm can be folded up so that all its joints lie on a given finite line segment is, however, NP-complete (and has been dubbed, for obvious reasons, the CARPENTER’S RULE FOLDING problem [17]). If one wishes to specify the final locations for all the vertices of G , the “path in a circle” problem remains solvable in polynomial time [17], but the “tree in a 3-dimensional non-convex polyhedron” problem is PSPACE-complete [44]. On the other hand, the question of whether a given rigid convex polyhedron can be moved from one location to another without touching any of a collection of polyhedral obstacles (the SOFA MOVERS problem) can be solved in polynomial time in both 2 and 3 dimensions [44,46,47].

In keeping with the spirit of this column, our “Open Problem of the Month” is as unrelated to the preceding four NP-hard problems as they are to each other:

[OPEN] GRACEFUL GRAPH

INSTANCE: Graph $G = (V, E)$.

QUESTION: Does G have a “graceful” numbering, i.e., is there a one-to-one function $i: V \rightarrow \{0, 1, 2, \dots, |E|\}$ such that $\{|i(u) - i(v)| : \{u, v\} \in E\} = \{1, 2, \dots, |E|\}$?

Comment. The term “graceful graph” was introduced by Golomb [16]. Many graceful and ungraceful graphs have since been identified (see [15] for a bibliography), and it is conjectured (but not proved) that all trees are graceful. Although this concept might appear to have been devised mainly for the amusement of graph theorists, it turns out to have a variety of practical applications, from X-ray crystallography to missile guidance [8,9]. (Actually, the applications tend to concern a more general, and also open, problem: Given G , find the least N such that there is a labelling of V with distinct integers from $\{0, 1, \dots, N\}$ which yields distinct values of $|i(u) - i(v)|$ for all edges $\{u, v\}$. Such an N must be at least $|E|$, and if it equals $|E|$ the graph is graceful.) A related open problem concerns the complexity of identifying what Graham and Sloane [15] call “harmonious” graphs: ones in which vertices can be assigned distinct integers $i(v)$ so that $\{i(u) + i(v) \pmod{|E|} : \{u, v\} \in E\} = \{0, 1, \dots, |E| - 1\}$. The one related problem that is known to be NP-complete is the “harmonious coloring” problem of Hopcroft and Krishnamoorthy [18]. Here we are given G and an integer K , and wish to assign to each vertex a (not-necessarily-distinct) integer $i(v) \in \{1, 2, \dots, K\}$ in such a way that $|\{i(u), i(v) : \{u, v\} \in E\}| = |E|$, i.e., so that no two edges get the same set of “colors” on their endpoints.

5. ACKNOWLEDGMENT

Although in writing this column I do my best to give the impression of intimate familiarity with the forefronts of research in all relevant areas, I must admit to having some help from my colleagues here at Bell Labs. In particular, this month’s effort was greatly assisted by A. V. Aho, P. W. Anderson, M. R. Garey, R. L. Graham, J. C. Lagarias, S. R. Mahaney, A. M. Odlyzko, R. Y. Pinter, and S. Wolfram.

REFERENCES

1. (ANONMOUS), Math of a salesman, *Science* **82** **3** (Sept. 1982), 7-8.
2. L. M. ADLEMAN, private communication (1982).
3. L. M. ADLEMAN AND R. McDONNELL, An application of higher reciprocity to computational number theory, in “Proceedings 23rd Ann. Symp. on Foundations of Computer Science,” pp. 100-106, IEEE Computer Society, Los Angeles, 1982.
4. F. BARAHONA, On the computational complexity of Ising spin glass models, *J. Phys. A: Math. Gen.* **15** (1982), 3241-3253.
5. F. BARAHONA, R. MAYNARD, R. RAMMAL, AND J. P. UHRY, Morphology of ground states of two-dimensional frustration model, *J. Phys. A: Math. Gen.* **15** (1982), 673-699.
6. I. BIECHE, R. MAYNARD, R. RAMMAL, AND J. P. UHRY, On the ground states of the frustration model of a spin glass by a matching method of graph theory, *J. Phys. A: Math. Gen.* **13** (1980), 2553-2576.
7. R. G. BLAND, D. GOLDFARB, AND M. J. TODD, The ellipsoid method: a survey, *Operations Res.* **29** (1981), 1039-1091.

8. G. S. BLOOM AND S. W. GOLOMB, Applications of numbered undirected graphs, *Proc. IEEE* **65** (1977), 562-570.
9. G. S. BLOOM AND S. W. GOLOMB, Numbered complete graphs, unusual rulers, and assorted applications, in "Theory and Applications of Graphs," pp. 53-65, Lecture Notes in Mathematics, Vol. 642, Springer-Verlag, New York, 1978.
10. K.-H. BORGWARDT, Some distribution-independent results about the asymptotic order of the average number of pivot steps of the simplex method, *Math. Oper. Res.* **7** (1982), 441-462.
11. K.-H. BORGWARDT, The average number of pivot steps required by the simplex method is polynomial, manuscript (1982).
12. G. B. DANTZIG, Expected number of steps of the simplex method for a linear program with a convexity constraint, Report No. SOL 80-3R, Systems Optimization Laboratory, Department of Operations Research, Stanford University, Stanford, Calif., 1980.
13. G. B. DANTZIG, Reminiscences about the origins of linear programming, *Operations Res. Lett.* **1** (1982), 43-48.
14. J. D. DIXON, Asymptotically fast factorization of integers, *Math. Comp.* **36** (1981), 255-260.
15. R. L. GRAHAM AND N. J. A. SLOANE, On additive bases and harmonious graphs, *SIAM J. Algebraic and Discrete Methods* **1** (1980), 382-404.
16. S. W. GOLOMB, How to number a graph, in "Graph Theory and Computing (R. C. Read, Ed.)," pp. 23-37, Academic Press, New York, 1972.
17. J. E. HOPCROFT, D. JOSEPH, AND S. WHITESIDES, On the movement of robot arms in 2-dimensional bounded regions, in "Proceedings 23rd Ann. Symp. on Foundations of Computer Science," pp. 280-289, IEEE Computer Society, Los Angeles, 1982.
18. J. E. HOPCROFT AND M. S. KRISHNAMOORTHY, On the harmonious coloring of graphs, *SIAM J. Algebraic and Discrete Methods*, to appear.
19. J. E. HOPCROFT AND J. D. ULLMAN, "Introduction to Automata Theory, Languages, and Computation," Addison-Wesley, Reading, Mass., 1979.
20. T.-D. HUYNH, Deciding the inequivalence of context-free grammars with 1-letter terminal alphabet is Σ_2^P -complete, in "Proceedings 23rd Ann. Symp. on Foundations of Computer Science," pp. 21-31, IEEE Computer Society, Los Angeles, 1982.
21. D. B. IUDIN AND A. S. NEMIROVSKII, Evaluation of the informational complexity of mathematical programming problems, *Ekonomika i Matematicheskie Metody* **12** (1976), 128-142 (in Russian). English translation in *Matekon: Translations of Russian and East European Math. Economics* **13** (Winter '76-'77), 3-25.
22. D. B. IUDIN AND A. S. NEMIROVSKII, Informational complexity and effective methods of solution for convex extremal problems, *Ekonomika i Matematicheskie Metody* **12** (1976), 357-369 (in Russian). English translation in *Matekon: Translations of Russian and East European Math. Economics* **13** (Spring '77), 25-45.
23. E. KALTOFEN, A polynomial reduction from multivariate to bivariate integer polynomial factorization, in "Proceedings 14th Ann. ACM Symp. on Theory of Computing," pp. 261-266, Association for Computing Machinery, New York, 1982.
24. E. KALTOFEN, A polynomial-time reduction from bivariate to univariate integral polynomial factorization, in "Proceedings 23rd Ann. Symp. on Foundations of Computer Science," pp. 57-64, IEEE Computer Society, Los Angeles, 1982.
25. R. KANNAN, Improved algorithms for integer programming and related lattice problems, manuscript (1982).
26. L. G. KHACHYAN, A polynomial algorithm in linear programming, *Dokl. Akad. Nauk. SSSR* **244** (1979), 1093-1096 (in Russian). English translation in *Soviet Math. Dokl.* **20** (1979), 191-194.
27. S. KIRKPATRICK, C. D. GELATT, JR, AND M. P. VECCHI, Optimization by simulated annealing, *Science*, to appear.
28. V. KLEE AND G. J. MINTY, How good is the simplex algorithm?, in "Inequalities III (O. Shisha, Ed.)," pp. 159-175, Academic Press, New York, 1972.

29. J. C. LAGARIAS, The computational complexity of simultaneous Diophantine approximation problems, in "Proceedings 23rd Ann. Symp. on Foundations of Computer Science," pp. 32-39, IEEE Computer Society, Los Angeles, 1982.
30. J. C. LAGARIAS, private communication (1982).
31. A. LEMPEL, Cryptology in transition, *Comput. Surveys* **11** (1979), 285-303.
32. A. K. LENSTRA, H. W. LENSTRA, JR, AND L. LOVÁSZ, "Factoring polynomials with rational coefficients," Report No. 82-05, Mathematics Institute, University of Amsterdam, Amsterdam, 1982.
33. H. W. LENSTRA, JR, "Integer programming with a fixed number of variables," Report No. 81-03, Department of Mathematics, University of Amsterdam, Amsterdam, 1981.
34. A. I. LEVIN, On an algorithm for the minimization of convex functions, *Dokl. Akad. Nauk. SSSR* **160** (1965), 1244-1247 (in Russian). English translation in *Soviet Math. Dokl.* **6** (1965), 286-290.
35. L. A. LEVIN, Universal sorting problems, *Problemy Peredaci Informacii.* **9** (1973), 115-116 (in Russian). English translation in *Problems of Information Transmission* **9** (1973), 265-266.
36. N. MEGIDDO, Linear-time algorithms for linear programming in R^3 and related problems, in "Proceedings 23rd Ann. Symp. on Foundations of Computer Science," pp. 329-338, IEEE Computer Society, Los Angeles, 1982.
37. N. MEGIDDO, Solving linear-programming in linear-time when the dimension is fixed, manuscript (1982).
38. C. MEINEL, Adelman and Apple II bust public-key crypto code, *Info World* **4** (Oct. 4, 1982), 10,14.
39. R. C. MERKLE AND M. E. HELLMAN, Hiding information and signatures in trapdoor knapsacks, *IEEE Trans. Inform. Theory* **IT-24** (1978), 525-530.
40. M. A. MORRISON AND J. BRILLHART, A method of factoring and the factorization of F_7 , *Math. Comp.* **29** (1975), 183-205.
41. D. J. NEWMAN, Location of the maximum on unimodal surfaces, *J. Assoc. Comput. Mach.* **12** (1965), 395-398.
42. C. POMERANCE, Analysis and comparison of some integer factoring algorithms, manuscript (1982).
43. M. O. RABIN, "Digitalized signatures and public-key functions as intractable as factorization," Report No. MIT/LCS/TR211, MIT Laboratory for Computer Science, Cambridge, Mass., 1979.
44. J. H. REIF, Complexity of the mover's problem and generalizations, in "Proceedings 20th Ann. Symp. on Foundations of Computer Science," pp. 421-427, IEEE Computer Society, Los Angeles, 1979.
45. R. RIVEST, A. SHAMIR, AND L. ADLEMAN, A method for obtaining digital signatures and public-key cryptosystems, *Comm. ACM* **21** (1978), 120-126.
46. J. T. SCHWARTZ AND M. SHARIR, "On the 'piano movers' problem I: The case of a two-dimensional rigid polygonal body moving amidst polygonal barriers," Report No. TR 39, Department of Computer Science, New York University, New York, 1981.
47. J. T. SCHWARTZ AND M. SHARIR, "On the 'piano movers' problem II: General techniques for computing topological properties of real algebraic manifolds," Report No. TR 41, Department of Computer Science, New York University, New York, 1982.
48. B. M. SCHWARTZSCHILD, Statistical mechanics algorithm for Monte Carlo optimization, *Physics Today* **35** (May 1982), 17-19.
49. A. SHAMIR, A polynomial time algorithm for breaking the basic Merkle-Hellman cryptosystem, in "Proceedings 23rd Ann. Symp. on Foundations of Computer Science," pp. 145-152, IEEE Computer Society, Los Angeles, 1982.
50. N. Z. SHOR, Cut-off method with space extension in convex programming problems, *Kibernetika* **13** (1977), 94-95 (in Russian). English translation in *Cybernetics* **13** (1977), 94-96.

51. S. SMALE, On the average speed of the simplex method of linear programming, manuscript (1982).
52. J. F. TRAUB AND H. WOZNAKOWSKI, Complexity of linear programming, *Operations Res. Lett.* **1** (1982), 59-62.
53. S. URSIC, The ellipsoid algorithm for linear inequalities in exact arithmetic, in "Proceedings 23rd Ann. Symp. on Foundations of Computer Science," pp. 321-326, IEEE Computer Society, Los Angeles, 1982.
54. H. C. WILLIAMS, A modification of the RSA public-key encryption procedure, *IEEE Trans. Inform. Theory* **IT-26** (1980), 726-729.
55. B. YAMNITSKY AND L. A. LEVIN, An old linear programming algorithm runs in polynomial time, in "Proceedings 23rd Ann. Symp. on Foundations of Computer Science," pp. 327-328, IEEE Computer Society, Los Angeles, 1982.