COMPLEXITY ISSUES IN CODING THEORY

Alexander Barg

Bell Laboratories, Lucent Technologies
600 Mountain Avenue, Room 2C-375
Murray Hill, NJ 07974
abarg@research.bell-labs.com

Abstract. This paper deals with complexity issues in the theory of linear error-correcting codes. Algorithmic problems that we study are constructing good codes, encoding and decoding them. According to their complexity. problems are divided into easy, i.e., polynomial in the length n of the code, and difficult, i.e., exponential ones. The first part deals with easy problems. We present a construction of codes that correct a linear fraction of errors with complexity $n \log n$. The construction is based on well-known since the late 80ies explicit constructions of good expanding graphs. Another group of problems in this part is related to codes for non-Hamming errors, namely, erasures, defects (codes for memories with defective cells), and localized errors.
The second part, which forms the core of this paper, deals with difficult problems, first and foremost, maximum likelihood decoding of linear codes. We study separately the complexity of hard-decision and soft-decision decoding. For the hard-decision decoding case we present algorithms grouped in two classes, gradient-like decoding and information-set decoding. It turns out that this general approach is sufficient to study most if not all known general decoding methods. In the soft-decision decoding context, we first discuss possible problem settings and then implementations of decoding with reduced complexity.
The last part of the paper overviews most known NP-hard decoding problems including some recent nonapproximability results.
The supporting material includes many general properties of linear codes from well-known to rather sophisticated, and a brief discussion of models of computations and relevant settings for the study of complexity issues in coding theory. We also give examples of many methods studied. Sometimes they just illustrate concepts and definitions, but sometimes capture the most essential features of the proofs and on occasion even replace them.
Generally we give complete and self-contained proofs of the results.
The coverage is extended from classical algorithms up to very recent developments. We thoroughly study and compare different algorithms, especially those applicable to several seemingly non-related problems. This unified approach to algorithmic coding problems enables us to organize previously independent results in a self-contained part of coding theory.
This paper will appear as a chapter in Handbook of Coding Theory, V. Pless, W. Cary Huffman and R. Brualdi, Eds., Elsevier Science, to be published.

Keywords: linear code, complexity, encoding, regular-graph codes, maximum likelihood decoding, Gilbert–Varshamov bound, soft-decision decoding, NP-hard problems.

4 Intractable Problems

In this section we present NP-complete coding problems and closely related results. Being NP-complete does not imply many practical consequences for a problem. Namely, even if P $\neq$ NP, this does not exclude the existence of simple probabilistic algorithms, algorithms that have a low average-case complexity, or a possibility to easily approximate the exact solution (a noteworthy exception to this being Theorem 4.6).

As discussed in Section 1.2, to prove that a problem $\mathcal{P}$ in NP is complete in it, we have to show that there is an NP-complete problem that can be polynomially reduced to $\mathcal{P}$.

The following is a list of some coding problems.

P_1 (Weight of error). Given a linear code C, an integer w and a vector $\boldsymbol{y}$, is it true that $\mathsf{dist}\,(\boldsymbol{y}, C) \leq w$? Since we want to encode instances of the problem as concisely as possible, we stick to the following formulation: Given a matrix H, a vector $\boldsymbol{s}$, and a nonnegative integer w, is there a vector $\boldsymbol{y}$ of weight $\mathsf{wt}\,(\boldsymbol{y}) \leq w$ such that $H\boldsymbol{y}^T = \boldsymbol{s}^T$?

P_2 (Minimal weight) Given a matrix H and an integer w, is there a nonzero vector $\boldsymbol{c}$ of weight $\leq w$ that belongs to the kernel of H: $H\boldsymbol{c}^T = 0$?

P_3 (Codeword of given weight). In the same setting, is there a vector $\boldsymbol{c} \in \mathsf{ker}\,(H)$ of weight w?

P_4 (Maxweight). In the same setting, is there a vector $\boldsymbol{c} \in \mathsf{ker}\,(H)$ with $\mathsf{wt}\,(\boldsymbol{c}) \geq w$?

P_5 (Farthest-off point). Given a subset $M \subset E_2^n$ and $w \geq 0$, is there a vector $\boldsymbol{y} \in E_2^n$ with $\mathsf{dist}\,(\boldsymbol{y}, M) \geq w$?

P_6 (Partition rank). Given a binary matrix H with n columns, and positive integers w and i, find out whether there is a partition $\mathcal{N} = I \cup (\mathcal{N} \setminus I)$, $|I| = i$, such that

$$\mathsf{rank}\, H(I) + \mathsf{rank}\, H(\mathcal{N} \setminus I) \leq w.$$

P_7 (Perfect code in regular graph). Given a planar 3-regular graph $G(V, E)$ with distance between any 2 vertices defined as the number of edges in a shortest path connecting them, find out whether G contains an r-perfect code, i.e., a subset $A \subset V$ such that any vertex $a \in V \setminus A$ is at a distance at most r of exactly one member of A and any two vertices in A are more that r edges apart.

It is easy to see that problems P_1–P_7 are in NP. As a side remark, we note that problems complementary to P_1–P_7 are not necessarily in NP. Indeed, looking at P_1, suppose we want to check the statement that *all* vectors $\boldsymbol{y}$ with syndrome $\boldsymbol{s}$ have weight $> w$. We are not aware of a way to check this in polynomial time. Problems complementary to the decision problems from the class NP form the class coNP.

Various results about the NP-completeness are collected in the following theorem.

Theorem 4.1 *(a)* P_1 *is* NP-*complete for any fixed size of the ground alphabet* *q. It remains* NP-*complete even if* $C = A \otimes B$ *is a product code or a concatenated code* $A \boxtimes B$ *with nontrivial factors.*

(*b*) P_2 *is* NP-*complete for any given alphabet.*

(*c*) P_3 *is* NP-*complete for any given alphabet. If* $q = 2$*, it is NP-complete even if* $w = n/2$*. If* $q = 3$*, it is NP-complete even if* $w = n$*.*

(*d*) P_4 *is* NP-*complete for* $q = 2, 3$*.*

(*e*) *Problems* P_5–P_6 *are* NP-*complete for* $q = 2$*.*

(*f*) P_7 *is* NP-*complete even if* $r = 1$*.*

Remark 4.1 (*i*) Problem P_1 is closely related to minimum distance decoding. Indeed, any decoding algorithm can be used to solve P_1. Namely, given a syndrome $\boldsymbol{s}$, it is a matter of simple linear algebra to find a vector with this syndrome. Then running the decoding algorithm, we would obtain an answer to P_1. For this reason, one says that minimum distance decoding is NP-*hard*, meaning that it is not in the class NP itself but there is an NP-complete problem (P_1) that can be polynomially reduced to it.

Further evidence of the difficulty of Problem P_1 is given by Theorem 4.6, which says that even approximating the answer up to *any* given constant is NP-complete.

(*ii*) In the same fashion, Problem P_2 (P_3) is related to the minimum (resp., maximum weight) of the code, Problem P_5 to the covering radius of an (unrestricted) code, and Problem P_6 to the minimal trellis complexity (defined in Sect. 3.4.1).

(*iii*) For $q = 3$ and $w = n$, Problem P_2 can be also formulated as follows: given n elements of a fixed extension of the field $\mathbf{F}_3$, is it possible to place the signs in the sum $\sum_{i=1}^{n} \pm a_i$ so that it vanishes? This is the same as P_2 if we view columns of H as elements of an extension field. ∎

Turning to the proof of Theorem 4.1, we shall discuss only relatively simple reductions, omitting more difficult parts. The following problems are NP-complete, see Garey and Johnson [71].

K_1 (three-dimensional matching). Given a t-set W and a collection $U \subset W^3$ of triples find out whether there exists a subset $V \subset U$, $|V| = t$, such that for any two distinct triples $v_i, v_j \in V$ the corresponding components are different: $v_{i1} \neq v_{j1}, v_{i2} \neq v_{j2}, v_{i3} \neq v_{j3}$.

K_2 (monotone 1-in-3 SAT). Given a set of Boolean variables $U = \{u_1, \ldots, u_n\}$ and a set of clauses such that each clause contains three literals, none of which are negated, is there a truth assignment to U such that each clause contains exactly one true literal?

K_3 (maximal cut or cycle). Given a simple graph and $w \geq 0$, is there a cut (cycle) with at least w edges?

Proof of Theorem 4.1: (a). We reduce K_1 to P_1. Form a $3t \times |U|$ incidence matrix H of the set U. The columns of the matrix correspond to the triples $u \in U$. Let $u_i = (u_{i_1}, u_{i_2}, u_{i_3})$. Then the ith column of H contains $3t - 3$ zeros and 3 ones in positions $u_{i_1}, t + u_{i_2}, 2t + u_{i_3}$. In order to construct an instance of

P_1 that corresponds to the instance of K_1, put $\boldsymbol{s} = (11\dots1)$ and $w = t$. Then if U contains a sought subset and $\boldsymbol{y}$ is its characteristic vector, we have $H\boldsymbol{y}^T = \boldsymbol{s}^T$. Conversely, if for some $\boldsymbol{y}$ of weight $\le t$ this inequality holds true, then clearly $\mathsf{wt}\,(\boldsymbol{y}) = t$. Hence for every $i, 0 \le i \le 3t$, the sum $s_i = \sum_{j=1}^{|U|} y_j h_{ij}$ contains only one nonzero term, i_0 say, and $y_{i_0} = 1$, i.e., $\boldsymbol{y}$ defines a three-dimensional matching in U. To build a reduction for product codes, we can take H as the parity-check for the code B and choose A to be any nontrivial code that contains the all-one vector. We omit the details.

(b). We shall reduce P_1 to P_2. Let m be the number of rows and n the number of columns in H. The interesting case is $w \le m$ since any linear code contains a code vector of weight less or equal than its redundancy plus one. Observe that the reduction in (a) necessarily involves all $|U|$ columns of H; therefore, the proof is unscathed if we replace $\le$ by $=$ in the statement of P_1. We call this version Exact weight of error and denote it by P_1'.

Rephrasing P_1' as in Remark 4.1(iii) above, we conclude that the following problem P_1'' is NP-complete: given a set of elements $A = \{\alpha_1, \alpha_2, \dots, \alpha_n, \sigma\}$ of $\mathbf{F}_{q^m}$, $\sigma \ne 0$, and an integer $w \ge 0$, is there a nonempty subset of *distinct* elements $\{\alpha_{i_1}, \alpha_{i_2}, \dots, \alpha_{i_w}\} \subset A$ such that $\sum\limits_{j=1}^{w} \alpha_{i_j} = \sigma$?

Let us reduce P_1'' to P_2. The central part of the proof deals with constructing a code over $\mathbf{F}_{q^m}$ whose minimum distance is different according as the answer to P_2 is *yes* or *no*. This will prove that P_2 is NP-complete over $\mathbf{F}_{q^m}$. Then what remains is to prove that given a q^m-ary code $\mathcal{C}$ one can always construct a q-ary code C such that the distance of $\mathcal{C}$ can be computed in polynomial time given the distance of C.

Let us reduce P_1'' to P_2 for q^m-ary codes. Let $\{\alpha_1, \alpha_2, \dots, \alpha_n, \sigma\}$ be a given instance of P_1''. Form the matrix

$$H = \begin{bmatrix} 1 & 1 & \dots & 1 & 0 \\ \alpha_1 & \alpha_2 & \dots & \alpha_n & 0 \\ \alpha_1^2 & \alpha_2^2 & \dots & \alpha_n^2 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ \alpha_1^{w-2} & \alpha_2^{w-2} & \dots & \alpha_n^{w-2} & 0 \\ \alpha_1^{w-1} & \alpha_2^{w-1} & \dots & \alpha_n^{w-1} & 1 \\ \alpha_1^{w} & \alpha_2^{w} & \dots & \alpha_n^{w} & \sigma \end{bmatrix}.$$

The interesting case is $w \le n$. Then H has the rank $w+1$ and the dimension of the code $\mathcal{C} = \mathsf{ker}\,(H)$ equals $n - w$. The following lemma is a crucial observation.

Lemma 4.2

$$d(\mathcal{C}) = \begin{cases} w+1 & \textit{if there is a subset } \{\alpha_{i_1}, \alpha_{i_2}, \dots, \alpha_{i_w}\} \textit{ with} \sum \alpha_{i_j} = \sigma, \\ w+2 & \textit{otherwise.} \end{cases}$$

Proof: Recall that we have assumed that all α's are distinct. Therefore, any $w \times w$ square submatrix of H that does not include the last column, is nonsingular.

If it includes the last column, then

$$\det H = \sigma \begin{vmatrix} 1 & 1 & \dots & 1 \\ \alpha_{i_1} & \alpha_{i_2} & \dots & \alpha_{i_w} \\ \alpha_{i_1}^2 & \alpha_{i_2}^2 & \dots & \alpha_{i_w}^2 \\ \vdots & \vdots & & \vdots \\ \alpha_{i_1}^{w-2} & \alpha_{i_2}^{w-2} & \dots & \alpha_{i_w}^{w-2} \\ \alpha_{i_1}^{w-1} & \alpha_{i_2}^{w-1} & \dots & \alpha_{i_w}^{w-1} \end{vmatrix} - \begin{vmatrix} 1 & 1 & \dots & 1 \\ \alpha_{i_1} & \alpha_{i_2} & \dots & \alpha_{i_w} \\ \alpha_{i_1}^2 & \alpha_{i_2}^2 & \dots & \alpha_{i_w}^2 \\ \vdots & \vdots & & \vdots \\ \alpha_{i_1}^{w-2} & \alpha_{i_2}^{w-2} & \dots & \alpha_{i_w}^{w-2} \\ \alpha_{i_1}^{w} & \alpha_{i_2}^{w} & \dots & \alpha_{i_w}^{w} \end{vmatrix}.$$

The first term on the r.h.s. is the Vandermonde determinant $\Delta(i_1 \dots, i_w)$. Let us compute the second term. Changing α_{i_w} to x changes it into a polynomial in x. This polynomial has zeros $\alpha_{i_1} \dots, \alpha_{i_{w-1}}$ and thus, is divisible by $\prod_{j=1}^{w-1}(x - \alpha_{i_j})$. Continuing in this manner, we conclude that the necessary determinant is divisible by $\Delta(i_1 \dots, i_w)$. Comparing degrees, we see that it equals $\Delta(i_1 \dots, i_w)$ times a linear form of α's. Since the answer is symmetrical in α's, this linear form is $\sum_{j=1}^{w} \alpha_{i_j}$ times a constant. This constant must be one since the determinant in question does not have identical terms. Thus,

$$\det H = (\sigma - \sum_{j=1}^{w} \alpha_{i_j})\Delta(i_1 \dots, i_w),$$

which vanishes exactly when P_1'' has a positive answer. ∎

Thus, P_2 is shown to be NP-complete for q^m-ary codes. Note that in the course of this reduction we pass from $\mathbf{F}_q$ to its extension of degree m, where m is a part of the instance. It is known [141] that one can construct extensions (irreducible polynomials) in polynomial time.

The second part is, given an instance of P_2 for q^m-ary codes, to present a polynomial reduction to q-ary codes. This is easy since both codes have alphabets of the same characteristic. To build this reduction, we construct a q-ary concatenated code $C = \mathcal{C} \boxtimes B$ (see the definition in 2.1.2) with $\mathcal{C}$ the outer code. Our objective is, given C, to be able to determine the distance of $\mathcal{C}$ in polynomial time. This will be achieved by imposing restrictions on the parameters of B, which we denote by $[n_B, k_B, d_B]$. We use similar notation for the parameters of $\mathcal{C}$, namely, $[n_{\mathcal{C}}, k_{\mathcal{C}}, d_{\mathcal{C}}]$. Note that for C to be well-defined, k_B should be at least m. A double counting argument similar to Theorem 14.1 in Chapter 1 shows that the distance of C

$$d_{\square} \le n_B d_{\mathcal{C}} \frac{q^{m-1}(q-1)}{q^m - 1}.$$

Remember that we have to distinguish the cases $d_{\mathcal{C}} = w + 1$ and $d_{\mathcal{C}} = w + 2$. In the first case, we have

$$d_{\square} \le n_B (w+1) \frac{q^{m-1}(q-1)}{q^m - 1}. \tag{4.1}$$

The definition of C implies that

$$d_\square \geq d_{\mathcal{C}} d_B.$$

Therefore, if $d_{\mathcal{C}} = w + 2$, we have

$$d_\square \geq (w+2) d_B.$$

We need to chose d_B so that the r.h.s. of this inequality is strictly greater than that of (4.1). This will be satisfied if

$$\frac{d_B}{n_B} > \frac{w+1}{w+2} \frac{q^{m-1}(q-1)}{q^m - 1} = \frac{q-1}{q} - \frac{q-1}{q} \frac{q^m - w - 2}{(w+2)(q^m-1)},$$

where $w \leq m$. Such codes exist and can be polynomially constructed. Indeed, if the length $n_{\mathcal{C}}$ of the code $\mathcal{C}$ is small, its distance can be found by exhaustive search. If $n_{\mathcal{C}}$ grows, then so do w and m. Then we need a polynomially constructible code with distance approaching $(q-1)/q$. An example is given by concatenating a q^m-ary Reed–Solomon code with a q-ary first-order Reed–Muller code; we omit the details.

(c). Let us reduce K_1 to P_2. Let $|U| = n$. As above, form a $3t \times n$ incidence matrix of U and construct the following $(3t(n+1) \times 3t(n+1) + n)$ matrix:

$$H = \begin{bmatrix} H_1 & \\ I_n & \\ \vdots & I_{3t(n+1)} \\ & \\ I_n & \end{bmatrix}.$$

Let $w = 3t^2 + 4t$. Let $\boldsymbol{c} = (\boldsymbol{c}_1, \boldsymbol{c}_2)$, where $\boldsymbol{c}_1$ has length n and $\boldsymbol{c}_2$ has length $3t(n+1)$. If U contains a matching, then taking $\boldsymbol{c}_1$ as its incidence vector, we see that for the product Hc^T to vanish, $\boldsymbol{c}_2$ has to have weight $3t + t \cdot 3t$. Conversely, if $\mathsf{wt}\,(\boldsymbol{c}) = w$ and $H\boldsymbol{c}^T = 0$, then we have to prove that the product $\boldsymbol{s} = H_1 \boldsymbol{c}_1^T$ has weight $3t$:

$$\mathsf{wt}\,(\boldsymbol{c}_2) = \mathsf{wt}\,(\boldsymbol{s}) + 3t\mathsf{wt}\,(\boldsymbol{c}_1).$$

Adding $\mathsf{wt}\,(\boldsymbol{c}_1)$ to both parts, we obtain the equation

$$\mathsf{wt}\,(\boldsymbol{c}) = \mathsf{wt}\,(\boldsymbol{s}) + (3t+1)\mathsf{wt}\,(\boldsymbol{c}_1)$$

or

$$3t^2 + 4t = (3t+1)\mathsf{wt}\,(\boldsymbol{c}_1) + \mathsf{wt}\,(\boldsymbol{s}), \quad 0 \leq \mathsf{wt}\,(\boldsymbol{s}) \leq 3t,$$

which admits a unique solution $\mathsf{wt}\,(\boldsymbol{c}_1) = t, \mathsf{wt}\,(\boldsymbol{s}) = 3t$.

The reduction for $q = 2$, $w = n/2$ will be omitted.

If $q = 3$ and $w = n$, we can reduce K_2 to P_2 as follows.

Form the $r \times n$-incidence matrix B of the function $C = c_1 \& c_2 \& \ldots \& c_r$ by putting $b_{ij} = \chi(u_j \in c_i)$. Let $\boldsymbol{z} = (z_1, \ldots, z_n)$ be the vector corresponding to a truth assignment to the variables U, where $z_j = 1$ if u_j is set to *false* and $z_j = 2$ otherwise. The following simple fact is used in the proof.

Lemma 4.3 Let $\boldsymbol{z}$ be a n-vector of ones and twos. Then $B\boldsymbol{z}^T = (11\ldots1)^T$ over $\mathbf{Z}_3$ iff $\boldsymbol{z}$ provides a truth assignment with the properties required in K_2.

Proof: For the nonzero ternary numbers a_1, a_2, a_3, the only way that $a_1 + a_2 + a_3 = 1$ is that exactly one of them equals 2 while the two remaining equal 1. ∎

Form the matrix $H_2 = (B|\mathbf{1})$ by augmenting B with the all-one column and put $w = n+1$. If there exists an assignment $\boldsymbol{z}$ satisfying K_2, then the vector $\boldsymbol{x} = (\boldsymbol{z}|\mathbf{2})$ is annihilated by H_2. Conversely, suppose that there exists a vector $\boldsymbol{x}$ such that $H_2\boldsymbol{x}^T = 0$ and $\mathsf{wt}\,(\boldsymbol{x}) = n+1$. Then if $x_{n+1} = 2$, the necessary assignment is given by the first n coordinates of $\boldsymbol{x}$. If $x_{n+1} = 1$, then by the lemma, the necessary assignment is given by the vector $\boldsymbol{z} = (2x_1, \ldots, 2x_n)$.

(d). Codes and graphs. In Sect. 2 we constructed codes from regular graphs. There is also another way of associating a binary linear code to a graph $G\{V, E\}$, which takes more account of classical linear spaces in graphs. Let $|V| = v$ and $|E| = e$. A *cut* in G is a set of edges such that removing them increases the number of connected components by 1. Characteristic vectors of cuts form a linear subspace E_2^e of the space of all binary words of length e. The dimension of this subspace (code) C is $v-1$. The dual code $C^\perp$ has the parameters $[e, e-v+1]$ and equals the linear space of *cycles* of G.

Then P_3 is NP-complete for $q = 2$ by an obvious reduction from K_3. For $q = 3$ its completeness follows from part (b).

Proofs of parts (e)–(f) are omitted. ∎

Lemma 4.2 implies one interesting corollary. Namely, it is NP-complete to find out whether a code over a given field $\mathbf{F}_{p^m}$ is MDS or not, i.e., whether its distance is one greater than its redundancy.

We would like to draw the reader's attention to part (e) of this theorem. Problem P_5 accepts as an input string the entire code M, in striking contrast to all other problems considered, where we deal with a concise representation of the code of total length $\mathcal{O}(\log^2 |M|)$. Therefore, a natural question is what is the complexity of computing the covering radius if we are only allowed to store an input of a similar size. Then intuition suggests that the covering radius should be a more difficult problem than, for instance, decoding. Indeed, a naïve way to decode is to take a given vector $\boldsymbol{y}$ and compare it with all codewords in order to find the closest one. To find the covering radius one has to take a vector $\boldsymbol{y}$, compute its distance to the code, and repeat this for *all* vectors outside the code in order to find the farthest-off one.

P_8 (Upper bound on covering radius). Given a linear code C and $w \ge 0$, is it true that

$$\forall_{\boldsymbol{y}\in E_q^n}\, \exists_{\boldsymbol{c}\in C}\ (\mathsf{dist}\,(\boldsymbol{y}, \boldsymbol{c}) \le w)\,? \tag{4.2}$$

Since the covering radius equals the maximum weight of a coset leader, an equivalent formulation is to ask whether for any given $\boldsymbol{s}$ there is a vector with this syndrome whose weight does not exceed w.

Decision problems involving quantified expressions of type (4.2) are joined into a class Π_2^p. A problem is in Π_2^p if it can be solved in nondeterministic polynomial time using a one-step oracle for problems from coNP. Clearly, coNP $\subseteq \Pi_2^p$ and a fundamental conjecture states that this inclusion is proper. Under this assumption, a Π_2^p-complete problem is more difficult that any problem from coNP.

We state without proof the following theorem.

Theorem 4.4 P_8 *is* Π_2^p*-complete.* ∎

By building longer quantified expressions of the form $\forall\exists\forall\exists\ldots$ it is possible to form an hierarchy of complexity classes coNP $= \Pi_1^p \subseteq \Pi_2^p \subseteq \ldots$. Then the above conjecture suggests that every inclusion in this chain is proper. The order of quantifiers in the alternating sequence is quite essential. We have illustrated this difference above for the classes NP and coNP, i.e., for problems which ask for the existence of an object (NP) and problems that claim something for all objects from a certain set (coNP). Reversing the order ($\exists\forall\exists\forall\ldots$), we get an hierarchy of complementary classes NP $= \Sigma_1^p \subseteq \Sigma_2^p \subseteq \ldots$. The general form of the fundamental conjecture is that every inclusion in either chain is proper. The next result that we are going to discuss is related to this conjecture.

In Problem P_1 we have assumed that the code is a part of the input, i.e., that the matrix H is supplied with every particular instance. Relaxing this assumption, let us suppose that it is given once and forever and we can perform an arbitrarily large preprocessing.

P_9 (Error weight with preprocessing). Given a vector $\boldsymbol{s}$ and a number w, find out whether there is a vector $\boldsymbol{y}$ of weight $\le w$ with this syndrome: $H\boldsymbol{y}^T = \boldsymbol{s}^T$.

Theorem 4.5 *A polynomial algorithm for* P_9 *would imply that* $\bigcup_{i\ge 1}\Sigma_i^p = \Sigma_2^p$.

Proof: We shall give a reduction from K_1 to P_9 with the property that every instance U of K_1 can be polynomially transformed to an instance of P_9 in such a way that U has a matching if and only if the question of P_9 can be answered in the affirmative. That this implies our claim will then follow from a general structure theorem on nonuniform complexity, discussed for instance in [33].

Let W be a t-set, $N = t^3$, and let us number the triples $u_j \in W^3$ in a certain fixed way. Form a $((3t + 2N) \times 3N)$ matrix

$$H = \begin{bmatrix} & B & \\ & I_k & I_k \\ I_k & & I_k \end{bmatrix},$$

where B is a $3t \times N$ incidence matrix of the set W^3. Let $\boldsymbol{s} = (11\ldots1, \boldsymbol{z}_U, \boldsymbol{z}_U)$ be a $(3t + 2N)$-vector, where $\boldsymbol{z}_U = (\chi(u_j \in U), 1 \le j \le N)$ is the incidence vector of the set U. Finally, choose $w = t + |U|$.

If U contains a matching M and $\boldsymbol{z}_M$ is its incidence vector, then the vector $\boldsymbol{y} = (\boldsymbol{z}_M, \boldsymbol{z}_M, \boldsymbol{z}_{U\setminus M})$ satisfies the conditions of P_9. Conversely, if there is a vector $\boldsymbol{y} = (\boldsymbol{y}_0, \boldsymbol{y}_1, \boldsymbol{y}_2)$ with $\mathsf{wt}(\boldsymbol{y}) \le w$ and $H\boldsymbol{y}^T = \boldsymbol{s}^T$, then $B\boldsymbol{y}_1^T = (1\ldots1)^T$. However, $\mathsf{wt}(\boldsymbol{y}_0) + \mathsf{wt}(\boldsymbol{y}_2) \ge |U|$ whence $\mathsf{wt}(\boldsymbol{y}_1) \le w - |U| = t$ and therefore,

$\mathsf{wt}(\boldsymbol{y}_1) = t$. This implies, as in the proof of Thm. 4.1, Part (a), that the nonzero coordinates of $\boldsymbol{y}_1$ equal one and as an incidence vector, it defines a matching $M \subset W^3$. Finally, the identity $\boldsymbol{y}_1 + \boldsymbol{y}_2 = \boldsymbol{z}_U$ locates the ones of $\boldsymbol{y}_1$ within the coordinates corresponding to U, that is, $M \subseteq U$. ∎

Remark 4.2 Note the difference between this reduction and the one in part (a) of Theorem 4.1. There H depended on the instance of K_1, here it is fixed, and the only influence of a particular instance of the matching problem is through $\boldsymbol{y}$ and w.

We conclude our discussion with another NP-completeness related result that deals with approximating the solution to Problem P_1. From the proof of Theorem 4.1(a), we see that this problem remains NP-complete even if in its statement we ask about exact equality.

Let $\beta > 1$. An algorithm is said to approximate the solution of a problem (in our case, the weight of error) if its output is within the interval $[w, \beta w]$, w being the exact answer. If β is a constant, one speaks on approximating the weight of error up to a constant factor. If $\beta = 2^{\log^{0.5-\epsilon} n}$, $\epsilon > 0$, then it is said that an algorithm approximates a solution up to a *large factor*.

Approximating the weight of an error even within any large factor is likely to be difficult. In the following theorem we use one technical concept. A function f is *almost*-NP-*hard* if using f as an oracle, we could solve any NP problem in deterministic *quasi-polynomial* time, i.e., in time $\mathcal{O}(n^{poly(\log n)})$.

Theorem 4.6 *Approximating the weight of an error within any constant factor is NP-hard and within any large factor is almost-NP-hard.*

A simple way to prove a weaker nonapproximability result is to reduce Problem K_3 to P_1. Note that this reduction preserves approximations. Therefore, if K_3 is NP-hard to approximate up to a certain constant, then so is P_1. In particular, finding a 98% approximation in the maximal cut problem is known to be NP-hard. Therefore, the same holds for Problem P_1.

4.1 Notes

A comprehensive account of logical foundations related to the complexity classes is given in Garey and Johnson [71]. The NP-completeness proofs of problems K_1–K_3 are also found there.

The first paper on NP-complete problems related to coding was Berlekamp et al. [28]. The results of this paper correspond to parts (a) and (c) of our Theorem 4.1. That Weight of error is NP-complete for arbitrary q and for multilevel codes was observed in Barg [19].

Part (b) is a recent breakthrough due to Vardy [161]. The NP-completeness of the Minimal weight problem has been conjectured in Berlekamp et al. [28] and has resisted all attacks for 19 years despite repeated calls for a proof; see Johnson [88]. Some partial results were proved in Ntafos and Hakimi [124]. The

NP-completeness of deciding whether a binary code has a vector of weight $n/2$ has been proved by Calderbank and Shor (see [47]). The same result for ternary codes and weight n is from Barg [19].

For the NP-completeness of P_5, P_6, P_7 see Frances and Litman [65], Horn and Kschischang [84], and Kratochvil [100], respectively. An easy reduction for P_6 follows from part (b) of Theorem 4.1; see Vardy [161].

The construction of codes from graphs was introduced in Calabi [34], Hakimi [81]. Properties of classical linear spaces in graphs and their combinatorial evolution form the subject of matroid theory; see Welsh [165]. Many problems that are difficult for general linear codes, are polynomial for cycle codes of graphs. Examples are decoding, see Ntafos and Hakimi [124], finding a basis of minimal total weight, see Chickering et al. [40], Horton [85], and computing the covering radius, see Frank [66].

Theorem 4.4 is due to McLoughlin [122]. The complexity of the Weight of error problem with preprocessing (P_9) was proved in Bruck and Naor [33]. Their reduction is from K_3. The proof that we give, following Lobstein [24, pp. 121-123], has the advantage of being valid for any fixed code alphabet.

The nonapproximability theorem (Theorem 4.6) and the surrounding discussion is from Arora et al. [11], Stern [151]. Stern [151] constructs a sequence of approximation preserving reductions from 3-SAT (cf. Problem K_2). Arora et al. [11] also prove similar nonapproximability results for integral lattices in $\mathbf{R}^n$. The general theory of problems that are hard to approximate is found in Papadimitriou and Yannakakis [126],

Some NP-complete problems related to constrained codes are found in Ashley et al. [14]. In [129] Petrank and Roth study the complexity of deciding whether two linear codes are equivalent. An algorithm for establishing equivalence of two binary codes with no nontrivial automorphisms is presented in Sendrier [138].

* * *

In this chapter we studied the use of complexity theory in coding. However, complexity theory itself also profits from coding methods. Since the technique is not related to codes and the proofs are long, we have decided not to include these results. They are covered in Arora [9], [10], Babai [17], Feigenbaum [62], Kiwi [95].

In conclusion we would like to mention two research areas related to our topic.

First, as is well known and partly shown above, most best results in coding theory are achieved by averaging over the ensemble of random codes. Therefore, it is natural to ask whether basic problems, first and foremost, the decoding, are NP-complete *on the average* in the sense of the definitions in Levin [109], Gurevich [80].

The second area is related to algorithms for a "quantum computer," i.e., computations that operate with states of certain quantum systems. Shor [140] showed that using this hypothetical computing device it is possible to greatly expedite algorithmic solutions of some notoriously difficult problems. The existence of asymptotically good quantum codes has been proved in Calderbank and Shor [35]. Therefore, a most natural question is whether it is possible to construct simple quantum computations for the decoding of these quantum codes or of classical codes with a better performance than that of the classical algorithms discussed in this chapter.

Acknowledgment. A substantial part of this chapter was written during my stay at the Faculty of Mathematics and Computing Science of the Technical University of Eindhoven. I wish to thank Prof. H. C. A. van Tilborg for granting me an opportunity to spend the years 1995-96 at the faculty and for his support.

Discussions with Shiyu Zhou were helpful in improving the presentation of a number of results in Section 2.

References

[1] E. Agrell, "Voronoi regions for binary linear block codes," *IEEE Trans. Inform. Theory*, **IT-42** (1) (1996), 310–316.

[2] R. Ahlswede, L. A. Bassalygo, and M. S. Pinsker, "Asymptotically optimal binary codes of polynomial complexity correcting localized errors," *Problems of Info. Trans.,* **31** (2) (1995), 76–83 and 162–168 [4].

[3] A. V. Aho, J. E. Hoprcroft, and J. D. Ullman *The Design and Analysis of Computer Algorithms*, London: Addison-Wesley (1974).

[4] A. Albanese, J. Blömer, J. Edmonds, M. Luby, and M. Sudan, "Priority encoding transmission," *IEEE Trans. Inform. Theory*, **IT-42** (6) (1996), 1737–1747. Preliminary version in *Proc. 35th Annual Sympos. on Foundations of Computer Science* (FOCS'94), IEEE Press (1994), pp. 604–613.

[5] N. Alon, J. Bruck, J. Naor, M. Naor, and R. Roth, "Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs," *IEEE Trans. Inform. Theory,* **IT-38** (2) (1992), 509–516.

[6] N. Alon and F. R. K. Chung, "Explicit constructions of linear sized tolerant networks," *Discrete Mathematics*, **72** (1988), 15–19.

[7] N. Alon and M. Luby, "A linear time erasure resilient code with nearly optimal recovery," *IEEE Trans. Inform. Theory,* **IT-42** (6) (1996), 1732–1736. Preliminary version in N. Alon, J. Edmonds, and M. Luby, *Proc. 36th Annual Sympos. on Foundations of Computer Science* (FOCS'95), IEEE Press (1995), pp. 512–519.

[8] N. Alon and J. Spencer, *The Probabilistic Method*, Chichester: Wiley-Interscience (1992).

[9] S. Arora, *Probabilistic Checking of Proofs and Hardness of Approximation Problems*, Dept. of Computer Science, Princeton University, Techn. Rep. CS-TR-476-94 (1994), 158p., available from `http://www.eccc.uni-trier.de/eccc/`.

[4] The first page range refers to the Russian original (Проблемы передачи информации) published by the Russian Academy of Sciences and the second one to the English translation (Problems of Information Transmission) published by Plenum Press.