Electronic Colloquium on Computational Complexity, Report No. 96 (2014)

Solving Linear Equations Parameterized by Hamming Weight*

V. Arvind[a] Johannes Köbler[b] Sebastian Kuhnert[b] Jacobo Torán[c]

[a] Institute of Mathematical Sciences, Chennai, India
arvind@imsc.res.in

[b] Institut für Informatik, Humboldt-Universität zu Berlin, Germany
{koebler,kuhnert}@informatik.hu-berlin.de

[c] Institut für Theoretische Informatik, Universität Ulm, Germany
jacobo.toran@uni-ulm.de

Abstract

Given a system of linear equations $Ax = b$ over the binary field $\mathbb{F}_2$ and an integer $t \geq 1$, we study the following three algorithmic problems:

1. Does $Ax = b$ have a solution of weight at most t?
2. Does $Ax = b$ have a solution of weight exactly t?
3. Does $Ax = b$ have a solution of weight at least t?

We investigate the parameterized complexity of these problems with t as parameter. A special aspect of our study is to show how the maximum multiplicity k of variable occurrences in $Ax = b$ influences the complexity of the problem. We show a sharp dichotomy: for each $k \geq 3$ the first two problems are W[1]-hard (which strengthens and simplifies a result of Downey et al. [SIAM J. Comput. 29, 1999]). For $k = 2$, the problems turn out to be intimately connected to well-studied matching problems and can be efficiently solved using matching algorithms.

1 Introduction

There are well known efficient methods, like Gaussian elimination, to solve systems of linear equations $Ax = b$ over $\mathbb{F}_2$. The problem becomes harder when we are seeking for a solution u with certain constraints placed on its Hamming weight $\mathrm{wt}(u)$. This problem has been extensively studied in the context of error correcting codes as it is closely related to the minimum weight codeword problem: given a linear code defined by $Ax = 0$, what is the minimum weight of a non-zero codeword in it? This problem is known to be NP-hard [Var97a], and even hard to approximate within any constant factor, assuming NP $\neq$ RP [DMS03]. There are three related decision problems of interest for systems of linear equations $Ax = b$ over $\mathbb{F}_2$:

*An extended abstract of this article appears in the proceedings of IPEC 2014. This work was supported by the Alexander von Humboldt Foundation in its research group linkage program. The third author was supported by DFG grant KO 1053/7-2.

ISSN 1433-8092

1. $(A, b, t) \in \text{LinEq}_{\leq}$ if $Ax = b$ admits a solution u with $1 \leq \text{wt}(u) \leq t$.
2. $(A, b, t) \in \text{LinEq}_{=}$ if $Ax = b$ admits a solution u with $\text{wt}(u) = t$.
3. $(A, b, t) \in \text{LinEq}_{\geq}$ if $Ax = b$ admits a solution u with $\text{wt}(u) \geq t$.

Berlekamp et al. [BMT78] show that both $\text{LinEq}_{\leq}$ and $\text{LinEq}_{=}$ are NP-complete. When b is the all zeros vector, $\text{LinEq}_{\leq}$ is the minimum weight codeword problem which is NP-hard [Var97a], as already mentioned. Ntafos et al. [NH81] show that $\text{LinEq}_{\geq}$ is NP-complete (also see [Var97b]). See [Joh05] for a nice discussion of these hardness results.

When the weight threshold t is considered as parameter, we denote the resulting parameterized versions of these problems by $\text{LinEq}_{\leq,t}$, $\text{LinEq}_{=,t}$ and $\text{LinEq}_{\leq,t}$, respectively. Downey et al. [DFV$^+$99] studied special cases of $\text{LinEq}_{\leq,t}$ and $\text{LinEq}_{=,t}$: when the vector b is either the all zeros vector or b is the all ones vector. These two special cases are called Even and Odd, respectively, for the weight at most t version. As argued in Remark 2.1 below, all other cases for vector b are in fact equivalent to either one of them. Observe that in the Even case, setting all variables to zero is always a solution; this is why $\text{LinEq}_{\leq,t}$ and Even ask for solutions of weight at least 1. For the weight exactly t version, the problems are called Exact Even and Exact Odd. It turns out via a complicated proof in [DFV$^+$99], that Odd, Exact Odd and Exact Even are W[1]-hard. Whether Even is also W[1]-hard remains open. The problem $\text{LinEq}_{\geq,t}$, to our knowledge, has not been studied in the parameterized setting before. We show in Section 6 that in contrast to the other two, this problem is in FPT.

Our main contribution is the study of $\text{LinEq}_{\leq,t}$ and $\text{LinEq}_{=,t}$ in the light of some additional parameters: the maximum number k of occurrences of a variable in the system and the maximum size s of an equation. When k and s are restricted or used as an additional parameter, we denote this by an additional subscript to the respective problem. For example, k is treated as an additional parameter (besides t) in $\text{LinEq}_{\leq,t,k}$, and bounded by $k_{\max}$ in $\text{LinEq}_{\leq,t,k\leq k_{\max}}$.

Concerning parameter k, we show a sharp dichotomy in the complexity of the problem. We prove that $\text{LinEq}_{\leq,t,k\leq k_{\max}}$ and $\text{LinEq}_{=,t,k\leq k_{\max}}$ are fpt tractable for $k_{\max} \leq 2$, whereas for each $k_{\max} \geq 3$, both problems are W[1]-hard. For the weight exactly t version, the hardness also holds for $b = 0$, while this case remains open for the weight at most t version.

Our hardness proof is a direct reduction from the parameterized clique problem. It strengthens and is much simpler than the proofs in [DF99, DFV$^+$99] that (for unbounded occurrence multiplicity of the variables) go over a series of reductions running into nearly 10 pages. Furthermore, it gives alternative proofs of hardness for their results for Exact Even, Odd and Exact Odd.

For $k_{\max} = 2$, we establish a connection between the equation systems and graph matching problems. We show that $\text{LinEq}_{\leq,k\leq 2}$ and $\text{LinEq}_{\geq,k\leq 2}$ are solvable in polynomial time, while $\text{LinEq}_{=,k\leq 2}$ is solvable in randomized NC (RNC). The latter result follows from an interesting connection between $\text{LinEq}_{=,k\leq 2}$ and Red-Blue Perfect Matching [PY82] (also known as Exact Matching), which is known to be solvable in RNC [MVV87] but not known to be in P. We show in Section 4 that both problems are equivalent under logarithmic space reductions. Hence, proving that $\text{LinEq}_{=,k=2}$ is in P would imply that Red-Blue Perfect Matching is also in P, solving a long standing open question. Further we show that $\text{LinEq}_{=,t,k\leq 2}$ is fixed parameter tractable.

If the maximum equation size s is an additional parameter then, as we show in Section 5, all three problems are fixed parameter tractable. In particular, if $s \leq 2$ then even the parameter-free versions of all three problems are solvable in logarithmic space. A summary of the results is given in Table 1.

Table 1: Summary of results.

	parameter/restriction list α					
Problem	t	$t, k \leq 3$	$t, k \leq 2$	$k \leq 2$	s, t	$s \leq 2$
$\text{LinEq}_{\leq,\alpha}$	W[1]-hard [DFV+99]	W[1]-hard Thm. 3.1	FPT Thm. 4.1	P Thm. 4.1	FPT Thm. 5.1	L-complete Thm. 5.5
$\text{LinEq}_{=,\alpha}$	W[1]-hard[a] [DFV+99]	W[1]-hard[a] Thm. 3.1	FPT Thm. 4.6	RNC Cor. 4.4	FPT Thm. 5.4	L-complete Thm. 5.5
$\text{LinEq}_{\geq,\alpha}$	FPT Thm. 6.1	FPT Thm. 6.1	FPT Thm. 6.1	P Thm. 4.2	FPT Thm. 6.1	L-complete Thm. 5.5

[a] Remains W[1]-hard for $b = 0$.

Our fpt algorithms involve standard techniques like color coding (Theorems 4.6 and 5.4), depth-bounded search trees (Theorem 5.1), and reduction to problem kernels (Theorem 6.1).

2 Basic transformations

In this section we describe some basic transformations between various linear equation system problems. We first remark that while there is a trivial disjunctive reduction from $\text{LinEq}_{\leq}$ to $\text{LinEq}_{=}$ that maps $(A, b, t) \mapsto \{(A, b, t') : 1 \leq t' \leq t\}$, there is no trivial reduction for the converse direction. Indeed, there can be many solutions of different weights, and thus $(A, b, t) \in \text{LinEq}_{=}$ does not imply $(A, b, t-1) \notin \text{LinEq}_{\leq}$.

We now turn to the different possibilities for the vector b. The following remark shows that (Exact) Even fpt reduces to (Exact) Odd, taking the focus away from the "mixed" case.

Remark 2.1. A system of linear equations $Ax = b$ over $\mathbb{F}_2$ can be easily transformed into an equivalent system $A'x' = 1$: Add a new variable x_0 and equation $x_0 = 1$. Convert each 0-equation into an equivalent 1-equation by adding x_0 to it. Then $Ax = b$ has a weight t solution if and only if $A'x' = b'$ has a weight $t + 1$ solution.

In the non-parameterized setting, the "mixed" case is also reducible to Even.

Remark 2.2. A system $Ax = b$ over $\mathbb{F}_2$ with n variables can be transformed into a system $A'x' = 0$ such that $Ax = b$ has a weight t solution if and only if $A'x' = 0$ has a weight $t + n + 1$ solution: add a new variable x_0 and a new equation $x_0 = 1$. Convert each 1-equation into an equivalent 0-equation by adding x_0 to it. Introduce n new variables $y_1, \ldots, y_n$ and replace the $x_0 = 1$ equation by the equations $x_0 + y_i = 0$ for $i = 1, \ldots, n$.

The following lemma shows that any instance (A, b, t) can be easily transformed into an equivalent instance of the form (A', b', kt) where each variable occurs at most three times. The idea is to introduce k copies of each variable, to replace each occurrence with a different copy, and to force the copies to take equal values using additional equations.

Lemma 2.3. *Let $Ax = b$ be a system of linear equations and let k be the maximum number of occurrences of any variable in it. Then an equivalent system $A'y = b'$ with at most three occurrences of each variable can be constructed in polynomial time, where equivalent means that*

a weight t solution for $Ax = b$ induces a weight kt solution for $A'y = b'$ and any weight t' solution for $A'y = b'$ induces a weight t'/k solution for $Ax = b$.

Proof. For each variable x_i in $Ax = b$, include k variables $y_{i,1}, \ldots, y_{i,k}$ in the constructed system $A'y = b'$, and force them to take the same value by adding the equations $y_{i,j} \oplus y_{i,j+1} = 0$ for $1 \leq j < k$. Additionally, modify $Ax = b$ by replacing the jth occurrence of x_i by $y_{i,j}$ and add the resulting equations to $A'y = b'$.

Any solution x of weight t for $Ax = b$ induces a solution y of weight kt for $A'y = b'$; this solution is defined by $y_{i,j} = x_i$. Conversely, in any solution y of $A'y = b'$ the newly added equations enforce $y_{i,j} = y_{i,j'}$. Thus y has weight kt for some t, and x defined by $x_i = y_{i,1}$ is a solution of weight t for $Ax = b$. □

As a consequence of Lemma 2.3 we can reduce all linear equation problems to the case $k \leq 3$. For example, it follows that $\text{LinEq}_{\leq,t,k}$ is fpt reducible to $\text{LinEq}_{\leq,t,k\leq 3}$ and that $\text{LinEq}_{=}$ is polynomial-time reducible to $\text{LinEq}_{=,k\leq 3}$.

To facilitate the presentation of some of our proofs, it is convenient to consider a more general problem in which each variable x_i occurring in $Ax = b$ has a positive integer weight w_i (encoded in unary). The weight t of a solution is the sum of the weights of the variables assigned value 1. The next lemma shows that the weighted case is polynomial-time reducible to the unweighted case (where all variables have weight 1).

Lemma 2.4. *Let $Ax = b$ be a system of linear equations with variable weights given in unary. Then an equivalent unweighted system $A'y = b'$ can be constructed in polynomial time, where equivalent means that a weight t solution for $Ax = b$ induces a weight t solution for $A'y = b'$ and vice versa. Moreover,*

(i) if all variables of $Ax = b$ occur in exactly 2 equations then all variables of $A'y = b'$ occur in exactly 2 equations.

(ii) if all variables of $Ax = b$ occur in exactly 3 equations and have odd weight, then all variables of $A'y = b'$ occur in exactly 3 equations.

Proof. For each variable x_i occurring in the input system $Ax = b$ of weight $w_i > 1$, we include w_i new variables $y_{i,1}, \ldots, y_{i,w_i}$ for the system $A'y = b'$. Pick the first equation of $Ax = b$ containing variable x_i and substitute x_i by y_{i,w_i} in it. Substitute the remaining occurrences of x_i in $Ax = b$ by $y_{i,1}$. Additionally, for $1 \leq j < w_i$ include a new equation $y_{i,j} \oplus y_{i,j+1} = 0$.

This set of equations defines $A'y = b'$. The definition ensures that in any solution to $A'y = b'$, for each i the variables $y_{i,1}, \ldots, y_{i,w_i}$ all take the same value. Thus, if any $y_{i,j}$ takes the value 1 then the entire set of these variables make a net contribution of weight w_i and are thus equivalent to the original weighted variable x_i in $Ax = b$.

Part (i) follows directly since each new variable appears exactly twice in this case. For Part (ii), we modify the reduction and additionally include the equation $y_{i,2} \oplus \ldots \oplus y_{i,w_i} = 0$. This enforces every variable to appear exactly three times. Moreover since $w_i - 1$ is even, the additional equation is implied by the other equations and hence does not affect the overall feasibility. □

We close this section by giving a useful graph theoretical interpretation of the linear equation problems.

Remark 2.5. We will consider systems $Ax = b$ with m variables and n equations, that is, A is an $n \times m$ matrix over $\mathbb{F}_2$. It will be convenient to interpret A as the incidence matrix

of a hypergraph. With this interpretation each equation becomes a vertex and each variable becomes a hyperedge that consists of all vertices (equations) in which it occurs. Note that this might give a multi-hypergraph since different variables might occur in exactly the same equations.

A vertex v_j will be called *even* if $b_j = 0$, and *odd* if $b_j = 1$. A solution of weight t is a selection of t hyperedges that covers each even vertex with an even number of hyperedges and each odd vertex with an odd number of hyperedges. Observe that in the case that every variable appears exactly twice in the equation system we get a standard multi-graph in which each edge connects two vertices.

3 At most three occurrences of each variable

This section is devoted to our main result showing that $\textsc{LinEq}_{\leq,t,k\leq k_{\max}}$ and $\textsc{LinEq}_{=,t,k\leq k_{\max}}$ are $\mathsf{W}[1]$-hard for each $k_{\max} \geq 3$.

Theorem 3.1. $\textsc{LinEq}_{\leq,t,k\leq 3}$ *and* $\textsc{LinEq}_{=,t,k\leq 3}$ *are* $\mathsf{W}[1]$*-hard. The hardness even holds for the case that each variable occurs exactly three times.*

To prove Theorem 3.1 we make use of the hypergraph interpretation of a linear system of equations as explained in Remark 2.5. The key step is the design of a selector gadget, which can be used to select a specified number of vertices from a given vertex set $V = \{v_1, \ldots, v_n\}$. Besides the vertices in V, the gadget contains a special *start vertex* a and a set U of internal vertices, i.e., the vertex set is $V \cup U \cup \{a\}$. We say that a set $\mathcal{S}$ of hyperedges *activates* a vertex if $\mathcal{S}$ covers it an odd number of times. Further, we call $\mathcal{S}$ *admissible* if it activates the start vertex a but no internal vertex in U. Using this notation we will construct the hyperedge set $\mathcal{E}$ of the gadget $\mathrm{Sel}^a_{k,V}$ in such a way that the minimal admissible subsets $\mathcal{S}$ of $\mathcal{E}$ activate besides a exactly the k-element subsets of V.

The construction of $\mathrm{Sel}^a_{k,V}$ is illustrated in Figure 1. The set of internal vertices is

$$U = \{u_{\ell,i} \,:\, 1 < \ell < k \wedge \ell \leq i \leq n - k + \ell\}.$$

The intended semantics is that if a minimal admissible subset $\mathcal{S}$ covers the vertex $u_{\ell,i}$, then v_i is the ℓth smallest of the activated vertices from V. The hyperedge set of $\mathrm{Sel}^a_{k,V}$ is $\mathcal{E} = \bigcup_{\ell=1}^{k-1} \mathcal{E}_k$, where

$$\begin{aligned}
\mathcal{E}_1 &= \big\{\{a, v_i, u_{2,i'}\} \,:\, 1 < i < i' \leq n - k + 2\big\},\\
\mathcal{E}_\ell &= \big\{\{u_{\ell,i}, v_i, u_{\ell+1,i'}\} \,:\, \ell \leq i < i' \leq n - k + \ell + 1\big\} \text{ for } \ell = 2, \ldots, k-2,\\
\mathcal{E}_{k-1} &= \big\{\{u_{k-1,i}, v_i, v_{i'}\} \,:\, n - k \leq i < i' \leq n - 1\big\},
\end{aligned}$$

and the hyperedges in $\mathcal{E}_\ell$ are called *level* ℓ *hyperedges* for $\ell = 1, \ldots, k-1$. In the weighted version $\mathrm{Sel}^{a,w}_{k,V}$ of the gadget, all its hyperedges have weight w.

Lemma 3.2. *Let $V = \{v_1, \ldots, v_n\}$ and let k and w be positive integers. For any subset $W \subseteq V$ of size k, there is an admissible set $\mathcal{S} \subseteq \mathcal{E}$ of weight $(k-1)w$ for the selector gadget $\mathrm{Sel}^{a,w}_{k,V}$ that activates exactly a and the vertices in W. Moreover, any admissible set $\mathcal{S} \subseteq \mathcal{E}$ of weight less than $(k+1)w$ for $\mathrm{Sel}^{a,w}_{k,V}$ has weight exactly $(k-1)w$ and activates exactly k of the vertices in V.*

Proof. For $W = \{v_{i_1}, \ldots, v_{i_k}\}$ with $i_1 < \cdots < i_k$, consider the set that consists of the hyperedge $\{a, v_{i_1}, u_{2,i_2}\}$, the hyperedges $\{u_{\ell,i_\ell}, v_{i_\ell}, u_{\ell+1,i_{\ell+1}}\}$ for $1 < \ell < k-1$, and the hyperedge $\{u_{k-1,i_{k-1}}, v_{i_{k-1}}, v_{i_k}\}$. This set is admissible, has weight $(k-1)w$ and activates exactly the vertex a and the vertices in W.

To prove the *moreover* part, a straight-forward induction over $\ell = 1, \ldots, k-1$ shows that any admissible set $\mathcal{S}$ of hyperedges must contain an odd number of level ℓ hyperedges. Thus any such set $\mathcal{S}$ of weight less than $(k+1)w$ contains exactly one hyperedge from each level, implying that $\mathcal{S}$ has weight $(k-1)w$. For $\ell = 1, \ldots, k-2$ let v_{i_ℓ} be the vertex in V covered by the level ℓ hyperedge of $\mathcal{S}$ and let $v_{i_{k-1}}$ and v_{i_k} be the two vertices in V covered by the level $k-1$ hyperedge of $\mathcal{S}$. The construction of the gadget ensures that $i_1 < i_2 < \cdots < i_k$. Hence, $\mathcal{S}$ activates exactly k vertices from V. □

Proof of Theorem 3.1. We reduce from the W[1]-complete clique problem which asks whether a given graph has a clique of size k, where k is treated as parameter. Let $G = (V, E)$ and k be the given instance. We will construct an equation system $Ax = b$ with exactly one 1-equation where each variable occurs exactly three times. Continuing with the hypergraph view, we will use several instances of the selector gadget; each uses its own internal vertices. Besides the internal vertices, the hypergraph contains one special start vertex a (which is the only odd vertex), one vertex for each graph vertex in V, and one vertex for each graph edge in E. Let $w = k^2$ if k is odd, and $w = k^2+1$ otherwise. Add the selector gadget $\mathrm{Sel}^{a,w}_{k,V}$ to the constructed hypergraph. For each graph vertex $v \in V$, let $E(v)$ denote the set of edges incident to it, and add the selector gadget $\mathrm{Sel}^{v,1}_{k-1,E(v)}$. Its role is to ensure that if v is selected by $\mathrm{Sel}^{a,w}_{k,V}$, then v must be adjacent to all other selected vertices. See Fig. 2 for an illustration of this construction. As the selector gadget has only hyperedges of size 3 and as w is odd, Lemma 2.4 implies that the weights can be removed while maintaining 3-uniformity.

We show that for $t = (k-1)w + k(k-2)$, the graph G has a clique of size k iff the equation system described by the constructed hypergraph has a solution of weight at most t.

If G contains a k-clique C, choose the admissible hyperedge subset of $\mathrm{Sel}^{a,w}_{k,V}$ that activates exactly the vertices in C. Then, for each clique vertex $v \in C$, add the admissible hyperedge subset for $\mathrm{Sel}^{v,1}_{k-1,E(v)}$ that activates $\{e \in E(v) \,:\, e \subseteq C\}$. Combining these hyperedge sets yields a solution of weight t.

Now consider any solution to the equation system of weight at most t. As $(k+1)w \geq k^3 + k^2 > k^3 - k - 1 \geq t$, Lemma 3.2 implies that this solution contains exactly $(k-1)$ hyperedges from $\mathrm{Sel}^{a,w}_{k,V}$, which activate a set C of exactly k vertices in V. As these have to be covered an even number of times, each has to be covered an odd number of times from within its selector gadget. So for each $v \in C$, the solution must include at least $k-2$ hyperedges of

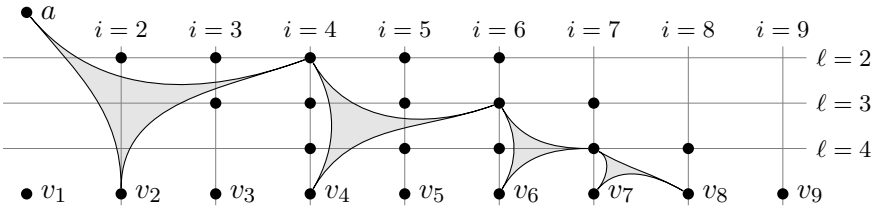

Figure 1: The vertices of the selector gadget $\mathrm{Sel}^{a}_{5,\{v_1,\ldots,v_9\}}$ and the minimal admissible subset of hyperedges that leads to the activation of $\{v_2, v_4, v_6, v_7, v_8\}$.

the selector gadget $\mathrm{Sel}^{v,1}_{k-1,E(v)}$. As this accounts for the remaining weight permitted by t, the solution cannot include further hyperedges. In particular, all vertices from E that are covered at all are graph edges that are incident to a vertex in C. As all vertices in E are even, these edges have to be covered twice by the solution, implying that every vertex $v \in C$ has $k-1$ neighbors in C, thus C is a k-clique.

Finally, note that the constructed equation system admits a solution of weight at most t if and only if it admits one of weight exactly t. □

Using the construction of Remark 2.1, we obtain alternative proofs that Odd and Exact Odd are W[1]-hard. To generalize it to Exact Even, we can multiply all weights by 2, add a new hyperedge $\{a\}$ of weight one, and ask for a solution of weight $2t+1$.

4 At most two occurrences of each variable

We show that the three problems are easier when every variable appears at most twice in $Ax = b$. It turns out that these problems can be solved using standard matching algorithms. LinEq$_{\leq,k\leq 2}$ and LinEq$_{\geq,k\leq 2}$ have deterministic polynomial-time algorithms, whereas LinEq$_{=,k\leq 2}$ has a randomized polynomial time algorithm (in fact a randomized NC algorithm).

Firstly, we note that we can easily transform the instance to the case when every variable in the system $Ax = b$ appears in *exactly* two equations without any change in the parameter t. We include the new equation $\sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}x_j = \sum_{i=1}^{n} b_i$ (obtained by adding up all equations in $Ax = b$) to obtain a new system $A'x = b'$. Note that $Ax = b$ and $A'x = b'$ have identical solutions. Furthermore, the new equation $\sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}x_j = \sum_{i=1}^{n} b_i$ has on its left-hand side precisely the sum of all single occurrence variables of the system $Ax = b$. Hence every variable in $A'x = b'$ occurs exactly twice.

As observed in Remark 2.5, when every variable appears exactly twice, the system can be represented as an undirected multi-graph whose vertex set is the set of equations and edges are the variables. Two vertices u and v are joined by an edge e iff the variable e occurs in both equations u and v. We will use this interpretation to design the algorithms in this subsection.

Theorem 4.1. LinEq$_{\leq,k\leq 2} \in$ P.

Proof. Given an instance (A, b, t) of LinEq$_{\leq,k\leq 2}$, we construct the graph G associated with $Ax = b$. The set of edges with value 1 in a solution to the system consists of an edge disjoint set of paths connecting the odd vertices by pairs and possibly some edge disjoint cycles.

If there are odd vertices we do not need to consider the cycles, since we are searching for a solution of minimum weight. Such a solution corresponds to a set of edge disjoint paths

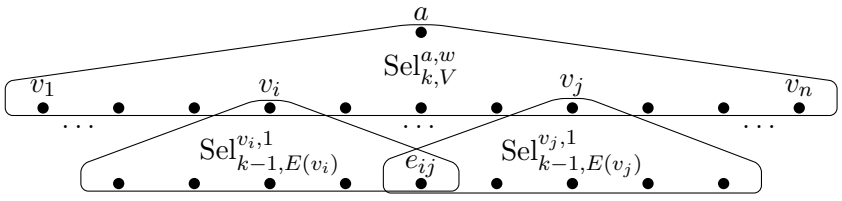

Figure 2: Hypergraph view of the equation system that has a weight t solution if and only if the underlying graph G has a k-clique. The gadgets $\mathrm{Sel}^{v_i,1}_{k-1,E(v_i)}$ and $\mathrm{Sel}^{v_j,1}_{k-1,E(v_j)}$ share the vertex e_{ij} iff the graph edge e_{ij} connects the graph vertices v_i and v_j.

of minimum total length pairing the odd vertices. This can be obtained by computing the minimum distance between all pairs of odd vertices in the graph. With this we can construct a weighted clique in the following way: each vertex in the clique represents an odd vertex in the graph. The edge between two clique vertices is weighted with the minimum distance between the corresponding odd vertices in the original graph. We claim that a perfect matching with minimum weight in the clique defines a solution of minimum weight in the system. To see this, observe that if two edges $\{a_1, a_2\}$ and $\{b_1, b_2\}$ in the perfect matching of minimum weight would correspond to paths that share at least one edge in G, then the total length of the shortest paths between a_1 and one of the b-vertices and a_2 and the other b-vertex would be smaller than $d(a_1, a_2) + d(b_1, b_2)$ since the new paths would not contain the common edge. This implies that a perfect matching of minimum weight corresponds to a minimum weight solution of the system. Since minimum weight perfect matching can be solved in polynomial time, the result follows.

When all vertices are even, we need to ensure that at least one variable is set to 1. In this case, a minimum weight non-trivial solution is just a cycle in G with a minimum number of edges. This also can be computed in polynomial time. □

Using similar ideas we can show that the weight at least t version of the problem can also be solved in polynomial time.

Theorem 4.2. $\text{LinEq}_{\geq,k\leq 2} \in \mathsf{P}$.

Proof. Given an instance (A, b, t) of $\text{LinEq}_{\geq,k\leq 2}$, we interpret $Ax = b$ as a graph $G = (V, E)$ as in the previous result. W.l.o.g. we can suppose that all vertices have degree at least 2. For $v \in V$ let $\text{pd}(v)$ be the parity of its degree and $\text{peq}(v)$ be the parity of the corresponding equation, i.e. $\text{peq}(v) = 1$ if v corresponds to an odd equation and $\text{peq}(v) = 0$ otherwise.

Let B be the set of vertices whose degree parity does not coincide with $\text{peq}(v)$. That is, $B = \{v \in V \,:\, \text{pd}(v) \neq \text{peq}(v)\}$. If $|B|$ is odd then there is no solution to the system. In order to see this, observe that since every edge appears in exactly two equations the system has a solution only if the set of vertices with $\text{peq}(v) = 1$ is even. We can partition the set of vertices v with $\text{peq}(v) = 1$ into two sets, those having even degree and those having odd degree. This means:

$$\left|\{v \in V \,:\, \text{peq}(v) = 1 \wedge \text{pd}(v) = 1\}\right| \equiv \left|\{v \in V \,:\, \text{peq}(v) = 1 \wedge \text{pd}(v) = 0\}\right| \mod 2$$

Since in every graph the set of vertices with odd degree is even, we can again partition this set into those vertices in odd equations and those in even equations and we get:

$$\left|\{v \in V \,:\, \text{pd}(v) = 1 \wedge \text{peq}(v) = 1\}\right| \equiv \left|\{v \in V \,:\, \text{pd}(v) = 1 \wedge \text{peq}(v) = 0\}\right| \mod 2$$

Putting both congruences together we conclude that B is even.

The algorithm for finding a solution of maximum weight constructs a weighted clique with the vertices of B in which every edge is weighted with the minimum distance in G between its two endpoints and finds a minimum weight perfect matching in the clique, as in the previous theorem. The paths in G corresponding to the edges in the minimum weight perfect matching are removed. Let us call this set E'. We claim that the remaining edges, $E \setminus E'$, are a maximum weight solution for the system. $E \setminus E'$ is a solution because after removing the edges from the matching, the parity of the degree of each vertex coincides with the parity of its equation. If there are no vertices of odd degree after removing the edges, then the graph is

Eulerian and its edges form a solution. Otherwise a path starting in a vertex with odd degree in $G' = (V, E \setminus E')$ and continuing along its edges will eventually arrive to another vertex of odd degree. Removing the edges along this path, this process can be continued until all vertices have even degree. Observe that any solution must include an edge disjoint pairing in G of all the vertices in B. This proves that the solution of the algorithm has maximum weight. □

We show next that LINEQ$_{=,k=2}$ is equivalent to RED-BLUE PERFECT MATCHING (RBPM), a problem introduced by Papadimitriou et al. [PY82]. This problem is defined in the following way: Given a graph G with blue and red edges and a number t, is there a perfect matching in G with exactly t red edges? RBPM can be solved in randomized NC [MVV87], but until now, no deterministic polynomial time algorithm for it is known. In fact, not even the parameterized version of this problem (with t as parameter) is known to lie in FPT.

Theorem 4.3. LINEQ$_{=,k\leq 2}$ *and* RBPM *are many-one equivalent under logarithmic space reductions. This also holds if* $b = 0$ *is required for the instances of* LINEQ$_{=,k\leq 2}$.

Proof. Let $Ax = b$ be a system of equations in which every variable appears exactly twice and let $G = (V, E)$ be its interpretation as a graph.

We first assume that $b = 0$. Let u be a solution of weight t. Since u selects for each vertex v an even number of all edges incident to v, u corresponds to a union of edge disjoint cycles in G with exactly t edges. Now consider the graph $G' = (V', E')$ that is obtained from G by expanding each vertex $v \in V$ with degree d_v into d_v new vertices $v_1, \ldots, v_{d_v}$ and connecting all pairs of these vertices by red edges. The original edges incident with v in G are each connected to one of the new vertices and are all colored blue (see Figure 3). Notice that in G', u corresponds to a union of vertex disjoint cycles with exactly $2t$ edges, where each cycle consists of alternating red and blue edges. Hence, the t red edges on these cycles form a matching that can be extended to a perfect matching of G' by adding all blue edges that are not lying on any cycle of u. Conversely, any perfect matching of G' with t red edges yields a solution of weight t by taking its symmetric difference with the set of all blue edges. This shows that $Ax = b$ has a solution of weight t if and only if G' has a perfect matching with t red edges.

If G has $r > 0$ odd vertices, each solution u corresponds to a union of edge disjoint cycles and paths with exactly t edges, where exactly the endpoints of the paths are odd vertices. We construct G' as before but expand each odd vertex into a red clique of size $d_v + 1$ by adding a special clique vertex v_0 that is connected via red edges to the other d_v clique vertices $v_1, \ldots, v_{d_v}$. In this graph, each solution u corresponds to a union of edge disjoint cycles and paths with exactly t blue and $t + r/2$ red edges, where exactly the endpoints of the paths are special clique vertices. Similarly to the even case, the $t + r/2$ red edges of u form a matching that can be extended to a perfect matching of G' by adding all blue edges that are not lying on any cycle or path. Conversely, any perfect matching M in G' has to match each special clique vertex via a red edge, implying that an odd number of the blue edges connected to its clique does not

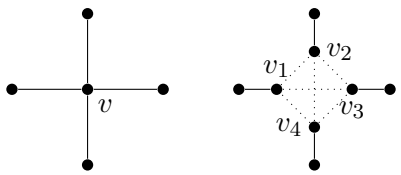

Figure 3: Expansion of a vertex v of degree 4. The dotted edges are red edges.

belong to M. Hence, if M has $t + r/2$ red edges, taking the symmetric difference of M with the set of all blue edges yields again a solution of weight t.

We reduce now the RBPM problem to a system of equations in which every variable appears twice. Given a graph $G = (V, E)$ with n vertices and m red and blue edges, and an integer $t \leq m$ we can construct a system $Ax = b$ with n odd equations, each of them corresponding to a vertex and having the incident edges as its variables. A perfect matching in G corresponds to a solution of weight $\frac{n}{2}$ that assigns exactly one variable of each equation the value 1. Let G' be the graph that is obtained from G by assigning weight n^2 to each red edge. Then G has a perfect matching with t red edges if and only if the weighted system corresponding to G' has a solution of weight exactly $\frac{n}{2} + t(n^2 - 1)$. By Lemma 2.4, the weights can be removed while maintaining the property that each variable appears exactly twice.

The reduction can be adapted to a system of equations being all even by adding a new weighted variable with weight n^3 to each of the original equations and setting all the equations to zero. Now there is a perfect matching with t red edges in G if an only if there is a solution of weight $n^4 + t(n^2 - 1) + \frac{n}{2}$. Again by Lemma 2.4 the weighted variables can be transformed into weight one variables. □

Interestingly, whereas the forward reduction also works in the parameterized setting (with t as parameter), this is not true for the converse reduction. Further, observe that the variants of Red-Blue Perfect Matching in which we ask for a matching with at most or at least t red edges are known to be in P. This provides alternative proofs for Theorems 4.1 and 4.2.

As RBPM is in randomized NC [MVV87] we obtain the following corollary.

Corollary 4.4. $\text{LinEq}_{=,k\leq 2} \in \mathsf{RNC}$.

In the first part of the proof of the previous theorem, we identify a natural graph problem as equivalent to RBPM: given an undirected graph G and an integer t, is there a set of edge disjoint cycles in G containing exactly t edges? We call this problem Exact Undirected Cycle Sum, in analogy to Exact Cycle Sum which asks the same question for a directed graph G [PY82]. The latter problem can be solved in RNC [BR95], but to our knowledge this was not previously known for the undirected version.

Corollary 4.5. Exact Undirected Cycle Sum *is many-one equivalent to* RBPM *under logspace reductions and therefore also in* RNC.

We close this section by showing that in the parameterized setting, a solution of weight t can be found in fpt time when each variable occurs at most twice.

Theorem 4.6. $\text{LinEq}_{=,t,k\leq 2} \in \mathsf{FPT}$.

Proof. Let (A, b, t) be the input instance and let $G = (V, E)$ be the corresponding graph. If $b = 0$, let u_0 be the empty solution. Otherwise, using the algorithm of Theorem 4.1, we compute a solution u_0 of minimum weight for $Ax = b$. If $|u_0| \geq t$, we are done. Otherwise, observe that every solution of $Ax = b$ can be written as a sum (modulo 2) of u_0 and some edge-disjoint cycles of G (that might overlap with u_0). To find a suitable set of cycles, we use the color coding method introduced in [AYZ95]. Each edge in u_0 receives its own unique color (recall that $|u_0| < t$). Let C_{u_0} be the set of colors of the edges in u_0. The remaining edges are colored uniformly at random using t new different colors. In case that there is a solution of weight exactly t, the probability that all the edges in the solution have different colors depends only

on t and it is at least $\frac{t!}{t^t}$. A *color pattern* for a cycle is a sequence of colors to be encountered on the cycle. Now consider each possible set C of disjoint color patterns (their number only depends on t). For any set of disjoint cycles that realizes C, the corresponding solution has weight equal to the number of colors that appear in C or in C_{u_0} but not in both. If C leads to solutions of weight t, it remains to check if each color pattern $c_1, \ldots, c_k$ in C can be realized in G. The latter can be checked dynamically by computing sets $S_i(v)$, with $v \in V$ and $0 \leq i \leq k$, such that $u \in S_i(v)$ if and only if there is a path from u to v that realizes $c_1, \ldots, c_i$. Initially, $S_0(v) = \{v\}$ for each $v \in V$. For $i \in \{1, \ldots, k\}$, the set $S_i(v)$ is the union of all $S_{i-1}(u)$ for which $\{u, v\}$ is an edge of color c_i. There is a cycle realizing $c_1, \ldots, c_k$ if and only if there is a vertex v with $v \in S_k(v)$.

The probabilistic part in the previous algorithm can be derandomized using a perfect hash family as explained in [AYZ95]. □

5 Using the equation size as an additional parameter

In this section we show that the weight at most t and the weight exactly t versions of the problem become fixed parameter tractable when we treat the maximum equation size s as an additional parameter. For the weight at least t version we show in Section 6 that even $\text{LinEq}_{\geq,t}$ is in FPT.

We call a solution $u \neq 0$ of a system $Ax = b$ *minimal* if for any solution $u' \neq 0$ with $u'_i \leq u_i$ for all i it holds that $u' = u$.

Theorem 5.1. $\text{LinEq}_{\leq,t,s} \in \mathsf{FPT}$. *Moreover, for each instance, all minimal solutions of weight at most t can be found in fpt time.*

Proof. The algorithm traverses the following search tree to find all minimal solutions of weight at most t. If $b = 0$, the first branch is to select a variable, set it to 1 and continue with the resulting system over the remaining $m - 1$ variables. This m-way branching is only needed once to avoid the trivial all zeroes solution. If $b \neq 0$ and the number of variables set to 1 so far is smaller than t, we pick the first equation with $b_j = 1$ and branch over all variables that occur in this equation. In each branch, we set the chosen variable to 1 and continue with the system over the remaining variables. As soon as all equations are satisfied by setting the remaining variables to 0 (i.e., $b = 0$), we reach at a successful leaf providing a solution of weight at most t. If already t variables have been set to 1 and $b \neq 0$, the current node is declared to be an unsuccessful leaf.

Since for every minimal solution u of weight at most t there is a path that selects at each node one more variable from u, the tree enumerates any such solution. Further, the tree can be traversed in fpt time as its depth is bounded by t and the number of its leaves is bounded by $s^{t-1}m$. □

To solve the weight exactly t case, we will again design a color coding algorithm similar to that in Theorem 4.6, where minimal solutions take the role of cycles. The following lemma shows that any solution $u \neq 0$ of a system $Ax = 0$ is the sum of disjoint minimal solutions.

Lemma 5.2. *Any solution $u \neq 0$ of a homogeneous system $Ax = 0$ over $\mathbb{F}_2$ is the sum of disjoint minimal solutions.*

Proof. Let $u \neq 0$ be any solution. If it is not minimal, let u' be a minimal solution that selects a proper subset of the variables selected by u. Then $u \oplus u'$ is also a solution, is disjoint from u', and has smaller weight than u. So an inductive argument over the weight of u gives the lemma. □

Next, we observe the following colored variant of Theorem 5.1. When the variables are colored, we say that a solution u *respects* a set C of colors if u contains exactly one variable of each color in C, and no other variables.

Lemma 5.3. *Given a system $Ax = b$ over $\mathbb{F}_2$, a coloring of its variables and a set C of colors, all minimal solutions that respect C can be found in fpt time when $|C|$ and the maximum size s of the equations are treated as parameters.*

Proof. The algorithm proceeds as the one of Theorem 5.1 with the following modifications: At each branching, it only considers variables that have a color in C that has not yet been used. And at each leaf it additionally checks that for each color in C a variable has been selected. □

Now, the following theorem can be proved along the same lines as Theorem 4.6.

Theorem 5.4. $\text{LinEq}_{=,t,s} \in \mathsf{FPT}$.

Proof. Let (A, b, t) be the given instance, and let u_0 be a solution of minimum weight for $Ax = b$, obtained using the algorithm of Theorem 5.1 (or by setting $u_0 = 0$ in case $b = 0$). If $\text{wt}(u_0) \geq t$, we are done. Otherwise we color each variable selected by u_0 with a unique color, and the remaining variables uniformly at random using t additional colors. Any solution to $Ax = b$ can be written as $u_0 \oplus u_1$, where u_1 is a solution to $Ax = 0$. By Lemma 5.2, we can decompose each such u_1 as $u_1 = \bigoplus_{u \in U} u$, where U is a set of disjoint minimal solutions to $Ax = 0$. Again, the probability that all variables in a solution to $Ax = 0$ receive distinct colors depends only on t and is at least $\frac{t!}{t^t}$. A *color pattern* is a set of colors. Iterating over all sets C of color patterns (their number only depends on t), we check if there are exactly t colors that occur in C or the variables selected by u_0 but not both, and if each color pattern in C is respected by some minimal solution to $Ax = 0$. The latter is possible by Lemma 5.3. It remains to derandomize this algorithm using a perfect hash family as in [AYZ95]. □

We close this section by considering restrictions on the parameter s. In the case $s \leq 2$ we can assume that all equations contain exactly 2 variables. Let G be the graph that has one vertex for each literal and an edge between each pair of literals that are forced to be equivalent by some equation. If the system is satisfiable, the connected components of G can be grouped into pairs of complementary equivalence classes. Define the size of an equivalence class as the number of positive literals in it. By choosing the one of smaller/larger size from each pair of complementary equivalence classes gives a minimum/maximum weight solution. Furthermore, the weight exactly t version reduces to Unary Subset Sum: It suffices to check whether a subset of the size differences of all pairs sums up to t minus the size of a minimum solution. As both Undirected Connectivity and Unary Subset Sum can be solved in logarithmic space [Rei08, EJT10], we have the upper bounds of the following theorem.

Theorem 5.5. $\text{LinEq}_{\leq,s\leq 2}$, $\text{LinEq}_{=,s\leq 2}$, *and* $\text{LinEq}_{\geq,s\leq 2}$ *are all* L*-complete.*

The lower bounds follow from the fact that satisfiability of conjunctions of parities of size at most 2 is hard for L [JLL76]. Indeed, let $Ax = b$ be a system on n variables with $s \leq 2$. For

each variable x_i, add a second variable x'_i and the equation $x_i \oplus x'_i = 1$. This transformation preserves satisfiability, and every solution of the resulting system has weight n.

Complementing Theorem 5.5, the following lemma shows that the general case can be reduced to the case $s \leq 3$, implying that all three problems remain NP-hard under this restriction.

Lemma 5.6. *Given a system of linear equations $Ax = b$ and a number t we can construct a new system $A'y = b'$ with equations of size at most 3 and a number t' so that there is a solution of weight t for the first system if and only if there is a solution of weight t' for the second one.*

Proof. An equation of size s, $x_1 \oplus \cdots \oplus x_s = b$ for $b \in \{0, 1\}$ can be split adding two new variables y_1 and y_2, into the three equations $x_1 \oplus x_2 \oplus y_1 = 0$, $y_2 \oplus \cdots \oplus x_k = \overline{b}$ and $y_1 \oplus y_2 = 1$, of sizes 3, $k-1$, and 2, respectively. Since in any solution exactly one of the variables y_1 and y_2 is set to 1, the weight of a solution in the new system increases by one. The splitting step can be repeated until all equations have size at most three, increasing the weight for the new solutions accordingly. □

Corollary 5.7. $\textsc{LinEq}_{\leq,s\leq 3}$, $\textsc{LinEq}_{=,s\leq 3}$, *and* $\textsc{LinEq}_{\geq,s\leq 3}$ *are all* NP*-hard.*

6 The weight at least t version

In this section we give an fpt algorithm finding solutions of weight at least t.

Theorem 6.1. $\textsc{LinEq}_{\geq,t} \in \mathsf{FPT}$.

Proof. Let $Ax = b$ be the given equation system and let t be the given weight threshold. Using Gaussian elimination, we can decide whether $Ax = b$ is feasible and compute the dimension d of the solution space of $Ax = 0$ in polynomial time. If d exceeds $t \log m$ then there must be a solution of weight at least t, since there can be only

$$\sum_{i=0}^{t-1} \binom{m}{i} < 2^{t \log m}$$

solutions of weight less than t. Otherwise, we compute a linearly independent spanning set $\{v_1, v_2, \ldots, v_d\}$ of at most $t \log m$ solutions for $Ax = 0$, along with a particular solution u of $Ax = b$. Any solution to $Ax = b$ is of the form $u + \sum_{i=1}^{d} \alpha_i v_i$, where $\alpha_i \in \mathbb{F}_2$. If one of the solution vectors u and $u + v_i$ for $i = 1, \ldots, d$ has support at least t, it witnesses that the input is a positive instance. Otherwise all but $t^2 \log m$ coordinates are always zero in any solution vector. Hence, the number of relevant variables (that can take value 1 in any solution) is bounded by $t^2 \log m$. Discarding the other variables, we now have a system of linear equations with at most $t^2 \log m$ variables. We can now brute-force search for all solutions of Hamming weight at most $t - 1$. Note that the number S of such solutions is bounded by

$$\sum_{i=0}^{t-1} \binom{t^2 \log m}{i} < (t^2 \log m)^t.$$

Keeping in mind the easily checked fact that $(\log m)^{O(t)} = t^{O(t)} \operatorname{poly}(m)$, this search takes fpt time. Finally, comparing S with the dimension d, if $S = 2^d$, then all solutions have weight less than t, otherwise there must be a solution of weight at least t. □

The previous algorithm may not always construct a solution of weight at least t if it exists. For solving the search problem in fpt time, we can use self-reduction in a standard way. An alternative randomized algorithm for constructing a solution can be obtained by modifying the above procedure slightly. If the dimension of the solution space of $Ax = 0$ exceeds $2t \log m$ then we randomly sample a solution to obtain a solution of weight at least t with high probability. On the other hand, if the dimension d of the solution space is bounded by $2t \log m$ we proceed as in the above proof to compute a spanning set $\{v_1, v_2, \ldots, v_d\}$ of the solution space and keep only the $2t^2 \log m$ many relevant variables (if the solutions u and $u + v_i$ for $i = 1, \ldots, d$ have weight at most $t - 1$). Now, we can apply a simple branching algorithm, by branching on the $2t^2 \log m$ many variables (on each branch we set the corresponding variable to 1). This branching search tree will be of depth t and hence of size at most $t^{O(t)} \operatorname{poly}(m)$. The system of equations has a solution of weight at least t iff at some leaf of this depth-t tree the residual system of linear equations is feasible, which means that the partial weight t assignment we have can be extended to a solution.

References

[AYZ95] Noga Alon, Raphael Yuster and Uri Zwick, Color coding. *Journal of the ACM* 42(4), 844–856, 1995.

[BMT78] Elwyn R. Berlekamp, Robert J. McEliece and Henk C.A. van Tilborg, On the inherent intractability of certain coding problems. *IEEE Trans. Inform. Theory* 24, 384–386, 1978.

[BR95] Danilo Bruschi and F. Ravasio, Random parallel algorithms for finding exact branchings, perfect matchings, and cycles. *Algorithmica,* 13(4), 346–356, 1995.

[DF99] Rod G. Downey and Michael R. Fellows, *Parameterized Complexity.* Springer, 1999.

[DFV+99] Rod G. Downey, Michael R. Fellows, Alexander Vardy and Geoff Whittle, The parametrized complexity of some fundamental problems in coding theory. *SIAM Journal on Computing* 29(2): 545–570, 1999.

[DMS03] Ilya Dumer, Daniele Micciancio and Madhu Sudan, Hardness of approximating the minimum distance of a linear code. *IEEE Transactions on Information Theory*, 49(1), 22–37, 2003.

[EJT10] Michael Elberfeld, Andreas Jakoby, and Till Tantau, Logspace versions of the theorems of Bodlaender and Courcelle. *FOCS Conference,* 143–152, 2010.

[JLL76] Neil D. Jones, Y. Edmund Lien, and William T. Laaser, New problems complete for nondeterministic log space. *Mathematical Systems Theory,* 10(1), 1–17, 1976.

[Joh05] David S. Johnson, The NP-completeness column. *ACM Transactions on Algorithms,* 1(1), 160–176, 2005.

[NH81] Simeon C. Ntafos and S. Louis Hakimi, On the complexity of some coding problems. *IEEE Transactions on Information Theory*, 27(6): 794-796, 1981.

[MVV87] Ketan Mulmuley, Umesh Vazirani and Vijay Vazirani, Matching is as easy as matrix inversion. *Combinatorica* 7:105–113, 1987.

[PY82] Christos Papadimitriou and Mihalis Yannakakis, The complexity of restricted spanning tree problems. *Journal of the ACM* 29:285–309, 1982.

[Rei08] Omer Reingold, Undirected connectivity in log-space. *Journal of the ACM* 55(4), 2008.

[Var97a] Alexander Vardy, The intractability of computing the minimum distance of a code. *IEEE Trans. Inform. Theory* 43, 1757–1766, 1997.

[Var97b] Alexander Vardy, Algorithmic Complexity in Coding Theory and the Minimum Distance Problem. *Proc. 29th ACM Symposium on Theory of Computing,* 92–109, 1997.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de