International Journal of Foundations of Computer Science


# MOD-2 INDEPENDENCE AND DOMINATION IN GRAPHS

MAGNÚS M. HALLDÓRSSON *
*University of Iceland, Reykjavik, Iceland*
*mmh@hi.is*

and

JAN KRATOCHVÍL **
*Charles University, Prague, Czech Republic.*
*honza@kam.ms.mff.cuni.cz*

and

JAN ARNE TELLE
*University of Bergen, Bergen, Norway.*
*telle@ii.uib.no*

Received (received date)
Revised (revised date)
Communicated by Editor's name

ABSTRACT

We develop an $O(n^3)$ algorithm for deciding if an $n$-vertex digraph has a subset of vertices with the property that each vertex of the graph has an even number of arcs into the subset. This algorithm allows us to give a combinatorial interpretation of Gauss-Jordan and Gauss elimination on square boolean matrices. In addition to solving this independence-mod-2 (even) set existence problem we also give efficient algorithms for related domination-mod-2 (odd) set existence problems on digraphs. However, for each of the four combinations of these two properties we show that even though the existence problem on digraphs is tractable, the problems of deciding the existence of a set of size exactly $k$, larger than $k$, or smaller than $k$, for a given $k$, are all NP-complete for undirected graphs.

## 1. Introduction

A large class of well-studied independence and domination properties in graphs can be characterized by two sets of nonnegative integers $\sigma$ and $\rho$. A ($\sigma$, $\rho$)-set $S$ in a graph has the property that the number of neighbors every vertex $u \in S$ (or $u \notin S$) has in $S$, is an element of $\sigma$ (of $\rho$, respectively) [7]. In a recent paper [3] it

---

* Adjunct affiliation University of Bergen, Norway

** Research support in part by Czech research grants GAUK 194/1996 and 158/1999, and GAČR 201/1996/0194.

is shown that deciding if a given graph has a $(\{0\}, \rho)$-set, *i.e.* vertices in $S$ forming an independent set with further $\rho$-imposed domination constraints, is NP-complete whenever there is a non-negative integer $x \notin \rho$ with $x+1 \in \rho$, unless $\rho$ are exactly the positive numbers, and polynomial in all other cases. For the cases $0 \notin \rho$ the vertices in $S$ form a dominating set. In the present paper we consider cases of independence and domination modulo 2, where $\sigma$ and $\rho$ are either the set of all even numbers or the set of all odd numbers. We denote these sets EVEN and ODD, respectively, and consider both decision and optimization versions of the four cases of $(\sigma, \rho)$ equal to (EVEN, EVEN), (EVEN, ODD), (ODD, ODD), (ODD, EVEN).

In the next section, we develop an $O(n^3)$ algorithm to decide if a given graph $G$ has an (EVEN, EVEN)-set, *i.e.* a non-empty set $S$ such that each vertex of $G$ has an even number of neighbors in $S$. This disproves a 1994 conjecture stating that no non-trivial $(\sigma, \rho)$-set existence problems were solvable in polynomial time [2]. In section 3 we show that quite trivially this problem is equivalent to determining if the adjacency matrix of the given graph is singular, and that our algorithm is a combinatorial interpretation of Gauss-Jordan elimination on square boolean matrices. A recent paper giving combinatorial interpretations of various matrix algorithms left such a view of Gaussian elimination, for general matrices, as an open problem [5]. We provide a partial answer to this question by using the connection with (EVEN, EVEN)-sets to give a combinatorial interpretation of Gaussian elimination for square boolean matrices.

In section 4 we give polynomial algorithms also for the three remaining existence problems. In section 5 we consider the complexity of deciding if a given graph has a desired set of size at least $k$, at most $k$, or exactly $k$, for a given integer $k$. By reductions from NP-complete coding problems asking for codewords of given length, we show that these maximization, minimization and exact versions of all four problems are NP-complete.

## 2. Existence of independence mod-2 sets

We will be viewing an undirected graph as a directed graph with arcs $uv$ and $vu$ for each edge $\{u, v\}$. We first generalize the $(\sigma, \rho)$ problems to directed graphs with loops, and denote the set of out-neighbors of vertex $v$ by $v_{out} = \{u : vu \in E\}$ and its in-neighbors by $v_{in} = \{u : uv \in E\}$. If the graph $G$ is not clear from context we write $v_{out}(G), v_{in}(G)$.

**Definition 1.** A nonempty subset of vertices $S$ of a directed graph $G = (V, E)$ is a $(\sigma, \rho)$-set if $|v_{out} \cap S| \in \sigma$ for any $v \in S$ and $|v_{out} \cap S| \in \rho$ for any $v \in V \setminus S$.

Note that we could also have chosen to count in-neighbors, since in the graph with all arcs reversed this would define the exact same vertex subsets as $(\sigma, \rho)$-sets. This simple transformation implies that decision problems over $(\sigma, \rho)$-sets will have the same time complexity regardless of whether we count in-neighbors or out-neighbors.

Our algorithm for deciding if a graph $G$ has an (EVEN, EVEN)-set (which by Definition 1 must be non-empty) will consist of repeatedly applying a graph operation that will maintain the property of interest. This will give a series of graphs $G^{(0)}, G^{(1)}, ..., G^{(i)}$, starting with the input graph and ending with a graph for which it will be trivial to decide if it has an (EVEN, EVEN)-set. Let $\oplus$ be the symmetric difference operator, *i.e.* $A \oplus B = \{x \in A \cup B : x \notin A \cap B\}$. The main observation is that for any two vertices $u$ and $r$, (EVEN, EVEN)-sets are invariant under the operation:

$$u_{out} := u_{out} \oplus r_{out}$$

**Lemma 2.** *Let $G'$ be the graph $G$ altered by $u_{out}(G') := u_{out}(G) \oplus r_{out}(G)$ for two vertices $u, r$. Then $S$ is an (EVEN, EVEN)-set of $G$ if and only if $S$ is an (EVEN, EVEN)-set of $G'$.*

*Proof.* Outgoing neighbors for any vertex $x \neq u$ are identical in $G$ and $G'$. For the forward direction of the proof it therefore suffices to show that $|u_{out}(G') \cap S| = |u_{out}(G) \cap S| + |r_{out}(G) \cap S| - 2|u_{out}(G) \cap r_{out}(G) \cap S|$ is even. But since $S$ is an (EVEN, EVEN)-set of $G$, all 3 terms in the right-hand side of the above equality are even and thus so is their sum. Conversely, we have $|u_{out}(G) \cap S| = |u_{out}(G') \cap S| - |r_{out}(G) \cap S| + 2|u_{out}(G) \cap r_{out}(G) \cap S|$ also even for similar reasons.

Our algorithm will for each of the $n$ vertices in $G$ maintain an *in-flag* and an *out-flag*, initially all lowered. In the $i$th stage of the algorithm we choose a vertex $c$ with lowered in-flag that has at least one incoming neighbor $r$ with lowered out-flag. In addition to raising the in-flag of $c$ and the out-flag of $r$ the code for this stage consists of the loop:

for each $u \in c_{in} \setminus \{r\}$ do $u_{out} := u_{out} \oplus r_{out}$

In the resulting graph $G^{(i)}$ the vertex $c$ will have only the single incoming neighbor $r$. Once flags are raised they are never lowered, thus after $n$ successful stages each vertex would have exactly one incoming and one outgoing neighbor. Clearly, such a graph can have no (EVEN, EVEN)-set. However, if there is a vertex in $G^{(i)}$ with lowered in-flag which has no incoming neighbor with lowered out-flag, then an (EVEN, EVEN)-set exists and we halt. Before proving this fact we give the algorithm formally below. Sets $C$ and $R$ represent the subsets of vertices having raised in-flags and out-flags, respectively.

<u>$\exists$ (EVEN, EVEN)-SET ALGORITHM</u>
*input*: digraph $G = (V, E)$
$C := R := \emptyset$
$i := 0$
while $(i < n)$ and $(\nexists x \in V \setminus C : x_{in} \subseteq R)$ do
    { $i := i + 1$
      pick $c \in V \setminus C$ and set $C := C \cup \{c\}$
      pick $r \in c_{in} \setminus R$ and set $R := R \cup \{r\}$
      for each $u \in c_{in} \setminus \{r\}$ do $u_{out} := u_{out} \oplus r_{out}$ }
if $(i = n)$ then $\nexists$ (EVEN, EVEN)-set
else { let $x \in V \setminus C : x_{in} \subseteq R$
    $S := \{x\} \cup \{v \in C : v \in y_{out} \wedge y \in x_{in}\}$ is an (EVEN, EVEN)-set}

**Lemma 3.** *If $i < n$ upon completion of the algorithm then $S$ is an (EVEN, EVEN)-set of the current graph $G^{(i)}$.*

*Proof.* A vertex $v \notin R$ has no outgoing edges to $C$ so $|v_{out} \cap S| = 0$. Note that $|x_{in}| = |\{v \in C : v \in y_{out} \wedge y \in x_{in}\}|$ since $x_{in} \subseteq R$ and each vertex of $R$ has exactly one, distinct, outgoing neighbor in $C$. Each vertex $v \in x_{in}$ has therefore 2 outgoing neighbors in $S$, namely $x$ and $v_{out} \cap C$, while any vertex $w \in R$ with $w \notin x_{in}$ has no outgoing neighbors in $S$.

By applying Lemma 2 inductively it follows that the algorithm for existence of (EVEN, EVEN)-sets is correct. Its time complexity is $O(n^3)$ since in each of the at most $n$ stages the chosen vertex $c$ has at most $n$ incoming neighbors that each have their at most $n$ outgoing neighbors updated.

**Theorem 4.** *The algorithm decides, in time $O(n^3)$, if the input graph has an (EVEN, EVEN)-set or not.*

### 3. Gaussian elimination on boolean matrices

Consider what the existence of an (EVEN, EVEN)-set $S$ in a graph $G$ implies for the boolean adjacency matrix $A_G$ of $G$. Clearly, the columns corresponding to vertices in $S$ sum to the all-zero vector (over GF(2)). Conversely, any non-empty set of columns summing to the all-zero vector is linearly dependent and the corresponding vertices form an (EVEN, EVEN)-set. Thus the matrix $A_G$ has less than full rank, *i.e.* is singular, *i.e.* has determinant zero, if and only if $G$ has an (EVEN, EVEN)-set.

**Theorem 5.** *A square boolean matrix is singular if and only if its associated directed graph has an (EVEN, EVEN)-set.*

Note that the algorithm given for the existence of (EVEN, EVEN)-sets works for any digraph, even one with self-loops. In fact, viewing it as a matrix algorithm over GF(2) it is equivalent to Gauss-Jordan elimination, as follows: In the main loop of the algorithm a new column ($c \notin C$) is processed, a non-zero pivot (entry $rc$) is chosen from the remaining pivot rows ($r \notin R$), and row operations are performed to make all other entries in column $c$ equal to zero. If the algorithm completes all $n$ stages then we are left with a permutation matrix, and otherwise we find a set of columns that are linearly dependent.

Even if it has the same asymptotic time complexity, Gaussian elimination is usually preferred over Gauss-Jordan in practice, as the constant term is smaller. Let us consider Gaussian elimination as an algorithm for determining existence of (EVEN, EVEN)-sets. The changes from the previous algorithm are in: (i) labelling of chosen vertices for ease, (ii) all in-neighbors of $c^i$ in $R$ (previously only $r^i$) are left untouched in the main loop, and (iii) definition of (EVEN, EVEN)-set $S$.

```
GAUSS ∃ (EVEN, EVEN)-SET ALGORITHM
input: digraph G = (V, E)
C := R := ∅
i := 0
while (i < n) and (∄x ∈ V \ C : x_in ⊆ R) do
      { i := i + 1
        pick c^i ∈ V \ C and set C := C ∪ {c^i}
        pick r^i ∈ c^i_in \ R and set R := R ∪ {r^i}
        for each u ∈ c^i_in \ R do u_out := u_out ⊕ r^i_out }
if (i = n) then ∄ (EVEN, EVEN)-set
else { S := {x}
      for k := i downto 1   if |r^k_out ∩ S| is odd then S := S ∪ {c^k}
      S is an (EVEN, EVEN)-set}
```

**Lemma 6.** *The GAUSS ∃ (EVEN, EVEN)-set algorithm is correct.*

*Proof.* Assume the algorithm completes with $i < n$. Then each $v \in V \setminus R$ has zero out-neighbors to $x$ by the halting condition of the main loop, and zero outgoing neighbors to $\{c^1, c^2, ..., c^i\}$ as the only arcs to $c^k$ left after iteration $k$ of the main loop are from $\{r^1, ..., r^{k-1}\} \subseteq R$. Since $r^k$ has an arc to $c^k$, but none to $\{c^1, ..., c^{k-1}\}$ the reverse ordering of the final loop in the definition of $S$ implies that each $r^k \in R$ will have an even number of out-neighbors to $S$. Hence, $S$ is an (EVEN, EVEN)-set.

On the other hand, if $i = n$, we show by reverse induction on $k$ that $c^k$ cannot belong to an (EVEN, EVEN)-set $S$. Assume $c^n, ..., c^{k+1} \notin S$, for $k \leq n$. We cannot have $c^k \in S$ as the only out-neighbor of $r^k$ among $\{c^1, ..., c^k\}$ is $c^k$, so that $r^k$ would then have had exactly one out-neighbor in $S$.

We thus have a combinatorial interpretation of Gaussian elimination for square boolean matrices.

## 4. Existence of domination mod-2 sets

In this section we prove the following result.

**Theorem 7.** *The existence of $(\sigma, \rho)$-sets of type (ODD, ODD), (ODD, EVEN) and (EVEN, ODD) in directed graphs can be decided in polynomial time.*

*Proof.* Let $G$ have $n$ vertices and let $A_G$ be its adjacency matrix. We denote by $\mathbf{1}$ and $\mathbf{0}$ the all-one and all-zero vectors of dimension $n$ and by $\mathbf{I}$ the $n \times n$ identity matrix. We have observed that $G$ has an (EVEN, EVEN)-set if and only if there is a non-zero vector $\mathbf{x}$ such that $A_G\mathbf{x} = \mathbf{0}$. Similarly, a vector $\mathbf{x}$ is the characteristic vector of an (ODD, ODD)-set if and only if $A_G\mathbf{x} = \mathbf{1}$. Similarly, for an (ODD, EVEN)-set we have $(A_G + \mathbf{I})\mathbf{x} = \mathbf{0}$ and for an (EVEN, ODD)-set we have $(A_G + \mathbf{I})\mathbf{x} = \mathbf{1}$. Thus, deciding the existence of these kinds of sets can be done in polynomial time by solving linear equations.

## 5. Existence of sets of a given size

In this section we show that deciding the existence of independence and domination mod-2 sets of a given size $k$, whether exactly $k$, at least $k$ or at most $k$, is

NP-complete even for undirected graphs. Note that the properties studied are not hereditary, so that a graph may for example have an (EVEN, EVEN)-set of size $k$, but none of size larger or smaller than $k$. Our reductions will be from NP-complete problems in coding theory, that for our purposes can be described as follows:

**Codeword of given weight:** Given a binary $r \times c$ matrix $H$ and an integer $w$, is there a vector $\mathbf{x}$ with $w$ ones s.t. $H\mathbf{x} = \mathbf{0}$?

This problem on binary linear codes was shown NP-complete in [1]. The problem **Codeword of maximal weight**, asking for a vector of weight **at least** $w$ is also NP-complete for binary codes [6]. Finally, the problem **Codeword of minimal weight** for binary linear codes, asking for a non-zero vector of weight **at most** $w$ was conjectured NP-complete in [1], and finally proven to be so in a recent paper [9]. These problems are equivalent to asking if the orthogonal complement of the linear space generated by the columns of $H$ contains a non-zero vector of weight $w$, at least $w$, or at most $w$ (in other words, if there are exactly $w$, at least $w$, or nonempty set of at most $w$ columns of $H$ that sum up to the all-zero vector). They are thus very close to (EVEN, EVEN)-set problems. However, inputs to the (EVEN, EVEN)-set problems are square matrices, and for undirected graphs also symmetric matrices with zeros on the diagonal. We first show NP-completeness for the maximum, minimum and exact versions of the (EVEN, EVEN)-set undirected graph problems, and then use these results to give reductions for the other three properties.

**Theorem 8.** *The problems Codeword of maximal weight and Codeword of minimal weight remain NP-complete for symmetric matrices with all-zero diagonals.*

*Proof.* The problems are clearly in NP. We first resolve the maximal weight version by giving a polynomial-time reduction from the NP-complete problem Codeword of maximal weight. Given a boolean $r \times c$ matrix $H$ and an integer $w$ we construct a symmetric matrix with all-zero diagonals $G$ such that $G$ has a codeword of size at least $k = 2r + w$ iff $H$ has a codeword of weight at least $w$. $G$ will have the following form:

$$\begin{pmatrix} \mathbf{0} & \mathbf{0} & H \\ \mathbf{0} & \mathbf{0} & H \\ H^t & H^t & \mathbf{0} \end{pmatrix}$$

where $H^t$ is the transpose of $H$, the lower-right $\mathbf{0}$ is the $c \times c$ all-zero matrix, and the other $\mathbf{0}$s are $r \times r$ all-zero matrices. This is a square $(2r + c) \times (2r + c)$ symmetric matrix with zeros on the diagonal. Since the leftmost $2r$ columns sum to the all-zero vector, we conclude that this matrix has a set of at least $2r + w$ columns summing to the all-zero vector iff $H$ has a set of at least $w$ columns summing to the all-zero vector.

We next resolve the minimal weight version by reduction from Codeword of minimal weight. Given a boolean $r \times c$ matrix $H$ and an integer $w$, we construct a symmetric all-zero diagonal matrix $G$ such that $G$ has a codeword of size at most $k = w$ iff $H$ has a codeword of weight at most $w$.

We may assume wlog that $r$ is even, since we could add an all-zero row to $H$ otherwise. $G$ will have $(w+1) \times (w+1)$ blocks $W_{ij}, i, j = 1, 2, \ldots, w+1$ where $W_{1,w+1} = H$ and $W_{w+1,1} = H^t$. The blocks $W_{1,w} = W_{i,w+1-i} = W_{i,w+2-i}$ for $i = 2, \ldots, w$ will contain the symmetric permutation matrix $P$ of size $r$ by $r$ with the unique 1-entry in each row and column in position $(r+1-i, i), i = 1..r$ (since $r$ is even $P$ has zeroes on the diagonal.) All other blocks are all-zero matrices of appropriate size. For the case $w = 3$ the matrix $G$ thus becomes:

$$\begin{pmatrix} \mathbf{0} & \mathbf{0} & P & H \\ \mathbf{0} & P & P & \mathbf{0} \\ P & P & \mathbf{0} & \mathbf{0} \\ H^t & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix}$$

This is a $3r + c$ by $3r + c$ ($wr + c$ by $wr + c$) symmetric matrix with zeros on the diagonal consisting of 4 by 4 ($w + 1$ by $w + 1$) blocks. The placement of the permutation matrices ensures that choosing a column from any but the rightmost column of blocks will force a choice of a column from all the columns of blocks, *i.e.* forcing a choice of at least $w + 1 = 4$ columns. Hence the matrix has a set of at most $w = 3$ columns summing to the all-zero vector iff all columns come from the rightmost block, *i.e.* from $H$. A similar argument applies to the general case.

**Corollary 9.** *Given an undirected graph $G$ and an integer $k$, deciding if $G$ has a non-empty (EVEN, EVEN)-set of size at least $k$, at most $k$, or exactly $k$ is NP-complete.*

The corollary is immediate since the minimum and maximum versions are equivalent to the analogous codeword problems and the exact version follows by a Cook reduction from either of the other two. We turn to the other problems.

**Theorem 10.** *The maximum, minimum and exact versions of the (ODD, ODD), (ODD, EVEN) and (EVEN, ODD) problems are all NP-complete, even for undirected graphs.*

*Proof.* Given a graph $G$ as input to a known NP-complete problem (as specified below), Figure 1 shows the constructed graphs $G_1, ..., G_5$ for five separate NP-completeness reductions for maximization and minimization versions, and NP-completeness of the exact versions will follow from this. For each graph in Figure 1 is shown two vertices of the graph $G$, and the corresponding subgraph that is attached to every vertex of $G$ to form $G_i$. In all reductions below, $G$ is a graph with vertices $\{v_1, ..., v_n\}$.

The first reduction is from min and max versions of (EVEN,EVEN) to min and max versions of (ODD,EVEN). Given a graph $G$ and an integer $k$ subject to min or max variant of (EVEN,EVEN) problem, we construct $G_1$ and ask for an (ODD,EVEN)-set in $G_1$ of size at most or at least $2k$, respectively. Let $G_1$ be the graph consisting of a copy of $G$ plus added leaf vertices $\{x_1, ..., x_n\}$ with $x_i$ adjacent to $v_i$.

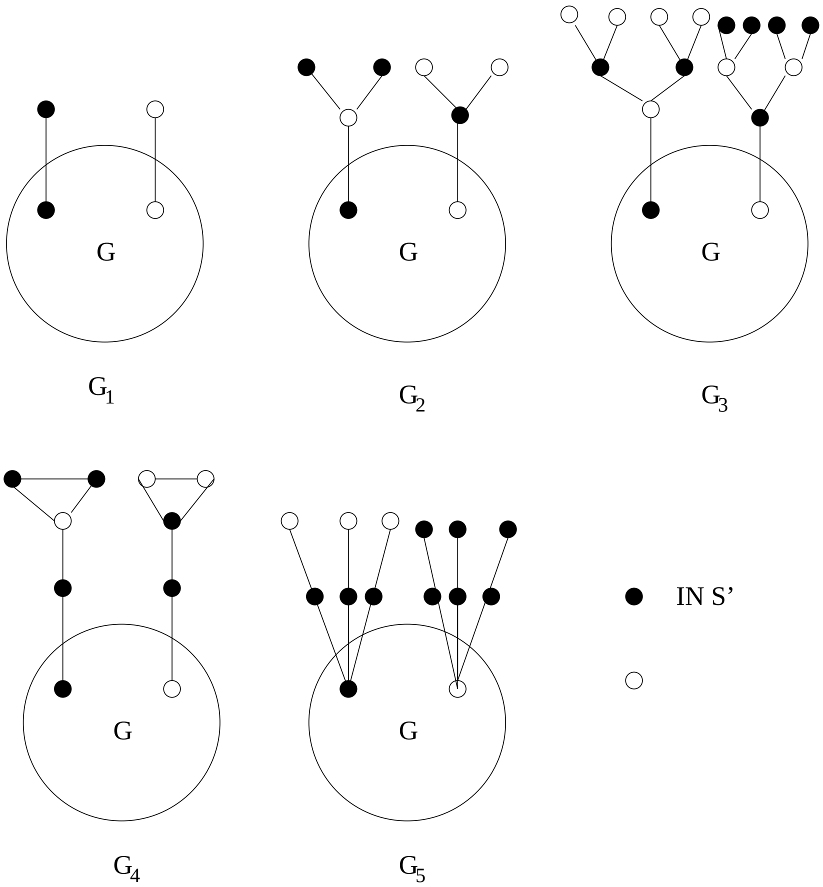


**Fig. 1.** The constructed graphs $G_1, ..., G_5$ for five separate reductions. For each graph is shown two vertices of the given graph $G$, and the corresponding subgraph that is attached to every vertex of $G$ to form $G_i$. The two unique possibilities for membership in a $(\sigma, \rho)$-set $S'$ of $G_i$ are shown, with black vertices belonging to $S'$ and white not.

**Claim 1** *$G$ has an (EVEN, EVEN)-set of size $k$ if and only if $G_1$ has an (ODD, EVEN)-set of size $2k$.*

*Proof.* Let $S$ be an (EVEN, EVEN)-set of $G$ of size $k$. We show that $S' = S \cup \{x_i : v_i \in S\}$ is an (ODD, EVEN)-set of $G_1$. The new leaf vertices have either 1 or 0 $S'$-neighbors depending on whether they belong to $S'$ or not, as desired. A vertex $v \in V(G)$ with $v \in S$ has $|v_{out}(G_1) \cap S'| = |v_{out}(G) \cap S| + 1$, an odd number, while $v \in V(G)$ with $v \notin S$ has the same $S'$-neighbors as $S$-neighbors, an even number. Thus, $S'$ forms an (ODD, EVEN)-set of size $2k$. Conversely, an (ODD, EVEN)-set $S'$ of $G_1$ contains a new leaf vertex $x_i$ if and only if it contain its neighbor $v_i$, so that $S' \cap V(G)$ forms an (EVEN, EVEN)-set of $G$ of appropriate size.

We now give a reduction from the max (EVEN,EVEN) problem to the max (EVEN,ODD) problem. Given a graph $G$ and an integer $k$ subject to the max

(EVEN,EVEN) problem, we construct $G_2$ and ask for an (EVEN, ODD)-set in $G_2$ of size at least $2k+n$. Let $G_2$ be the graph consisting of a copy of $G$ plus added vertices $\{x_1, y_1, z_1, ..., x_n, y_n, z_n\}$ with $x_i$ adjacent to $v_i, y_i, z_i$. $G_2$ is thus $G$ with a two-level complete binary tree attached to each vertex of $G$.

**Claim 2** *$G$ has an (EVEN, EVEN)-set of size $k$ if and only if $G_2$ has an (EVEN, ODD)-set of size $2k+n$.*

*Proof.* Let $S$ be an (EVEN, EVEN)-set of $G$ of size $k$. We show that $S' = S \cup \{x_i : v_i \notin S\} \cup \{y_i, z_i : v_i \in S\}$ is an (EVEN, ODD)-set of $G_2$. The new vertices have 0 $S'$-neighbors if they belong to $S'$ and either 3 or 1 if they do not, as desired. A vertex $v \in V(G)$ with $v \in S$ has no new $S'$-neighbors, while $v \in V(G)$ with $v \notin S$ has gained the $S'$-neighbor $x_i$. Thus $S'$ forms an (EVEN, ODD)-set of size $2k+n$. Conversely, in any (EVEN, ODD)-set $S'$ of $G_2$ either both $y_i$ and $z_i$ are in $S'$ or none of them are. If they both are then $x_i \notin S'$ but $v_i \in S'$ while if none of them are then $x_i \in S'$ but $v_i \notin S'$. We conclude that $S' \cap V(G)$ forms an (EVEN, EVEN)-set of $G$ of appropriate size.

Since $G_2$ always has an (EVEN,ODD)-set of size $n$, consisting of $x_i$ for each $i$, it cannot be used in a reduction for the min (EVEN, ODD) problem. Instead, given a graph $G$ and an integer $k$ subject to the max (EVEN,EVEN) problem, we construct a new graph $G_3$ and ask for an (EVEN, ODD)-set in $G_3$ of size at most $5n-2k$. $G_3$ is constructed by attaching a three-level complete binary tree to each vertex of $G$. Such a tree thus contains one vertex at level 1, two at level 2, and four at level 3.

**Claim 3** *$G$ has an (EVEN, EVEN)-set of size $k$ if and only if $G_3$ has an (EVEN, ODD)-set of size $5n-2k$.*

*Proof.* For any (EVEN,EVEN)-set $S$ of $G$ we have an (EVEN,ODD)-set $S'$ of $G_3$ consisting of vertices in $S$ and vertices at alternate levels of each attached tree, so that a tree attached to a vertex in $S$ (respectively, not in $S$) has both vertices at level 2 in $S'$ (respectively, all five vertices at levels 1 and 3 in $S'$). If $S$ has size $k$, $S'$ has size $3k + 5(n-k) = 5n-2k$. For the other direction, note that any (EVEN,ODD)-set of $G_3$ must, for each attached tree, contain all the vertices at alternate levels, and it contains the vertex at level 1 if and only if its neighbor from $G$ is not in the (EVEN, ODD)-set.

We now give a reduction from the max (EVEN,EVEN) problem to the max (ODD,ODD) problem. Given a graph $G$ and an integer $k$ subject to the max (EVEN,EVEN) problem, we construct $G_4$ and ask for an (ODD, ODD)-set in $G_4$ of size at least $2k+2n$. Let $G_4$ be the graph consisting of a copy of $G$ plus added vertices $\{x_1, y_1, z_1, w_1, ..., x_n, y_n, z_n\}$ with a triangle on $y_i, z_i, w_i$ and with $x_i$ adjacent to $v_i$ and to $y_i$.

**Claim 4** *$G$ has an (EVEN, EVEN)-set of size $k$ if and only if $G_4$ has an (ODD, ODD)-set of size $2k+2n$.*

*Proof.* It will suffice to show that any (ODD,ODD)-set $S'$ of $G_4$ must for each $1 \le i \le n$ contain exactly the vertices $\{v_i, x_i, z_i, w_i\}$ or $\{x_i, y_i\}$. This holds since out of the triangle-forming vertices $y_i, z_i, w_i$ either $y_i$ is the only member of $S'$, or

$z_i, w_i$ are members of $S'$ but $y_i$ is not. In the former case, $x_i$ is also a member of $S'$ but $v_i$ is not, while in the latter case both $x_i$ and $v_i$ are in $S'$. We conclude that $S' \cap V(G)$ forms an (EVEN, EVEN)-set of $G$ of appropriate size.

Since $G_4$ always has an (ODD,ODD)-set of size $2n$, consisting of $x_i$ and $y_i$ for each $i$, it cannot be used in a reduction for the min (ODD, ODD) problem. Instead, given a graph $G$ and an integer $k$ subject to the max (EVEN,EVEN) problem, we construct a new graph $G_5$ and ask for an (ODD, ODD)-set in $G_5$ of size at most $6n-2k$. $G_5$ is constructed from a copy of $G$ by adding $6n$ vertices $\{x_1^j, y_1^j, ..., x_n^j, y_n^j\}$ for $j = 1, 2, 3$, with $x_i^j$ adjacent to both $v_i$ and $y_i^j$ ($G_5$ can be constructed by first adding three leaves to each vertex and then subdividing each new edge.)

**Claim 5** *G has an (EVEN, EVEN)-set of size k if and only if $G_5$ has an (ODD, ODD)-set of size $6n - 2k$.*

*Proof.* Let $S$ be an (EVEN, EVEN)-set of $G$ of size $k$. We show that $S' = S \cup \{x_i^j : 1 \leq j \leq 3, 1 \leq i \leq n\} \cup \{y_i^j : v_i \notin S\}$ is an (ODD, ODD)-set of $G_5$. The new vertices all have a single $S'$-neighbor as desired, while a vertex $v_i \in V(G)$ gains the 3 extra $S'$-neighbors $x_i^1, x_i^2, x_i^3$, so that $S'$ forms an (ODD, ODD)-set of $G_5$ of size $3n + 3(n-k) + k = 6n - 2k$. Conversely, any (ODD, ODD)-set $S'$ of $G_5$ must contain $x_i^j, 1 \leq j \leq 3, 1 \leq i \leq n$ and it contains $y_i^j$ iff it does not contain $v_i$. Hence, $S' \cap V(G)$ forms an (EVEN, EVEN)-set of $G$ of appropriate size.

This concludes the proof of the Theorem.

## 6. Conclusion

We have resolved the complexity of $(\sigma, \rho)$-set existence, maximization, minimization and exact size problems for the cases where $\sigma, \rho \in \{EVEN, ODD\}$. The only other cases of polynomial-time solvable $(\sigma, \rho)$-set existence problems we know are either the trivial cases, for example $\sigma = \{0\}, \rho = \{1, 2, ...\}$ where the answer is always positive since every graph has an independent dominating set, those solvable by a simple greedy algorithm, see [8], or by exhaustive search, see [3]. We believe these are the only easy cases.

**Conjecture 1** *The only $(\sigma, \rho)$-set existence problems solvable in polynomial time, apart from the trivial cases and those resolved by exhaustive search or a simple greedy algorithm, are when $\sigma, \rho \in \{EVEN, ODD\}$.*

To decide if a graph had an (EVEN,EVEN)-set we essentially did Gaussian elimination on its boolean adjacency matrix. The more general graph property resolved by Gaussian elimination on square matrices over the finite field $Z_p$ for a prime $p$ is: Given an edge-weighted digraph, can we assign vertex weights (not all zero) in such a way that after multiplying each edge weight by the weight of its sink vertex, the weights of edges leaving each vertex sum to zero mod $p$?

## References


1. E. Berlekamp, R.J. McEliece and H.C.A. van Tilborg, On the inherent intractability of certain coding problems, *IEEE Trans. Inform. Theory. Vol.29, No.3, 1978, 384-386.*
2. M. Halldórsson, unpublished.

3. M. Halldórsson, J. Kratochvíl and J.A. Telle, Independent sets with domination constraints, *Proceedings ICALP'98 - 25th International Colloquium on Automata, Languages and Programming, Aalborg, Denmark, July 1998, LNCS vol. 1443, 176-187*
4. M. Mahajan and V. Vinay, Determinant: combinatorics, algorithms, complexity, *Chicago Journal of Theoretical Computer Science, 1997:5, 1997.* Preliminary version SODA'97.
5. M. Mahajan and V. Vinay, Determinant: Old Algorithms, New Insights, in *Proceedings SWAT'98 - 6th Scandinavian Workshop on Algorithm Theory, Stockholm, Sweden, July 1998, LNCS Vol. 1432, 276-287.*
6. S.C. Ntafos and S.L. Hakimi, On the complexity of some coding problems, *IEEE Trans. Inform. Theory, Vol. 27, 1981, 794-796.*
7. J.A. Telle, Characterization of domination-type parameters in graphs, *Proceedings of 24th Southeastern International Conference on Combinatorics, Graph Theory and Computing -Congressus Numerantium* Vol.94 1993, 9-16.
8. J.A. Telle, Complexity of domination-type problems in graphs, *Nordic Journal of Computing 1(1994), 157-171.*
9. A. Vardy, The intractability of computing the minimum distance of a code, *IEEE Trans. Inform. Theory. Vol.43 No. 6, 1997, 1757-1766.* Preliminary version STOC'97.