COUNTING LINEAR EXTENSIONS OF RESTRICTED POSETS

SAMUEL DITTMER* AND IGOR PAK*

Abstract. The classical 1991 result by Brightwell and Winkler [BW91] states that the number of linear extensions of a poset is #P-complete. We extend this result to posets with certain restrictions. First, we prove that the number of linear extension for *posets of height two* is #P-complete. Furthermore, we prove that this holds for *incidence posets of graphs*. Finally, we prove that the number of linear extensions for *posets of dimension two* is #P-complete.

1. Introduction

Counting *linear extensions* (#LE) of a finite poset is a fundamental problem in both Combinatorics and Computer Science, with connections and applications ranging from Statistics to Optimization, to Social Choice Theory. It is primarily motivated by the following basic question: given a partial information of preferences between various objects, what are the chances of other comparisons?

In 1991, Brightwell and Winkler showed that #LE is #P-complete [BW91], but for various restricted classes of posets the problem remains unresolved. Notably, they conjectured that the following problem is #P-complete:

> #H2LE (*Number of linear extensions of height-2 posets*)
> **Input:** A partially ordered set P of height 2.
> **Output:** The number $\boldsymbol{e}(P)$ of linear extensions.

Here *height two* means that P has two levels, i.e. no chains of length 3. This problem has been open for 27 years, most recently reiterated in [Hub14, LS17]. Its solution is the first result in this paper.

Theorem 1.1. #H2LE *is* #P*-complete.*

Our second result is an extension of Theorem 1.1. It was proposed recently by Lee and Skipper in [LS17], motivated by non-linear combinatorial optimization.

> #IPLE (*Number of linear extensions of incidence posets*)
> **Input:** A graph $G = (V, E)$.
> **Output:** The number $\boldsymbol{e}(I_G)$ of linear extensions of the incidence poset I_G.

Here the incidence poset I_G is defined as a height 2 posets with vertices V on one level, edges E on another level, and the inequalities defined by adjacencies in G.

Theorem 1.2. #IPLE *is* #P*-complete.*

*Department of Mathematics, UCLA, Los Angeles, CA, 90095.
Email: {samuel.dittmer,pak}@math.ucla.edu.
February 15, 2018.

Theorem 1.2 implies Theorem 1.1, of course. Formally, the proofs of both results are independent, but use the same technical ideas of using number theory to to obtain targeted reductions modulo primes. However, since the proof Theorem 1.1 is both technically and conceptually simpler, we chose to include both proofs.

Our main and the most difficult result is the solution of the following natural problem posed in 1988 by Möhring [Möh89, p. 163], and then again in 1997 by Felsner and Wernisch [FW97] motivated by different applications.

> #D2LE (*Number of linear extensions of dimension-2 posets*)
> **Input:** A partially ordered set P of dimension two.
> **Output:** The number $\boldsymbol{e}(P)$ of linear extensions of P.

Here the poset P is said to have *dimension two* if it can be represented by a finite set of points $\{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \mathbb{R}^2$, with the inequalities $(x_i, y_i) \preccurlyeq (x_j, y_j)$ if $x_i \le x_j$ and $y_i \le y_j$, $i \neq j$. Equivalently, poset P has dimension two if and only if its comparability graph $\Gamma(P)$ has complement $\overline{\Gamma(P)} \simeq \Gamma(P^*)$, for a *dual poset* P^* (see e.g. [Tro92]).

Theorem 1.3. #D2LE *is* #P*-complete.*

As a motivation, Felsner and Wernisch [FW97], show that #D2LE is equivalent to the following problem on the number of possible bubble sorted permutations τ from a given $\sigma \in S_n$ (see also [BjW91, Reu96]).

> #Bruhat (*Size of principal ideal in the weak Bruhat order*)
> **Input:** A permutation $\sigma \in S_n$.
> **Output:** The number $\boldsymbol{e}(\sigma)$ of permutations $\tau \in S_n$ with $\tau \le \sigma$.

Here we write $\tau \le \sigma$ if τ can be obtained from σ by a *bubble sorting*: repeated application of adjacent transpositions which the minimal possible number of inversions:

$$\sigma \,=\, \tau \cdot (i_1, i_1+1) \cdots (i_\ell, i_\ell+1), \quad \text{where} \quad \operatorname{inv}(\sigma) \,=\, \operatorname{inv}(\tau) + \ell.$$

The *weak Bruhat order* B_n is defined to be $(S_n, \le)$. In #Bruhat, we consider the principal ideal $P_\sigma = B_n \cap \{\omega \le \sigma\}$, so in the notation above $\boldsymbol{e}(\sigma) = \boldsymbol{e}(P_\sigma)$. Note that #Bruhat is in #P.

We include a quick proof of the reduction of #D2LE to #Bruhat in §3.1, both for completeness and to introduce the framework for the proof of the main result.

Theorem 1.4. #Bruhat *is* #P*-complete.*

The proof of Theorem 1.4 is presented in two stages. First, we will describe a combinatorial problem #RigidCircuit. In Lemma 3.3 we give a parsimonious reduction from #3SAT, which is #P-complete, to #RigidCircuits. Then, in Lemma 3.5, we use a more complicated set of reductions from #RigidCircuits to #Bruhat to show that #Bruhat is #P-complete.

Let us emphasize that the proof of Lemma 3.5 is *computer assisted*, i.e. it has gates found by computer, but which in principle can be checked directly. See §8.2 for a detailed discussion of computational aspects of the proof.

Remark 1.5. The proof in [BW91] uses a modulo p argument and the Chinese Remainder Theorem, which we also employ for all results (cf. 8.3). In fact, this approach is one of the few applicable for these problems, since the existence of FPRAS (see below) strongly suggests the impossibility of a parsimonious reduction of #3SAT and its relatives.

Historical review. The notion of #P-*completeness* was introduced by Valiant [Val79] a a way to characterize the class of computationally hard counting problems; see [MM11, Pap94] for a modern treatment. Brightwell and Winkler [BW91] proved #P-completeness of #LE in 1991, resolving an open problem from 1984 (cf. 8.1). They applied the result to show that computing the volume of convex polytopes in $\mathbb{R}^n$ is #P-hard. This connection was first established by Khachiyan [Kha93] based on the work of Stanley [Sta86]. It was later used to improve sorting under partial information, see [KL91].

The #LE problem is somewhat related to the problem counting order ideals in a poset, known to be #P-complete [PB83]. In contrast with the latter problem, #LE has FPRAS which allows $(1+\varepsilon)$-approximation of $\boldsymbol{e}(P)$. This was proved by Karzanov and Khachiyan [KK91] and independently by Matthews [Mat91]. See also [BD99, BGHP10, FW97, Hub06] for improvement upon the Karzanov–Khachiyan Markov chain and mixing time bounds.

Moreover, there are several classes of posets for which the counting is known to be polynomial: the dimension-2 posets given by Young diagrams of skew shape (see e.g. [MPP18, Sta97]), the *series-parallel posets* (also dimension 2, see [Möh89, §2.4]), a larger class of posets with *bounded decomposition diameter* [Möh89, §4.2], *sparse posets* [EGKO16, KHNK16], posets whose covering graphs have disjoint cycles (see [Atk89]), and *N-free posets* with bounded width and spread [FM14].

The *height-2 posets* is an important and well studied class of posets. Brightwell and Winkler write: "We strongly suspect that Linear Extension Count for posets of height 2 is still #P-complete, but it seems that an entirely different construction is required to prove this" [BW91]. They got this half-right – note that our construction builds on top of their result.

In fact, the linear extensions of height-2 posets do seem to have a much easier structure than the general posets. For example, Trotter, Gehrlein and Fishburn [TGF92] prove the famous 1/3–2/3 *conjecture* for this class, a problem that remains open in full generality. Similarly, in a recent paper [CRS09], Caracciolo, Rinaldi and Sportiello, study a new Markov chain on linear extensions of height-2 posets, which they call *corrugated surfaces*. They are motivated by the *Bead Model* in Statistical Mechanics and *standard Young tableau* sampling. They claim, based on computer experiments, a nearly linear mixing time for this Markov chain. Recently, Huber [Hub14] noticed the connection and proved the nearly linear mixing time for a different Markov chain in this case.

Incidence posets are not as classical as height-2 posets, but have also been studied quite intensely. We refer to recent papers [LS17, TW14] for an overview of the area and further references.

The study of posets of a given dimension is an important area, and the dimension 2 is both the first interesting dimension and special due to the duality property. See monograph [Tro92] for a comprehensive treatment. Posets of dimension 2 have a sufficiently rigid combinatorial structure to make various computational problems tractable. For example, the decision problem whether a poset has dimension 2 is in P(see e.g. [Tro95], as is the above mentioned problem of counting ideals of dimension-2 posets, see [Möh89, p. 163]. Another surprising property of dimension-2 posets is an asymptotically sharp lower and upper bounds on the product $\boldsymbol{e}(P)\boldsymbol{e}(P^*)$, where P^* is the *dual poset* defined above, see [BBS99].

The *weak Bruhat order* is a fundamental object in Algebraic Combinatorics, well studied in much greater generality, see e.g. [BjB05]. As we mentioned above, the connection between #D2LE and #Bruhat has been rediscovered a number of times in varying degree of generality, see [BjW91, FW97, Reu96].

Finally, *computer assisted proofs* are relatively rare in computational complexity. Let us mention [K+17, MR08] for two recent NP-completeness results with substantial computational component, and [BDGJ99, Zwi02] for two older computer assisted complexity results. To the best of our knowledge this paper is the only computer assisted proof of #P-completeness, and the only one which uses algebraic systems to encode logical gates. We refer to [Mac01] for a historical and sociological overview of the method and further references.

Paper structure. We start with a highly technical proof of theorems 1.4 and 1.3. In sections 3 and 4 we present the construction, in Section 5 we gives proofs of technical lemmas, and in the appendix list systems of algebraic equations defining parameters of the logical gates. In sections 6 and 7 we give complete proofs of theorems 1.1 and 1.2, respectively. Let us emphasize that the proof of Theorem 1.1 is completely independent from the rest of the paper and is streamlined as much as possible to be accessible to a larger audience. We conclude with final remarks and open problems in Section 8.

2. Basic definitions and notation

2.1. **Posets.** We assume the reader is familiar with basic definitions on posets, see e.g. [Tro95] and [Sta97, Ch. 3]. We describe a *linear extension* of a poset $\mathcal{P} = (X, <)$ on a set X with n elements as an *assignment* of the values $\{1, 2, \ldots, n\}$ to X, or as a *labeling* of X by the values $\{1, 2, \ldots, n\}$.

Let $\ell : X \to \{1, 2, \ldots, n\}$ be a linear extension of $\mathcal{P}$, and let X be given a default ordering, say $X = \{x_1, \ldots, x_n\}$. Then the function $i \mapsto \ell(x_i)$ is a permutation in S_n. We call this the permutation *induced* by the linear extension.

2.2. **Permutations.** For the technical constructions in Section 4 we express all permutations in one-line notation, in other words as a sequence where the integers from 1 to n occur exactly once. For several of these constructions, we wish to generalize permutations by either omitting or repeating numbers. We can treat an arbitrary sequence of n integers as a permutation in S_n by relabeling the elements from 1 to n, from smallest to largest, and, when a number is repeated, from left to right. For example, we would relabel the sequence

$$(7, 7, 5, 3, 3, 5)$$

by replacing the two 3's with a 1 and a 2, the two 5's with a 3 and a 4, and the two 7's with a 5 and 6, giving the permutation

$$(5, 6, 3, 1, 2, 4).$$

We will describe this relabeling explicitly where it helps to clarify the presentation, and talk about *shifting* elements up or down.

We use the term *block* exclusively to refer to a sequence of consecutive integers in consecutive position, and write it by replacing the sequence with an integer encased in a box: $\boxed{3}$.

2.3. **Other notation.** We write $\mathbb{N} = \{0, 1, 2, \ldots\}$ for the set of nonnegative integers, and $\mathbb{F}_q$ to denote the finite field with q elements. Let $[n] = \{1, 2, \ldots, n\}$ and $\binom{[n]}{k}$ to denote k-subsets of $[n]$. To make our notation more readable, when writing vectors in $\mathbb{F}_q^d$, we omit parentheses and commas, so that $(0, 1)$ becomes 01.

We refer to [MM11, Pap94] for notation, basic definitions and results in computational complexity. We use ϕ for logical gates. We introduce a new notation $\phi \rtimes (v_1, v_2)$ to be a result of a certain operation corresponding to (v_1, v_2) applied to ϕ, see §4.2.

3. The setup for #D2LE and #Bruhat

3.1. **Linear extensions and the Bruhat order.** We begin with a known result that #D2LE is equivalent to #Bruhat.

Lemma 3.1 ([FW97])**.** *For every $\sigma \in S_n$, there exists a poset P_σ of dimension two with n elements such that $\boldsymbol{e}(P_\sigma) = \boldsymbol{e}(\sigma)$. Conversely, for every poset P of dimension two with n elements, there exists $\sigma \in S_n$ such that $\boldsymbol{e}(P) = \boldsymbol{e}(\sigma)$.*

Proof. Given a permutation $\sigma \in S_n$, we form a poset P_σ of dimension 2 by taking the points $p_i = (i, \sigma^{-1}(i)) \in \mathbb{R}^2$, with the standard product ordering. A linear extension of P_σ is a function from the p_i's to $\{1, 2, \ldots, n\}$, which induces a permutation τ as described in Section 2. Then τ is a linear extension of P_σ if and only if $\tau(i) < \tau(j)$ whenever $i < j$ and $\sigma^{-1}(i) < \sigma^{-1}(j)$. When this holds, for $\omega = \tau^{-1}$ we have $\omega \leq \sigma$ in the weak Bruhat order, so that $\boldsymbol{e}(P_\sigma) = \boldsymbol{e}(\sigma)$.

Conversely, given a poset P of dimension two, it can be represented as a collection of points $p_i \in \mathbb{R}^2$ with the product ordering. We translate the points of p_i so that they are all in the first quadrant, and then, for some sufficiently small $\varepsilon > 0$, perform the affine transformation:

$$p_i \mapsto \begin{pmatrix} 1 & \varepsilon \\ \varepsilon & 1 \end{pmatrix} p_i.$$

This transformation ensures that no two points are in the same row or column without changing the ordering on P. Label the points from 1 to n, reading from left to right, and replace the x-coordinates with these labels. Similarly, replace the y-coordinates with the labels 1 through n, read from bottom to top. The points now represent the poset P_σ, for some $\sigma \in S_n$. We thus have $\boldsymbol{e}(P) = \boldsymbol{e}(P_\sigma) = \boldsymbol{e}(\sigma)$. □

3.2. **Rigid circuits.** In this subsection, we define rigid circuits, which will be the principal gadget in our proof of Theorem 1.4. Visually, a rigid circuit consists of a collection of wires laid out in the plane. The wires run horizontally, from left to right. They carry a binary signal, with a 1 representing TRUE, and a 0 representing FALSE. Adjacent wires can feed into logic gates, where they interact in some way; wires cannot cross except at logic gates.

At the far left of the picture, the wires represent binary inputs. The bottom wire at the far right is the output wire. The circuit is satisfied by a choice of inputs if the output wire reads TRUE. Formally, we give the following definitions:

A *circuit state* with k wires is a vector $v \in \mathbb{F}_2^k$. A *general rigid circuit* with m circuit states and k wires is a sequence of m circuit states $(v_1, \ldots, v_m)$, each with k wires, together with a list of relations $(L_1, \ldots, L_{m-1})$ on $\mathbb{F}_2^k$, such that $(v_i, v_{i+1}) \in L_i$, for $1 \leq i \leq m-1$. The relations L_i we call *logic gates*.

We next define *specialized rigid circuits*, which are the circuits we will use throughout the paper, by restricting our choice of logic gates. We define four *simple logic gates* as follows.

IDENTITY gate L_1: The identity function from $\mathbb{F}_2 \to \mathbb{F}_2$.
SWAP gate L_2: A function from $\mathbb{F}_2^2 \to \mathbb{F}_2^2$ that sends $ab \to ba$, for $a, b \in \mathbb{F}_2$.
ANDOR gate L_3: A function from $\mathbb{F}_2^2 \to \mathbb{F}_2^2$ that sends $ab \to (a \text{ AND } b)(a \text{ OR } b)$, where AND and OR are bitwise operations, for $a, b \in \mathbb{F}_2$.

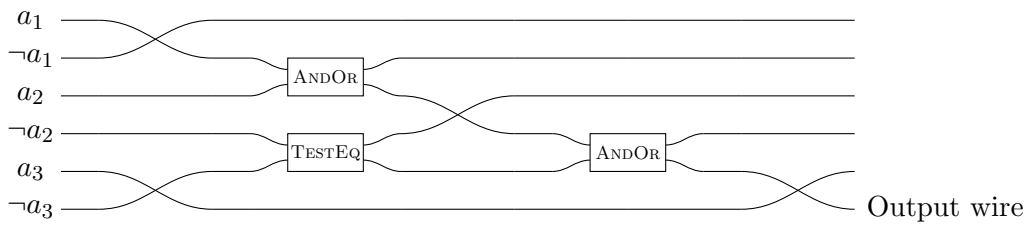

FIGURE 1. A specialized rigid circuit C with $\boldsymbol{e}(C) = 4$. We force $\neg a_2 = \neg a_3$, and the output wire carries the value of the clause $(a_1 \vee a_2 \vee \neg a_2)$.

TESTEQ gate L_4: A relation on $\mathbb{F}_2^2$ that contains $\big\{(11, 11), (00, 00)\big\}$.

Note that the TESTEQ gate merely copies the signal when both wires share the same truth value. If the wires contain different truth values, there is no acceptable next circuit state. In this case, we say the circuit *shorts out*.

Let v and v' be circuit states with k and k' wires, respectively. We define the *coupling* of v and v', which we write as $v \wedge v'$, by concatenating the entries of v and v' to give a circuit state with $k + k'$ wires. Let L and L' be logic gates on k and k' wires, respectively. We define the *coupling* of L and L', which we write as $L \wedge L'$, to be the relation on $\mathbb{F}_2^{k+k'}$ where $(v_1 \wedge v_1', v_2 \wedge v_2') \in L \wedge L'$ precisely when $(v_1, v_2) \in L$ and $(v_1', v_2') \in L'$. A *compound logic gate* is a logic gate made by coupling together copies of the four simple logic gates.

Note that a compound logic gate on (v_i, v_{i+1}) determines v_{i+1} from v_i as long as the circuit does not short out. In our construction, it is sufficient to use compound logic gates where all but one of the gates coupled together are IDENTITY gates. By abuse of notation, we still generally refer to such a compound logic gate by the one simple gate in the coupling that is not an IDENTITY gate. So, for example, a compound logic gate that swaps the wires in positions i and $i+1$ and otherwise is made up of IDENTITY gates we will call a SWAP gate.

A *specialized rigid circuit* is a general rigid circuit with m circuit states and $2k$ wires, such that each logic gate is a compound logic gate and the initial circuit state $v_1 = (a_1, \ldots, a_{2k})$ has exactly one of each pair a_{2i-1}, a_{2i} set to TRUE. We therefore relabel the entries of v_1 as $(a_1, \neg a_1, \ldots, a_k, \neg a_k)$, where $\neg$ denotes bitwise NOT. A *satisfying assignment* of a circuit C is a choice of v_1 such that the circuit does not short out and the last term of v_m is set to TRUE.

We refer to circuits by the capital letter C, and call the number of satisfying assignments $\boldsymbol{e}(C)$. We can now state the following:

#RIGIDCIRCUIT
Input: A specialized rigid circuit C.
Output: The number $\boldsymbol{e}(C)$ of satisfying assignments of C.

Throughout this paper, we will refer to specialized rigid circuits simply as rigid circuits or as circuits when our meaning is clear. Before moving on, we observe the following:

Lemma 3.2. *For every rigid circuit C with a satisfying assignment v_1, there will be exactly k wires set to* TRUE *in each circuit state.*

Proof. There must be k wires set to TRUE in v_1. We note that none of the four simple gates can change the number of TRUE wires, which completes the proof. □

Now we give the first step of our reduction from #3SAT.

Lemma 3.3. *There is a parsimonious reduction from* #3SAT *to* #RigidCircuit.

Proof. Let I be an instance of #3SAT with u variables and v clauses. We form a rigid circuit with $6v$ wires, so that there is one pair of wires for each time a variable or its negation appears in a clause.

We label these $6v$ wires as $(a_1, \neg a_1, a_2, \neg a_2, \ldots, a_{3v}, \neg a_{3v})$. We want some of these wires to represent multiple instances of the same variable. To force $a_i = a_j$, use Swap gates to move a_i and a_j next to each other, and then run them through a TestEq gate. The circuit will then short out unless both $a_i = a_j$ and $\neg a_i = \neg a_j$.

We then use Swap gates to re-arrange the variables so that the order of variables in the first $3v$ wires match the clauses of I. We use two AndOr gates on each clause to produce the desired disjunctions. At this point in the construction, the $3i$-th wire carries the value of the i-th clause, for i between 1 and k. Now, we use more Swap gates to move these k wires to the far left of the circuit state, and use $(k-1)$ AndOr gates to compute the conjunction of all of the clauses, which ends up in the first wire. Finally, we swap the first wire into the last position of our circuit state.

It takes $O(v)$ uses of Swap gates to move any two wires adjacent to each other, so this entire process requires $m = O(v^2)$ circuit states. □

3.3. **Primes and circuits in the Bruhat order.** There is no parsimonious reduction from #RigidCircuit to #Bruhat, because there exist rigid circuits with no satisfying assignments, but every element $\sigma \in S_n$ has $\boldsymbol{e}(\sigma) \geq 1$. Instead, we will use a collection of permutations σ that allow us to compute the residue of #RigidCircuit modulo enough primes that we can then use the Chinese Remainder Theorem to compute #RigidCircuit.

We need the following number theory result:

Proposition 3.4 (see e.g. [BW91, p. 4])**.** *For $k \geq 4$, the product of primes between k and k^2 is at least $2^k k!$.*

In Section 4 we will prove the following:

Main Lemma 3.5. *For every rigid circuit C with m circuit states and $2k$ wires, $k > 7$, and every prime p between k and k^2, there is $n = O(mk^{10})$, and $\sigma \in S_n$, such that $\boldsymbol{e}(C) \equiv -\boldsymbol{e}(\sigma) \mod p$.*

Proof of Theorem 1.4. We construct a polynomial time reduction from #3SAT to #Bruhat. Given a problem in #3SAT, we first apply Lemma 3.3 to obtain a rigid circuit C with m circuit states and $2k$ wires. We next apply Lemma 3.5 to find, for each prime p between k and k^2, some choice of n and $\sigma \in S_n$ with $\boldsymbol{e}(C) \equiv -\boldsymbol{e}(\sigma) \mod p$. Then, as in [BW91], we use the Chinese Remainder Theorem to compute the residue of $\boldsymbol{e}(C)$ modulo the product of primes between k and k^2.

Since there are at most 2^k satisfying assignments of a particular rigid circuit, applying Proposition 3.4 completes the proof. □

4. Circuit constructions

4.1. **Bruhat circuits.** To prove Lemma 3.5, we produce a permutation σ that emulates the design in §3.2. We build a Bruhat circuit, with Bruhat circuit states, simple Bruhat logic gates, and compound Bruhat logic gates.

We need to modify our circuits as follows. Let C be a specialized rigid circuit with $2k$ wires and m circuit states. First, we add $(p-k)$ pairs of wires and use TestEq gates to

set the value of $a_{k+1}, a_{k+2}, \ldots, a_p$ equal to the initial value of a_1. We then stack $(p-1)$ copies of this modified circuit together, and use TestEq gates to ensure that each copy of the circuit will have the same initial assignment as every other copy.

Next, whenever a TestEq or AndOr gate acts on a pair of wires (in any copy of the circuit), we use Swap gates to move those wires to the first two positions of the circuit state vector. We perform the desired TestEq or AndOr operation, and then use Swap gates to put the wires back in their previous positions.

Finally, we use Swap gates to bring the last wire of each circuit copy into the final $(p-1)$ positions of our final circuit state. We write C_p for the resulting circuit, and call it a *mod-p parallel circuit*.

The motivation for these modifications comes later, in the technical requirements of Lemma 4.3 and the constructions in §4.4 and §4.6. For now, though, we note that $\boldsymbol{e}(C) = \boldsymbol{e}(C_p)$, and, by Lemma 3.2, in every valid circuit assignment, each circuit state of C_p will contain exactly $(p^2 - p)$ wires set to TRUE.

A *Bruhat circuit framework* is a permutation $\sigma \in S_n$ together with a classification of the elements in $\{1, 2, \ldots, n\}$ into one of three categories.

The *separators* are a list of elements $s_1 < s_2 < \cdots < s_m$ with $\sigma^{-1}(s_1) < \cdots < \sigma^{-1}(s_m)$. By convention we let $s_0 = \sigma^{-1}(s_0) = 0$ and $s_{m+1} = \sigma^{-1}(s_{m+1}) = n + 1$ where needed. For each remaining element x, there is some i, with $0 \leq i \leq m$, such that

$$\sigma^{-1}(s_i) < \sigma^{-1}(x) < \sigma^{-1}(s_{i+1}).$$

We require either

$$s_i < x < s_{i+1},$$

in which case we call x a *stable element*, or

$$s_{i-1} < x < s_i,$$

in which case we call x a *variable*.

We require that for each choice of i, with $1 \leq i \leq m$ there are

$$N = 2p^2 - 2p$$

variables. We label the variables satisfying $\sigma^{-1}(s_i) < \sigma^{-1}(x) < \sigma^{-1}(s_{i+1})$ as $x_{i1} > x_{i2} > \cdots > x_{iN}$. We require further that $\sigma^{-1}(x_{i1}) < \sigma^{-1}(x_{i2}) < \cdots < \sigma^{-1}(x_{iN})$.

We now make the following essential observations. Let $\tau \in S_n$ be chosen with $\tau \leq \sigma$. Let x be a stable element and x_{ij} be a variable satisfying

$$\sigma^{-1}(s_i) < \sigma^{-1}(x), \sigma^{-1}(x_{ij}) < \sigma^{-1}(s_{i+1}).$$

Then:

$$\tau^{-1}(s_1) < \cdots < \tau^{-1}(s_m) \qquad \text{and} \qquad \tau^{-1}(s_i) < \tau^{-1}(x) < \tau^{-1}(s_{i+1}),$$

and either

$$\tau^{-1}(s_i) < \tau^{-1}(x_{ij}) < \tau^{-1}(s_{i+1}) \qquad \text{or} \qquad \tau^{-1}(s_{i-1}) < \tau^{-1}(x_{ij}) < \tau^{-1}(s_i).$$

Given a Bruhat circuit framework σ and some $\tau \leq \sigma$, for $1 \leq i \leq m$ we assign to τ a *Bruhat circuit state* $v_i \in \mathbb{F}_2^N$ as follows. Write $v_i = (a_{i1}, a_{i2}, \ldots, a_{iN})$, with $a_{ij} \in \mathbb{F}_2$. Then take $a_{ij} = 1$ if $\tau^{-1}(s_i) < \tau^{-1}(x_{ij}) < \tau^{-1}(s_{i+1})$, and $a_{ij} = 0$ otherwise.

Note that the since the y_{ij}'s are arranged in strictly decreasing order, they can be rearranged arbitrarily in τ, so that every possible circuit state can be realized as a Bruhat circuit state. In particular, for every possible circuit assignment $(v_1, \ldots, v_m)$, there is a unique permutation τ with Bruhat circuit state equal to $(v_1, \ldots, v_m)$ maximal in the Bruhat order. It is obtained by moving each variable which takes the value FALSE in circuit state v_i

immediately to the left of s_i, keeping those FALSE variables in descending order. Call this permutation $\tau|v_1, \ldots, v_m$.

In summary, we have:

$$\boldsymbol{e}(\sigma) \;=\; \sum_{(v_1,\ldots,v_m)} \boldsymbol{e}(\tau|v_1,\ldots,v_m), \tag{4.1}$$

where the sum is taken over every possible set of circuit states $(v_1, \ldots, v_m)$. In the next four subsections, we will show how to control the value of $\boldsymbol{e}(\tau|v_1, \ldots, v_m)$ to encode the logic of our circuit.

4.2. **Bruhat logic gates.** A *Bruhat logic gate* with k wires is a sequence ϕ of distinct integers such that the smallest k terms are in decreasing order, the last k terms are in decreasing order, and these two sets do not overlap. We refer to these elements as the input and output variables, respectively, of the logic gate.

For technical reasons, we also require that immediately preceding the last k terms is a block of $(p^3 - 1)$ consecutive elements, all less than the last k terms. We refer to this block, appropriately, as the *penultimate block*.

If $|\phi| = \ell$, we do not require ϕ to take values strictly in the set $\{1, \ldots, \ell\}$, but we still treat ϕ as a member of S_ℓ, as described in Section 2

The *evaluation* of a Bruhat logic gate ϕ with k wires at some pair of circuit states $(v_1, v_2) \in \mathbb{F}_2^k$ is given by deleting from ϕ each of the input variables corresponding to a 0 in v_1 and each of the output variables corresponding to a 1 in v_2. We write this as $\phi \rtimes (v_1, v_2)$.

Given a Bruhat circuit framework σ, we write down a collection of sequences $(\sigma_1, \ldots, \sigma_{m+1})$ as follows. For σ_i, write down all the elements of σ (taken in one line notation) between s_{i-1} and s_i, and then write down only the variables that occur between s_i and s_{i+1}.

Note that, for all i satisfying $2 \le i \le m$, the sequence σ_i is a Bruhat logic gate with $N = 2p^2 - 2p$ wires. Also note that the choice of $(\sigma_1, \ldots, \sigma_{m+1})$ determines our original Bruhat circuit framework σ uniquely.

For a given set of circuit states $(v_1, \ldots, v_m)$, we similarly define the sequence $(\tau_1, \ldots, \tau_{m+1})$, by writing $\tau|v_1, \ldots, v_m$ in one-line notation and breaking it apart at each separator s_i. Note that $\tau_i = \sigma_i \rtimes (v_{i-1}, v_i)$, for $2 \le i \le m$. By abuse of notation, we set $v_0 = v_{m+1} = \varnothing$, and let $\sigma_1 \rtimes (v_0, v_1) = \tau_1$ and $\sigma_{m+1} \rtimes (v_m, v_{m+1}) = \tau_{m+1}$.

Though the sequences τ_i are not permutations, we can treat them as permutations as described in Section 2, and so compute $\boldsymbol{e}(\tau_i)$. We can now rewrite (4.1) as

$$\boldsymbol{e}(\sigma) = \sum_{(v_1,\ldots,v_m)} \prod_{i=1}^{m+1} \boldsymbol{e}(\tau_i) \;=\; \sum_{(v_1,\ldots,v_m)} \prod_{i=1}^{m+1} \boldsymbol{e}\big(\sigma_i \rtimes (v_{i-1}, v_i)\big), \tag{4.2}$$

where the sum is taken over every possible set of circuit states $(v_1, \ldots, v_m)$.

We must have this product take the value 0 modulo p whenever $(v_1, \ldots, v_m)$ is not a satisfying assignment of C_p, and to take some nonzero constant value otherwise.

The simplest way to do this would be to construct Bruhat logic gates σ_i so that

$$\boldsymbol{e}\big(\sigma_i \rtimes (v_{i-1}, v_i)\big) \equiv 0 \mod p\,, \quad \text{when } (v_{i-1}, v_i) \notin L_{i-1},$$

and

$$\boldsymbol{e}\big(\sigma_i \rtimes (v_{i-1}, v_i)\big) \equiv 1 \mod p, \quad \text{when } (v_{i-1}, v_i) \in L_{i-1}.$$

To keep computations manageable, we weaken the condition by sometimes only requiring

$$\boldsymbol{e}\big(\sigma_i \rtimes (v_{i-1}, v_i)\big) \not\equiv 0 \mod p\,, \quad \text{when } (v_{i-1}, v_i) \in L_{i-1}.$$

In §4.6, we explain how to complete the proof of Main Lemma 3.5 under these weakened conditions. We postpone the construction of σ_1 and σ_{m+1} until §4.4.

As with rigid circuits, we say that a Bruhat circuit ϕ *shorts out* at v if, for every choice of v', we have

$$\boldsymbol{e}\big(\phi \rtimes (v, v')\big) \equiv 0 \mod p.$$

4.3. **Bruhat compound logic gates.** In this subsection, we explain how to couple two Bruhat logic gates ϕ with k wires and ϕ' with k' wires to produce a new Bruhat logic gate $\phi \wedge \phi'$ with $k + k'$ wires, emulating the behavior of coupled logic gates defined in §3.2.

First, we give a technical construction to ensure that the number of wires carrying the value TRUE remains constant through logic gates. This matches the statement for rigid circuits given in Lemma 3.2. Given a Bruhat logic gate ϕ with k wires, we say ϕ is *balanced* modulo p if $|\phi| - k \equiv 0 \mod p^3$.

To construct the *restriction* of ϕ, which we write $\phi_\circ$, we append to the beginning of ϕ the block $(\max(\phi) + 1, \max(\phi) + 2, \ldots, \max(\phi) + p^3 - 1)$, which we call the *initial block* of $\phi_\circ$. We have:

Lemma 4.1. *For a Bruhat logic gate ϕ that is balanced modulo p, we have*

$$\boldsymbol{e}\left(\phi_\circ \rtimes (v, v')\right) \equiv 0 \mod p$$

when v and v' do not have an equal number of wires carrying the value TRUE*. When v and v' do have an equal number of wires carrying the value* TRUE*, we have*

$$\boldsymbol{e}\left(\phi_\circ \rtimes (v, v')\right) \equiv \boldsymbol{e}\left(\phi \rtimes (v, v')\right) \mod p.$$

The proof of this lemma is given in Subsection 5.1.

We now return to the construction of $\phi \wedge \phi'$. We restrict to the case where ϕ is one of our simple Bruhat gates, and where ϕ' is a compound gate made up of IDENTITY and SWAP gates, since the construction in §4.1 requires us to place every TESTEQ or ANDOR gate at the top of our circuit. We require further that ϕ and ϕ' be balanced modulo p.

The construction of $\phi \wedge \phi'$ involves inserting $\phi'_\circ$ in place of the penultimate block of ϕ, shifting elements appropriately. To explain these shifts, we replace each of the elements of ϕ and $\phi'_\circ$ with ordered pairs of integers.

Let y be the value of the first entry of the penultimate block of ϕ. Replace each of the elements x in ϕ with the ordered pair $(x, 0)$. Replace the input variables x of $\phi'_\circ$ with $(0, x)$, and all other elements x of $\phi'_\circ$ with (y, x). Then delete the penultimate block of ϕ and insert the relabeled $\phi'_\circ$ in its place.

Now relabel the entries from 1 to $|\phi| + |\phi'|$, going in order from smallest to largest with respect to the lexicographical order on $\mathbb{Z}^2$. Call the result $\phi \wedge \phi'$. Note that $\phi \wedge \phi'$ is a Bruhat logic gate with $k + 1$ wires when ϕ is the IDENTITY gate, and $k + 2$ wires otherwise, and that $\phi \wedge \phi'$ is balanced modulo p.

We define the following operations on logic gates:

Definition 4.2. *Left insertion*, *middle insertion* and *right insertion*, denoted $L(\phi)$, $M(\phi)$ and $R(\phi)$, respectively, are operators on Bruhat logic gates defined as follows. The terms left, middle and right are all oriented with respect to the penultimate block. Left insertion inserts the element 1 into ϕ immediately to the left of the penultimate block, and shifts all other elements up by 1. Middle insertion increases the length of the penultimate block by 1. Right insertion inserts an element one larger than the largest element in the penultimate block to the very end of ϕ, and shifts all larger elements up by 1.

Also, let $M^{-1}(\phi)$ denote the inverse operation to M where we decrease the length of the penultimate block by 1.

The following lemma gives the set of conditions required for the coupling of logic gates to behave as desired. These equations produce the polynomials given in Appendix A that are used in §4.6 to complete the proof of Main Lemma 3.5.

Lemma 4.3. *Given ϕ and ϕ' as above, if ϕ is not the identity gate, $(\phi \wedge \phi')_\circ$ behaves as the coupling of the logic gates associated to $\phi_\circ$ and $\phi'_\circ$ when the following six equations are satisfied:*

$$|\phi| - 2 \equiv 0 \bmod p^3, \tag{1}$$

$$-2\boldsymbol{e}\big(M(\phi_\circ) \rtimes (10,11)\big) + \boldsymbol{e}\big(L(\phi_\circ) \rtimes (10,11)\big) + \boldsymbol{e}\big(R(\phi_\circ) \rtimes (10,11)\big) \equiv 0 \bmod p, \tag{2}$$

$$-2\boldsymbol{e}\big(M(\phi_\circ) \rtimes (01,11)\big) + \boldsymbol{e}\big(L(\phi_\circ) \rtimes (01,11)\big) + \boldsymbol{e}\big(R(\phi_\circ) \rtimes (01,11)\big) \equiv 0 \bmod p, \tag{3}$$

$$-2\boldsymbol{e}\big(M(\phi_\circ) \rtimes (00,01)\big) + \boldsymbol{e}\big(L(\phi_\circ) \rtimes (00,01)\big) + \boldsymbol{e}\big(R(\phi_\circ) \rtimes (00,01)\big) \equiv 0 \bmod p, \tag{4}$$

$$-2\boldsymbol{e}\big(M(\phi_\circ) \rtimes (00,10)\big) + \boldsymbol{e}\big(L(\phi_\circ) \rtimes (00,10)\big) + \boldsymbol{e}\big(R(\phi_\circ) \rtimes (00,10)\big) \equiv 0 \bmod p, \tag{5}$$

$$\begin{aligned} &2\boldsymbol{e}\big(M^2(\phi_\circ) \rtimes (00,11)\big) - 4\boldsymbol{e}\big(LM(\phi_\circ) \rtimes (00,11)\big) - 4\boldsymbol{e}\big(RM(\phi_\circ) \rtimes (00,11)\big) + \\ &\boldsymbol{e}\big(L^2(\phi_\circ) \rtimes (00,11)\big) + 2\boldsymbol{e}\big(LR(\phi_\circ) \rtimes (00,11)\big) + \boldsymbol{e}\big(R^2(\phi_\circ) \rtimes (00,11)\big) \equiv 0 \bmod p. \end{aligned} \tag{6}$$

When ϕ is the identity gate, we need two equations to be satisfied:

$$|\phi| - 1 \equiv 0 \bmod p^3, \tag{7}$$

$$-2\boldsymbol{e}\big(M(\phi_\circ) \rtimes (0,1)\big) + \boldsymbol{e}\big(L(\phi_\circ) \rtimes (0,1)\big) + \boldsymbol{e}\big(R(\phi_\circ) \rtimes (0,1)\big) \equiv 0 \bmod p. \tag{8}$$

The proof of this lemma is given in Subsection 5.2.

4.4. **Initializing and testing wires.** We now give explicitly the construction of σ_1 and σ_{m+1}, to initialize wires at the beginning of our circuit and test the value of the output wire at the end.

For σ_1, we begin with ψ, a compound Bruhat logic gate consisting of $p^2 - p$ copies of the identity wire, and then take $\sigma_1 = \psi_\circ \rtimes (\vec{0}, \vec{0})$, where $\vec{0}$ represents a circuit state with all wires set to FALSE. The identity gate construction given in Lemma 4.7 is simple enough that we can state what σ_1 looks like explicitly. It contains a sequence of $p^2 - p + 1$ blocks, each of size $p^3 - 1$, followed by the variables. The blocks themselves decrease by p^3 with each block step, and the variables fill in the missing terms.

For example, when $p = 2$, σ_1 has 2 wires, and we have

$$\sigma_1 = \boxed{17}\ \boxed{9}\ \boxed{1}\ 16\ 8,$$

where each of the numbers in boxes represent blocks of size $p^3 - 1$, using the $\boxed{x}$ notation as in §4.1 above. For ease in notation, we shift elements down and write instead

$$\sigma_1 = \boxed{5}\ \boxed{3}\ \boxed{1}\ 4\ 2.$$

We then modify σ_1 by duplicating each of the $p^2 - p$ terms at the end, in their respective positions. Our example above becomes

$$\sigma_1 = \boxed{5}\ \boxed{3}\ \boxed{1}\ 4\ 4\ 2\ 2.$$

Finally, we shift all elements of σ_1 up so that all of the elements are distinct and the final sequence is in strictly decreasing order. Our example now reads

$$\sigma_1 = \boxed{7}\ \boxed{4}\ \boxed{1}\ 6\ 5\ 3\ 2.$$

Note that σ_1 now has $N = 2p^2 - 2p$ output wires, as required.

Recall that, by the abuse of notation introduced in §4.2, for a vector

$$v = (a_1, \ldots, a_N) \in \mathbb{F}_2^N,$$

we let $\sigma_1(v_0, v)$ represent σ_1 after deleting every element corresponding to a 1 in v, where we take $v_0 = \varnothing$.

Having given the details of our construction, we now prove the following:

Lemma 4.4. *We have $\sigma_1 \rtimes (v_0, v) \equiv 1 \mod p$ precisely when exactly one of each pair $\{a_{2i-1}, a_{2i}\}$ is equal to 1, and $\sigma_1 \rtimes (v_0, v) \equiv 0 \mod p$ otherwise.*

Proof. By construction, when exactly one of each pair $\{a_{2i-1}, a_{2i}\}$ is equal to 1, then $\sigma_1 \rtimes (v_0, v) = \psi_\circ \rtimes (\vec{0}, \vec{0})$, so $\boldsymbol{e}(\sigma_1(v)) = \boldsymbol{e}(\psi_\circ \rtimes (\vec{0}, \vec{0})) \equiv 1 \mod p$. And, by Lemma 4.1, $\boldsymbol{e}(\sigma_1 \rtimes (v_0, v)) \equiv 0 \mod p$ unless $|v| = p^2 - p$.

The only case left to consider is when $|v| = p^2 - p$, but the condition of this lemma is false. In this case, there must be some i for which both a_{2i-1} and a_{2i} are equal to 1. However, σ_1 will then contain two adjacent blocks of size $p^3 - 1$ with no other elements whose values lie between those two blocks. In our example above, deleting both the 6 and the 5 leaves adjacent blocks $\boxed{7}$ and $\boxed{4}$ in σ_1.

The rearrangement of these two blocks are therefore independent of the rearrangement of the rest of σ_1, so that $\boldsymbol{e}(\sigma_1 \rtimes (v_0, v))$ is divisible by

$$\binom{2p^3 - 2}{p^3 - 1}.$$

This is $p^3 C_{p^3-1}$, where C_{p^3-1} is the $p^3 - 1$-th Catalan number. Thus the binomial coefficient is divisible by p, so $\boldsymbol{e}(\sigma_1 \rtimes (v_0, v)) \equiv 0 \mod p$, as desired. □

We now give the construction of σ_{m+1}. Recall, by the construction given in §4.1, that the final $p-1$ wires should all carry the value of the output wire, while the other $N - (p-1) = 2p^2 - 3p + 2$ wires are grouped into $p - 1$ sets of $2p - 1$ wires.

We begin with

$$\sigma_{m+1} = (N, \ldots, 2, 1).$$

This choice is forced on us, since the input variables always must be in decreasing order and the smallest elements of the permutation. To finish our construction, we insert the sequence

$$\big(N + 1, N + 2, \ldots, N + (p - 1)\big)$$

into σ_1 so as to divide the variables into the sets described above, i.e. first $p-1$ sets of $2p-1$ wires, followed by a final set of $p - 1$ wires.

We call the sequence $\big(N+1, N+2, \ldots, N+(p-1)\big)$ *dividers*. Again, by abuse of notation, given a vector $v \in \mathbb{F}_2^N$, we write $\sigma_{m+1} \rtimes (v, \varnothing) = \sigma_{m+1} \rtimes (v, v_{m+1})$ for the sequence obtained by deleting from σ_{m+1} each variable corresponding to a 0 in v.

Lemma 4.5. *If the last element of v is 1, i.e. if the output wire contains the value* TRUE*, then $\boldsymbol{e}(\sigma_{m+1} \rtimes (v, v_{m+1})) \equiv -1 \mod p$. Otherwise, $\boldsymbol{e}(\sigma_{m+1} \rtimes (v, v_{m+1})) \equiv 0 \mod p$.*

Proof. By construction, the entire final set of $(p-1)$ wires will either all be TRUE or all be FALSE. Before all the swapping we did at the end of our circuit in §4.1, each of the original $p-1$ circuits contained $2p$ wires, with p wires set to TRUE. So, after the swapping, if the final set of wires is TRUE, each of the other sets will have $p-1$ wires set to TRUE, while if the final set of wires is FALSE, each of the other sets will have p wires set to TRUE.

We treat each case separately:

Case 1: The desired wire carries the value FALSE. Then the p wires set to TRUE in each of the other $p-1$ sets correspond in $\sigma_{m+1}(v, v_{m+1})$ to a sequence of p consecutive decreasing elements. There are $p!$ rearrangements of each of those sets, and the rearrangements of those sets are independent of the rearrangement of the rest of the elements in the permutation, so that $\boldsymbol{e}(\sigma_{m+1}(v, v_{m+1})) \equiv 0 \mod p$.

Case 2: The desired wire carries the value TRUE. Then the $p-1$ wires set to TRUE in each of the other $p-1$ sets correspond in $\sigma_{m+1}(v, v_{m+1})$ to a sequence of $p-1$ consecutive decreasing elements. The final set of wires also contains $p-1$ decreasing elements. We count the number of rearrangements $\tau \leq \sigma_{m+1}(v, v_{m+1})$ based on the position of the dividers in τ.

When the dividers remain in exactly the same position, then no other element can move out of its set either, since the variables in τ must remain to the left of every divider they were already to the left of in σ. This leaves only rearrangements within the p sets of $p-1$ strictly decreasing elements, for a total count of $\big((p-1)!\big)^p \equiv (-1)^p \equiv -1 \mod p$ by Wilson's theorem.

For all other choices of positions for the dividers, there will be some gap of size at least p between dividers. The number of ways to rearrange the at least p elements that fill this gap will be divisible by $p!$ These rearrangements are independent of the rearrangement of the rest of the sequence, and so the contribution from every other choice of positions for the dividers is 0 modulo p. Thus the total number of rearrangements is congruent to $-1 \mod p$, as desired. □

4.5. **Parametrized gates.** We now describe how to construct a parametrized family of logic gates whose count of rearrangements is a polynomial in the parameters. We use this construction in §4.6 to give SWAP, ANDOR and TESTEQ gates for arbitrary primes p.

Let ϕ be a logic gate containing an increasing sequence $(x_1, \ldots, x_t)$. We form the *parametrization* of ϕ with respect to (x_i) by replacing each of the elements x_i with a block of consecutive elements of length $z_i \in \mathbb{N}$ and shifting the other elements of ϕ up appropriately, and denote this as $\boxed{\phi}(x_i, z_i)$.

We require that the sequence of x_i's conclude prior to the penultimate block of ϕ, and by convention write x_{t+1} for the penultimate block of ϕ, with $z_{t+1} = p^3 - 1$.

Lemma 4.6. *For every parametrization $\boxed{\phi}(x_i, z_i)$ of a logic gate ϕ, there is a polynomial $g(z_1, \ldots, z_{t+1})$ over $\mathbb{Q}$ such that $g(z_1, \ldots, z_t, p^3 - 1) = \boldsymbol{e}\left(\boxed{\phi}(x_i, z_i)\right)$ for every p.*

The proof of this lemma is given in Subsection 5.3.

4.6. **Mod-p modification.** Recall that for a logic gate L, we say $(v_1, v_2) \in L$ whenever (v_1, v_2) satisfies the logic gate, and $(v_1, v_2) \notin L$ otherwise. For computational reasons, it is easier to find simple logic gates if we relax the condition that $\boldsymbol{e}(\sigma \rtimes (v_1, v_2)) \equiv 1 \mod p$ whenever (v_1, v_2) satisfies the logic gate. We never alter the set of conditions $\boldsymbol{e}(\sigma \rtimes (v_1, v_2)) \equiv 0 \mod p$ when (v_1, v_2) fails to satisfy the logic gate. We describe the modified conditions below.

IDENTITY gate L_1: $\boldsymbol{e}\big(\sigma \rtimes (v_1, v_2)\big) \equiv 1 \mod p$ whenever $(v_1, v_2) \in L_1$.
SWAP gate L_2: $\boldsymbol{e}\big(\sigma \rtimes (v_1, v_2)\big) \equiv 1 \mod p$ whenever $(v_1, v_2) \in L_2$.
ANDOR gate L_3: $\boldsymbol{e}\big(\sigma \rtimes (v_1, v_2)\big) \not\equiv 0 \mod p$ whenever $(v_1, v_2) \in L_3$.
TESTEQ gate L_4: $\boldsymbol{e}\big(\sigma \rtimes (v_1, v_2)\big) \not\equiv 0 \mod p$ whenever $(v_1, v_2) \in L_4$.

We now have all of the conditions on our simple Bruhat logic gates, allowing us to state and prove the following:

Lemma 4.7. *For every prime $p \geq 2$, there is an* IDENTITY *gate satisfying the condition of Lemma 4.3. In addition, for each of* SWAP, ANDOR *and* TESTEQ, *there are parametrized Bruhat logic gates ϕ such that the conditions above together with the conditions in Lemma 4.3 give a system of polynomial equations in the parameters $\{z_i\}$ that has solutions modulo p for all primes $p \geq 11$.*

Proof. We prove the statement for each gate by giving an explicit construction. The following permutations represent our four desired gates before restriction. The IDENTITY gate works correctly modulo p without parametrization. The other three gates are parametrized with respect to the sequence $\{3, 4, 5, 6, 7\}$. The $\boxed{2}$ in the IDENTITY gate and the $\boxed{8}$ in the other three gates represent the penultimate blocks of size $p^3 - 1$.

We treat each of the equations in Lemma 4.3 first over $\mathbb{Q}$, so that the equations correspond to some algebraic variety over $\mathbb{Q}$. For each gate, we are able to give explicitly a rational point on that variety, and the rational nonzero values taken by the ANDOR and TESTEQ gates. For every prime $p \geq 11$, this corresponds to a solution modulo p.

(1) IDENTITY gate:

$$\phi = 1\ \boxed{2}\ 3.$$

The equations that ϕ must satisfy are (7) and (8) from Lemma 4.3, and the following four equations:

$$\boldsymbol{e}\big(\phi \rtimes (0,0)\big) = \boldsymbol{e}\big(\phi \rtimes (1,1)\big) = 1, \quad \boldsymbol{e}\big(\phi \rtimes (0,1)\big) = \boldsymbol{e}\big(\phi \rtimes (1,0)\big) = 0.$$

The last two equations are guaranteed to be satisfied by Lemma 4.1. Since $|\phi| = p^3 + 1$, we have that (7) is satisfied. Note that

$$\begin{aligned}\boldsymbol{e}\big(\phi \rtimes (0,0)\big) &= \boldsymbol{e}\big(\phi \rtimes (1,1)\big) = \boldsymbol{e}\big(L(\phi) \rtimes (0,1)\big) \\ &= \boldsymbol{e}\big(M(\phi) \rtimes (0,1)\big) = \boldsymbol{e}\big(R(\phi) \rtimes (0,1)\big) = 1,\end{aligned}$$

since these permutations are all just strictly increasing sequences of length p^3. Thus the remaining two equations given here and (8) are satisfied, as desired.

For each of the remaining three gates, there are 6 equations from Lemma 4.3, and an additional 16 equations for every possible pair of input and output wires. However, applying Lemma 4.1 as above, we see that 10 of these equations will be satisfied automatically. For the remainingg 12 equations, we use Lemma 4.6 to compute the corresponding polynomials in $\{z_i\}$, for $1 \leq i \leq 5$. We write out these polynomials explicitly in Appendix A.

(2) SWAP gate:

$$\phi = 2\ \boxed{3}\ 12\ \boxed{4}\ 1\ \boxed{5}\ 10\ \boxed{6}\ 13\ \boxed{7}\ \boxed{8}\ 11\ 9.$$

The system of equations in §A.1 has a unique solution over $\mathbb{Q}$:

$$(z_1, z_2, z_3, z_4, z_5) = (-1, -2, 0, 1, -2),$$

so the system of equations is solvable $\mod p$, for every prime $p \geq 2$.

(3) AndOr gate:

$$\phi = 2\ \boxed{3}\ 13\ \boxed{4}\ 11\ \boxed{5}\ 1\ \boxed{6}\ 10\ \boxed{7}\ \boxed{8}\ 12\ 9.$$

The system of equations in §A.2 reduces to a two-dimensional variety over $\mathbb{Q}$ of degree 2, with infinitely many rational points, including the point

$$(z_1, z_2, z_3, z_4, z_5) = (-2, 1, -3, 1, -1).$$

The nonzero values $\boldsymbol{e}(\sigma(v, v'))$ takes are 2 and 4, so we require $p \neq 2$, and the system of equations is solvable mod p for every prime $p \geq 3$.

(4) TestEq gate:

$$\phi = 2\ \boxed{3}\ 12\ \boxed{4}\ 10\ \boxed{5}\ 1\ \boxed{6}\ 13\ \boxed{7}\ \boxed{8}\ 11\ 9.$$

The system of equations in §A.3 reduces to a one-dimensional variety over $\mathbb{Q}$ of degree 1, with infinitely many rational points, including the point

$$(z_1, z_2, z_3, z_4, z_5) = (-2, -\tfrac{8}{3}, \tfrac{5}{3}, -3, 2),$$

with nonzero values of $\frac{7}{3}$ and $\frac{-8}{3}$ for $\boldsymbol{e}(\sigma(v, v'))$, so that the system of equations is solvable mod p for every prime $p \geq 11$.

□

4.7. **Proof of Main Lemma 3.5.** Given a rigid circuit C, we construct the mod-p parallel circuit C_p with $\boldsymbol{e}(C) = \boldsymbol{e}(C_p)$. We then construct a Bruhat circuit σ that mirrors the behavior of C_p.

By Lemma 4.4, our choices of variable assignments v_1 in the sum in (4.2) are restricted to those with $N = 2p^2 - 2p$ wires grouped in pairs, with exactly one wire set to TRUE and one wire set to FALSE in each pair. By lemmas 4.3 and 4.7, the inside product in (4.2) is congruent to 0 mod p except when $(v_1, \ldots, v_m)$ is a set of circuit states satisfying C_p.

By the parallel circuit construction, every time an AndOr gate operation occurs in C, it occurs $p-1$ times in C_p, acting on the same set of truth values, which gives a contribution of 1 mod p to the product in (4.2).

The same is true for the TestEq operations that occur after the parallel circuit has already been constructed. For the TestEq operations that occur in the construction of the parallel circuit (i.e. the TestEq operations used to force each of the copies of the circuit to have the same initial truth values), just repeat them $p-1$ times, which has no impact on the operation of the circuit.

The Identity and Swap operations all also give a contribution of 1 mod p to the product, by construction.

In summary, the contribution to the product from the operation of each of the gates is 1. The only contribution left comes from σ_{m+1}, which, by Lemma 4.5, multiplies the product by -1 if the output wire is TRUE and 0 otherwise. □

5. Proof of lemmas

5.1. **Proof of Lemma 4.1.** Write $|v|$ for the number of wires carrying the value TRUE in v. Since the elements in the initial block of $\phi_\circ$ are larger than every other element of ϕ, the

Bruhat order gives no restriction on the position of these elements relative to the position of the elements of ϕ in a rearrangement $\tau \leq \phi_\circ$. Thus:

$$\begin{aligned} \boldsymbol{e}\left(\phi_\circ \rtimes (v, v')\right) &= \binom{|\phi \rtimes (v, v')| + p^3 - 1}{p^3 - 1} \boldsymbol{e}\left(\phi \rtimes (v, v')\right) \\ &= \binom{|\phi| - (k - |v|) - |v'| + p^3 - 1}{p^3 - 1} \boldsymbol{e}\left(\phi \rtimes (v, v')\right). \end{aligned}$$

Since ϕ is balanced, we know $|\phi| - k \equiv 0 \mod p^3$. Write $|\phi| - k = ap^3$ and $|v| - |v'| = b$. We have $|b| \leq 2p^2 + 2p$. Observe that, for integers a, b with $a > 0$ and $0 < |b| \leq 2p^2 + 2p$, as long as $p \geq 3$ we have

$$\binom{ap^3 + p^3 - 1 + b}{p^3 - 1} \equiv 0 \mod p.$$

On the other hand, for $a > 0$ and $b = 0$, we have:

$$\binom{ap^3 + p^3 - 1 + b}{p^3 - 1} \equiv 1 \mod p.$$

We thus have $\boldsymbol{e}(\phi_\circ \rtimes (v, v')) \equiv 0 \mod p$ whenever $|v| \neq |v'|$. When $|v| = |v'|$, the binomial coefficient evaluates to 1 modulo p, so that we have $\boldsymbol{e}(\phi_\circ \rtimes (v, v')) \equiv \boldsymbol{e} \rtimes (\phi(v, v'))$, as desired. □

5.2. **Proof of Lemma 4.3.** Equations (1) and (8) follow immediately from the requirement that ϕ be balanced.

We prove the lemma in the case where ϕ is the Swap, AndOr or TestEq gate, and then explain how to adjust the proof when ϕ is the Identity gate. Write the input and output wires of $\phi \wedge \phi'$ as $v_1 \wedge v_1'$ and $v_2 \wedge v_2'$, where here $\wedge$ denotes concatenation, $v_i \in \mathbb{F}_2^2$, and $v_i' \in \mathbb{F}_2^k$, so that there are a total of $k + 2$ input and output wires in $\phi \wedge \phi'$.

For every rearrangement τ of $(\phi \wedge \phi') \rtimes (v_1 \wedge v_1', v_2 \wedge v_2')$, we can restrict to a rearrangement of $\phi_\circ'$ by looking only at the elements that come from $\phi_\circ'$ in τ (and shifting the elements back down to their previous values). Write this new rearrangement as $\tau|_{\phi_\circ'}$. Then, writing $\tau|_{\phi_\circ'}$ in one-line notation, $\tau|_{\phi_\circ'}$ begins with some sequence (possibly of length zero) of input variables, and ends with some sequence (possibly of length zero) of output variables. Call the length of the first sequence $\ell(\tau)$ and the length of the second sequence $r(\tau)$.

Now we wish to begin with a permutation $\tau' \leq \phi' \rtimes (v_1', v_2')$, and count the number of $\tau \leq (\phi \wedge \phi') \rtimes (v_1 \wedge v_1', v_2 \wedge v_2')$ with $\tau|_{\phi_\circ'} = \tau'$. Since we have fixed τ', we need to consider only the possible ways that the elements of ϕ can be rearranged with respect to each other or shuffled among the elements of ϕ'.

By Lemma 4.1, we have:

$$\boldsymbol{e}\big((\phi \wedge \phi') \rtimes (v_1 \wedge v_1', v_2 \wedge v_2')\big) \equiv 0 \mod p$$

unless $|v_1| + |v_1'| \equiv |v_2| + |v_2'| \mod p^3$. Thus, we restrict our attention to the case where that holds. Then, by the same argument used in the proof of Lemma 4.1, we have:

$$|\phi' \rtimes (v_1', v_2')| = |\phi'| + (|v_1'| - k) - |v_2'| \equiv |v_1'| - |v_2'| \equiv |v_2| - |v_1| \mod p^3.$$

Since $\phi_\circ'$ adds a block of size $p^3 - 1$, we have $|\phi_\circ' \rtimes (v_1', v_2')| \equiv |v_2| - |v_1| - 1 \mod p^3$.

We then note that the number of $\tau \leq (\phi \wedge \phi') \rtimes (v_1 \wedge v_1', v_2 \wedge v_2')$ with $\tau|_{\phi_\circ'} = \tau'$ is equal to:

$$\frac{1}{\ell(\tau')!\, r(\tau')!} \cdot \boldsymbol{e}\big(L^{\ell(\tau')} R^{r(\tau')} M^{|v_2| - |v_1| - \ell(\tau') - r(\tau')}(\phi) \rtimes (v_1, v_2)\big).$$

With the $\frac{1}{\ell(\tau')!\,r(\tau')!}$ term because repeated left and right insertion gives sets of consecutive decreasing elements that can be rearranged in $\ell(\tau')!$ and $r(\tau')!$ ways, respectively, but exactly one of these arrangements actually corresponds to τ'.

Next, we group permutations τ' based on the value of $\ell(\tau')$ and $r(\tau')$. Let $g(\ell, r)$ be the number of $\tau' \leq \phi' \rtimes (v_1', v_2')$ with $\ell(\tau') = \ell$ and $r(\tau') = r$. Of course, the value $g(\ell, r)$ also depends on $\phi' \rtimes (v_1', v_2')$. We omit this dependence from our notation for the sake of readability. We then have:

$$\boldsymbol{e}\big((\phi \wedge \phi') \rtimes (v_1 \wedge v_1', v_2 \wedge v_2')\big) \;=\; \sum_{\ell, r \geq 0} \frac{g(\ell, r)}{\ell!\, r!} \cdot \boldsymbol{e}\big(L^\ell R^r M^{|v_2| - |v_1| - \ell - r}(\phi) \rtimes (v_1, v_2)\big). \tag{5.1}$$

It remains to compute $g(\ell, r)$ for an arbitrary choice of $v_1 \wedge v_1'$, $v_2 \wedge v_2'$, ℓ, and r. Beginning or ending τ' with a particular sequence of elements is the same as counting the number of permutations of τ' with those elements removed. Thus we have:

$$\ell!\, r! \sum_{|v_1'| - |w_1'| = \ell} \; \sum_{|w_2'| - |v_2'| = r} \boldsymbol{e}\big(\phi_\circ' \rtimes (w_1', w_2')\big) \;=\; \sum_{\ell \leq h \leq k} \; \sum_{r \leq j \leq k} g(h, j). \tag{5.2}$$

Here the left hand sum is taken over circuit states w_1' obtained from v_1' by flipping ℓ wires from TRUE to FALSE and w_2' obtained from v_2' by flipping r from FALSE to TRUE. The $\ell!$ and $r!$ terms account for the ways to arrange the initial and final sequences, each strictly decreasing, of length ℓ and r respectively.

Either of the wire flips just described reduces $|v_1'| - |v_2'|$ by one, so that we have:

$$|v_2| - |v_1| = |v_1'| - |v_2'| = \ell + r + |w_1'| - |w_2'|.$$

Recall that, by Lemma 4.1, for the left hand side of (5.2) to be nonzero modulo p we must have $|w_1'| - |w_2'| = 0$.

Thus, on one hand, if $\ell + r > |v_1'| + |v_2'|$, the left hand side of (5.2) is always 0 modulo p, so that we conclude $g(h, j) \equiv 0 \mod p$ for every h, j with $h + j > \ell + r$.

On the other hand, if the left hand side of (5.2) is nonzero modulo p, we have

$$|v_2| - |v_1| = |v_1'| - |v_2'| = \ell + r \geq 0.$$

Since $v_1, v_2 \in \mathbb{F}_2^2$, we have $|v_2| - |v_1| \leq 2$, and so we conclude $0 \leq |v_1'| - |v_2'| \leq 2$.

Since ϕ' is composed entirely of IDENTITY and SWAP gates, each input wire in v_1' can be matched with one output wire in v_2' whose signal state it controls. If we require $\boldsymbol{e}(\phi_\circ' \rtimes (w_1', w_2'))$ to be nonzero, then all the input-output wire pairs in w_1', w_2' match, and somewhere between zero and two input-output pairs of wires in v_1', v_2' have an input wire reading TRUE and an output wire reading FALSE. We refer to such a pair as a (TRUE, FALSE) pair and note that the number of (TRUE, FALSE) pairs is equal to $|v_1'| - |v_2'|$.

Of course, whenever $|v_1'| - |v_2'| > 0$, the output wires do not correspond correctly to the input wires, so we need the count

$$\boldsymbol{e}\big((\phi \wedge \phi') \rtimes (v_1 \wedge v_1', v_2 \wedge v_2')\big) \equiv 0 \mod p$$

for this choice of $(v_1 \wedge v_1', v_2 \wedge v_2')$. We will now use (5.1) and (5.2) to do a computation which will produce the remaining equations given in the statement of this lemma.

Note that, because ϕ' is made up of IDENTITY and SWAP gates, the technical restrictions in §4.6 are enough to ensure that $\boldsymbol{e}(\phi_\circ' \rtimes (w_1', w_2')) \equiv 1 \mod p$ whenever $\boldsymbol{e}(\phi_\circ' \rtimes (w_1', w_2'))$ is nonzero modulo p.

Case 1: Zero (TRUE, FALSE) pairs in (v_1', v_2'). Then $\boldsymbol{e}(\phi_\circ' \rtimes (w_1', w_2'))$ is nonzero modulo p precisely when $w_1' = v_1'$ and $w_2' = v_2'$, so that (5.2) gives $g(0, 0) \equiv 1 \mod p$ and $g(h, j) \equiv$

$0 \mod p$ otherwise. Then (5.1) becomes:

$$\boldsymbol{e}\big((\phi \wedge \phi') \rtimes (v_1 \wedge v_1', v_2 \wedge v_2')\big) \equiv \boldsymbol{e}\big(\phi \rtimes (v_1, v_2)\big) \mod p.$$

So that $\boldsymbol{e}(\phi \wedge \phi')$ behaves like the Bruhat logic gate ϕ on the top two wires.

Case 2: One (TRUE, FALSE) pair in (v_1', v_2'). Then for $\ell = 1$, $r = 0$, there is exactly one choice of w_1', w_2' with $\boldsymbol{e}(\phi_\circ' \rtimes (w_1', w_2'))$ nonzero modulo p; this choice corresponds to switching the TRUE input wire in the (TRUE, FALSE) pair to FALSE. Likewise, for $\ell = 0$, $r = 1$, there is exactly one choice. Thus (5.2) gives $g(1,0) \equiv g(0,1) \equiv 1 \mod p$.

Now taking $\ell = r = 0$, we have $(w_1', w_2') = (v_1', v_2')$, and $\boldsymbol{e}(\phi_\circ' \rtimes (w_1', w_2')) \equiv 0 \mod p$ by Lemma 4.1. Then (5.2) gives $g(0,0) \equiv -2 \mod p$. There are four possible choices of v_1 and v_2 satisfying $|v_2| - |v_1| = 1$, which give (2), (3), (4), and (5).

Case 3: Two (TRUE, FALSE) pairs in (v_1', v_2'). We proceed with a calculation similar to the one above. For $\ell = 2$, $r = 0$ and $\ell = 0$, $r = 2$, there is exactly one choice of w_1', w_2', while for $\ell = 1$, $r = 1$, there are two choices. Then (5.2) gives:

$$g(2,0) \equiv g(1,1) \equiv g(0,2) \equiv 2 \mod p.$$

For $\ell + r = 1$ and $\ell + r = 0$, Lemma 4.1 tells us $\boldsymbol{e}(\phi_\circ' \rtimes (w_1', w_2')) \equiv 0 \mod p$, and we compute:

$$g(1,0) \equiv g(0,1) \equiv -4 \mod p \quad \text{and} \quad g(0,0) \equiv 2 \mod p.$$

There is only one choice of v_1 and v_2 satisfying $|v_2| - |v_1| = 2$, which gives (6).

This completes the proof when ϕ is the SWAP, ANDOR or TESTEQ gate. When ϕ is the IDENTITY gate, we follow the same argument and find that we must only consider the cases when $0 \le |v_2| - |v_1| \le 1$. Working through Case 1 and Case 2 above gives (8). □

5.3. **Proof of Lemma 4.6.** We describe a function that sends a permutation τ with $\tau \le \boxed{\phi}(x_i, z_i)$ to a permutation $\tau^* \le \phi$. Note that since the blocks are in increasing order in $\boxed{\phi}(x_i, z_i)$, they will still be in increasing order in τ. The only elements that can lie "within" one of these blocks (where "within" means to the right of some element from the block and to the left of another element of the block) are elements that were originally larger than the block and to its left, or smaller than the block and to its right.

To produce τ^*, push the elements of τ that have moved within blocks out of their blocks, either to the left or right, back to the side they came from. We treat the penultimate block the same way. Then replace the blocks with the old x_i's and shift everything back down.

Since ϕ has finite length, there are only finitely many choices of τ^*. For each τ^* we count the number of possible $\tau \le \boxed{\phi}(x_i, z_i)$ in the pre-image of the function described above. To do this computation, we consider the number of elements in τ immediately to the left of an x_i and larger than it, or immediately to the right of an x_i and smaller than it. Call the first number ℓ and the second r. Then we are counting rearrangements of the block sequence

$$\boxed{3}\,\boxed{2}\,\boxed{1}$$

with blocks of lengths ℓ, z_i and r, respectively, such that none of the elements from the $\boxed{3}$ or $\boxed{1}$ block cross the entire $\boxed{2}$ block. We sum over the number $h \le \ell$ of elements that move from the $\boxed{3}$ block into the $\boxed{2}$ block, and the number $j \le r$ of elements that move from

the $\boxed{1}$ block into the $\boxed{2}$ block:

$$\sum_{0\le h\le \ell}\sum_{0\le j\le r}\binom{z_i+h+j-2}{h}\binom{z_i+j-2}{j}.$$

We thus obtain:

$$\boldsymbol{e}\left(\boxed{\phi}(x_i,z_i)\right)=\sum_{\tau^*\le\phi}\prod_{i=1}^{t+1}\sum_{0\le h\le \ell}\sum_{0\le j\le r}\binom{z_i+h+j-2}{h}\binom{z_i+j-2}{j},$$

which is a polynomial in the z_i's and p^3-1, as desired. □

6. Height two posets

Let $\mathcal{P}=(X,<)$ be a poset on a set X of n elements $\{x_1,\dots,x_n\}$. Denote by $\Gamma=(X,E)$ its comparability graph, with oriented edges $(x_i,x_j)\in E$ if $x_i<x_j$ in $\mathcal{P}$. Denote by X' a identical copy of X with elements $\{x'_1,\dots,x'_n\}$.

Define the poset $\mathcal{Q}=(X\cup X',\prec)$ on $2n$ elements, by having $x_i\prec x'_i$ for all $x_i\in X$, and $x_i\prec x'_j$ for all $x_i<x_j$, with $x_i,x_j\in X$. In particular, the Hasse diagram of $\mathcal{Q}$ consists of $n+|E|$ edges. Note that $\mathcal{Q}$ is a poset of height 2, see Figure 3.

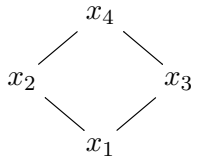

FIGURE 2. The Hasse diagram of a poset $\mathcal{P}$.

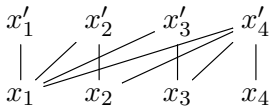

FIGURE 3. Poset $\mathcal{Q}$ associated to poset $\mathcal{P}$.

For every prime p between n and n^2, we construct the modified poset $\mathcal{Q}_p$ by adding, for all i and j satisfying $1\le i\le n$ and $1\le j\le p-2$, the element x_{ij} and the relation $x_{ij}\prec x'_i$. Note that $\mathcal{Q}_p$ is still of height 2 and has pn elements (see Figure 4).

We will use the number of linear extensions of $\mathcal{Q}$ and $\mathcal{Q}_p$ to compute the number of linear extensions of $\mathcal{P}$. Consider first the number $\boldsymbol{e}(\mathcal{Q})$ of linear extensions of $\mathcal{Q}$. Let $A\in\binom{[2n]}{n}$, i.e. A is a n-subset of $[2n]=\{1,2,\dots,2n\}$. Denote by $\boldsymbol{e}_A(\mathcal{Q})$ be the number of linear extensions of $\mathcal{Q}$ such that the values assigned to X' in the linear extension are the elements of A. Then

$$\boldsymbol{e}(\mathcal{Q})=\sum_{A\in\binom{[2n]}{n}}\boldsymbol{e}_A(\mathcal{Q}).$$

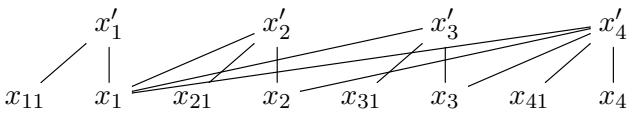

FIGURE 4. $\mathcal{Q}_p$ for $p=3$.

Lemma 6.1. $\boldsymbol{e}(\mathcal{P}) = \boldsymbol{e}_{\{2,4,6,\ldots,2n\}}(\mathcal{Q})$.

Proof. Since $2, 4, 6, \ldots$ are assigned to X', we must have $1, 3, 5, \ldots$ assigned to X. An easy induction argument shows that if $2k$ is assigned to x_i', then the element $2k-1$ must be assigned to x_i, for all $1 \le k \le n$. The additional relations on $\mathcal{Q}$ ensure that this is a linear extension of $\mathcal{Q}$ if and only if the corresponding assignment of values to X is a linear extension of $\mathcal{P}$. □

The above lemma should be compared with the following result:

Lemma 6.2. $\boldsymbol{e}(\mathcal{Q}_p) \equiv (-1)^n \boldsymbol{e}_{\{2,4,6,\ldots,2n\}}(\mathcal{Q}) \mod p$.

Proof. Throughout the proof of this lemma, we will consider colorings of a set of integers. A *coloring* is be a function from that set to some list of acceptable colors.

Let $A \in \binom{[2n]}{n}$, and write $A = \{a_1, \ldots, a_n\}$, with $a_1 < a_2 < \cdots < a_n$. A coloring of the set $[pn] = \{1, 2, \ldots, pn\}$ is called *A-compatible* if the following conditions are satisfied:

(1) there is a sequence of $2n$ integers $b_1 < \cdots < b_{2n}$ colored black,
(2) there are another n colors $C_1, \ldots, C_n$, and $p-2$ integers are colored with each of these colors,
(3) all of the elements colored with C_k lie before b_{a_k}.

Let $f_p(A)$ be the number of A-compatible colorings of $[pn]$. We observe that, given a linear extension of $\mathcal{Q}$ where the values assigned to X' belong to the set A, the number of linear extensions of $\mathcal{Q}_p$ that preserve the ordering on $X \cup X'$ is

$$f_p(A)\big((p-2)!\big)^n.$$

The b_k's represent the values assigned to $X \cup X'$ in the linear extension of $\mathcal{Q}_p$. Let x_i' be the element assigned the value a_k in the given linear extension of $\mathcal{Q}$. Then x_i' will be assigned b_{a_k} in the linear extension of $\mathcal{Q}_p$, and the collection of elements colored with C_k represents the values assigned to the elements x_{ij} attached to x_i'. There are $(p-2)!$ ways to assign these values, for each k, with $1 \le k \le n$, giving the formula above.

We then have, by Wilson's theorem:

$$\boldsymbol{e}(\mathcal{Q}_p) \,=\, ((p-2)!)^n \sum_{A \in \binom{[2n]}{n}} \boldsymbol{e}_A(\mathcal{Q})\, f_p(A) \,\equiv \sum_{A \in \binom{[2n]}{n}} \boldsymbol{e}_A(\mathcal{Q})\, f_p(A) \mod p.$$

In an A-compatible coloring of $\{1, 2, \ldots, pn\}$, there are $a_k - 1 + k(p-2)$ terms to the left of b_{a_k} colored either black or one of the colors $C_1, \ldots, C_k$. Among these terms, we can choose the position of the elements colored C_k arbitrarily. This gives

$$f_p(A) \,=\, \prod_{k=1}^{n} \binom{a_k - 1 + k(p-2)}{p-2}.$$

For $A = \{2, 4, 6, \ldots, 2n\}$, we have $a_k = 2k$, so this becomes

$$f_p\left(\{2, 4, 6, \ldots, 2n\}\right) \,=\, \prod_{k=1}^{n} \binom{kp-1}{p-2} \,\equiv\, (-1)^n \mod p.$$

For every other A with $\boldsymbol{e}_A(\mathcal{Q}) \neq 0$, we have $f_p(A) \equiv 0 \mod p$. Indeed, we must have $a_n = 2n$, since $2n \notin X$. We proceed by induction on $n - k$. Suppose that

$$(a_{k+1}, \ldots, a_n) \,=\, (2k+2, \ldots, 2n).$$

Then the relations $x_i \prec x_i'$ force a_k to be equal to either $2k$ or $2k+1$. If $a_k = 2k+1$, then

$$\binom{a_k - 1 + kp - 2k}{p-2} \;=\; \binom{kp}{p-2}$$

will divide $f_p(A)$. Since $\binom{kp}{p-2} \equiv 0 \mod p$, we have $f_p(A) \equiv 0 \mod p$ unless $a_k = 2k$. □

Proof of Theorem 1.1. Using the same Chinese Remainder Theorem argument we used in §3.3, the lemmas above show that computing $\boldsymbol{e}(\mathcal{Q}_p)$ for the primes between n and n^2 is sufficient to determine $\boldsymbol{e}(\mathcal{P})$. Since #LE is #P-complete, so is #H2LE. □

7. Incidence posets

7.1. **Counting incidence posets.** Given a graph $G = (V, E)$, we construct its incidence poset I_G, with elements corresponding to vertices *and* edges of G, with $x < y$ in $\mathcal{P}$ if and only if $x \in E$, $y \in V$ and y is an endpoint of x. We write $\boldsymbol{e}(G)$ for the number of linear extensions of I_G.

Our approach here is similar to our approach in Section 6. We produce, given a poset $\mathcal{P}$ and a prime $p > |\mathcal{P}|$, a graph $G_p(\mathcal{P})$ with:

$$\boldsymbol{e}\left(G_p(\mathcal{P})\right) \;\equiv\; (-1)^{|\mathcal{P}|} \cdot 8\, \boldsymbol{e}(\mathcal{P}) \mod p.$$

Let $G = (V, E)$ be a graph, with $V = \{x_1, \ldots, x_n\}$, and $\sigma \in S_n$ a permutation. Denote by $\boldsymbol{e}_\sigma(G)$ the number of linear extensions of I_G, which satisfy the following condition: when restricted to V, induce the permutation σ, so that $x_{\sigma^{-1}(1)} \le x_{\sigma^{-1}(2)} \le \cdots \le x_{\sigma^{-1}(n)}$. We have:

$$\boldsymbol{e}(G) \;=\; \sum_{\sigma \in S_n} \boldsymbol{e}_\sigma(G).$$

Informally, to compute $\boldsymbol{e}_\sigma(G)$ we visit the vertices of G in the order dictated by σ, accounting for the new edges we meet at each step.

Formally, given a permutation $\sigma \in S_n$, we produce the sequence $\{t_1, \ldots, t_n\}$, where t_i is the number of edges in E with $x_{\sigma^{-1}(i)}$ as an endpoint, and no endpoint $x_{\sigma^{-1}(j)}$ for $j < i$. Let $\{u_1, \ldots, u_n\}$ be the sequence of partial sums of the t_i's, so that

$$u_k \;=\; t_1 + \ldots + t_k.$$

Note that u_k is the total number of edges incident to the set of vertices $x_{\sigma^{-1}(1)}, \ldots, x_{\sigma^{-1}(k)}$.

Let $|E| = m$. Then we call a coloring of the set $\{1, 2, \ldots, m+n\}$ (G, σ)-compatible if the following conditions are satisfied:

(1) there is a sequence of n integers $b_1 < \cdots < b_n$ colored black,
(2) there are another n colors $C_1, \ldots, C_n$, and t_k integers are colored with the color C_k,
(3) all of the elements colored with C_k lie before b_k.

Let $f(G, \sigma)$ be the number of (G, σ)-compatible colorings. In such a coloring, there are $u_k + k - 1$ numbers to the left of b_k colored either black or one of the colors $C_1, \ldots, C_k$. Among these terms, we can choose the position of the elements colored C_k arbitrarily. This gives:

$$f(G, \sigma) \;=\; \prod_{k=1}^{n} \binom{u_k + k - 1}{t_k}.$$

A (G, σ)-compatible coloring corresponds to a collection of linear extensions of I_G counted by $\boldsymbol{e}_\sigma(G)$. The values assigned to the t_k new edges at $x_{\sigma^{-1}(k)}$ are given by the numbers colored with C_k, and these values can be assigned in $(t_k)!$ ways, so that we have:

$$\boldsymbol{e}(G) \,=\, \sum_{\sigma \in S_n} f(G,\sigma) \prod_{k=1}^{n} (t_k)! \,=\, \sum_{\sigma \in S_n} \prod_{k=1}^{n} (t_k)! \binom{u_k + k - 1}{t_k}. \tag{7.1}$$

In particular, when we are counting modulo p we can restrict our attention to permutations σ, which have corresponding sequences $\{t_1, \ldots, t_n\}$ with $t_i < p$ for all i. Informally, we want to visit each vertex of G in the order given by σ, deleting the edges incident to each vertex after we visit it, and ensure that no vertex has at least p edges by the time we visit it.

Now we give the actual construction of $G_p(\mathcal{P})$. The first step is to construct a gadget J_p, which is a graph defined as follows. Start with the complete bipartite graph $K_{p-1,p-1}$ on $2p-2$ vertices. Call these vertices $y_1, \ldots, y_{p-1}$ and $z_1, \ldots, z_{p-1}$ and add an additional $p-2$ edges from z_{p-1} to z_i for $1 \le i < p-1$. Note that each of the y_i's has degree $p-1$ and the z_i's have degree $\ge p$ (see Figure 5). We need the following:

Lemma 7.1. $\boldsymbol{e}(J_p) \equiv -8 \mod p$.

We defer the proof of this lemma to the end of this section.

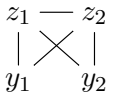

FIGURE 5. J_p for $p = 3$.

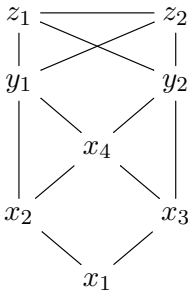

FIGURE 6. $G_p(\mathcal{P})$ for $\mathcal{P}$ as in Figure 2 and $p = 3$.

To construct $G_p(\mathcal{P})$, add below J_p the Hasse diagram of $\mathcal{P}$ (treated as an undirected graph). For each element $x \in \mathcal{P}$, let v_x be the number of elements in $\mathcal{P}$ that cover x. Add $p - 1 - v_x$ edges from x to the degree $p-1$ vertices y_i of J_p in an arbitrarily way (see Figure 6).

Theorem 1.2 follows immediately from the following:

Lemma 7.2. $\boldsymbol{e}(G_p(\mathcal{P})) \,\equiv\, (-1)^{|\mathcal{P}|+1} \cdot 8\, \boldsymbol{e}(\mathcal{P}) \mod p$

Proof. Every maximal element of $\mathcal{P}$ has $v_x = 0$, and so is connected to each of the y_i's in J_p. Since $\mathcal{P}$ has at least one maximal element, every element of J_p has degree $\ge p$. Thus every σ which visits a vertex in J_p before visiting every maximal element of $\mathcal{P}$ has a term $t_i \ge p$, so that $\boldsymbol{e}_\sigma(G_p(\mathcal{P})) \equiv 0 \mod p$. Likewise, of these permutations, every permutation σ that visits an element of $\mathcal{P}$ before visiting all of its immediate predecessors has $\boldsymbol{e}_\sigma(G_p(\mathcal{P})) \equiv 0 \mod p$.

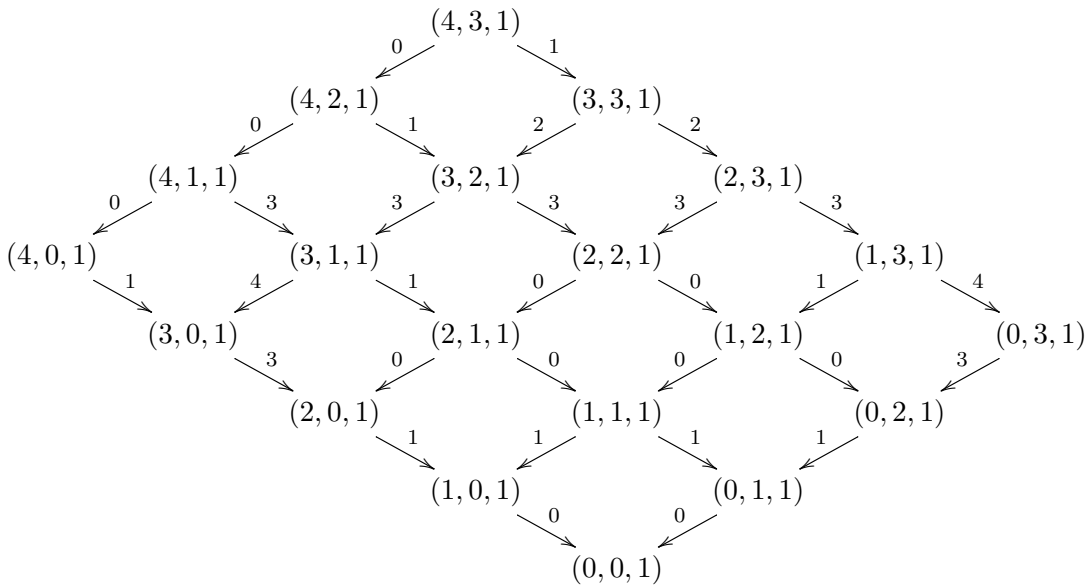

FIGURE 7. The $c = 1$ half of the directed graph $\mathcal{G}'$, with weights, for $p = 5$.

Thus we can restrict our count of $\boldsymbol{e}(G_p(\mathcal{P}))$ modulo p to permutations that have as their first n terms a linear extension of $\mathcal{P}$. For these permutations, we have $t_1 = t_2 = \ldots = t_n = p - 1$, so that $(t_k)! \equiv -1 \mod p$ by Wilson's theorem, and

$$\binom{u_k + k - 1}{t_k} = \binom{kp - 1}{p - 1} \equiv 1 \mod p.$$

Furthermore, for every $k > n$, we have $t_1 + \ldots + t_k = np - n + (t_{n+1} + \cdots + t_k) + k - 1$, so that

$$\binom{u_k + k - 1}{t_k} \equiv \binom{u_k - u_n + (k - n) - 1}{t_k} \mod p.$$

Now comparing the expressions for $\boldsymbol{e}(G_p(\mathcal{P}))$ and $\boldsymbol{e}(J_p)$ given by (7.1), we have

$$\boldsymbol{e}\left(G_p(\mathcal{P})\right) \equiv (-1)^{|\mathcal{P}|}\boldsymbol{e}(\mathcal{P})\boldsymbol{e}(J_p) \mod p,$$

and Lemma 7.1 completes the proof. □

Proof of Theorem 1.2. Using the same Chinese Remainder Theorem argument we used in §3.3 and Section 6, the two lemmas above show that computing $\boldsymbol{e}(G_p(P))$ for the primes between $|\mathcal{P}|$ and $|\mathcal{P}|^2$ is sufficient to determine $\boldsymbol{e}(\mathcal{P})$. Since #LE is #P-Complete, so is #IPLE. □

7.2. **Proof of Lemma 7.1.** Note that the values t_k and $u_k + k - 1$ in (7.1) are both independent of the order in which the previous $k - 1$ vertices are visited. They can be computed solely by identifying the vertex $x_{\sigma^{-1}(k)}$ and the collection of vertices $\{x_{\sigma^{-1}(i)}\}_{i<k}$. This motivates the following construction. Recall that the induced subgraphs of a graph G are those formed by deleting some vertices together with all incident edges. Take a directed graph $\mathcal{G}$ whose vertices are the induced subgraphs of J_p and whose edges point from each

subgraph to those obtained from it by deleting a single vertex. Attach to each edge the weight

$$(t_k)!\binom{u_k+k-1}{t_k} \;=\; (t_k)!\binom{u_k+k-1}{u_k-u_{k-1}}. \tag{7.2}$$

Then $\boldsymbol{e}(J_p)$ is equal to the sum of all weighted paths in $\mathcal{G}$ from J_p to the empty subgraph.

Let $J_p(a,b,c)$ be an induced subgraph of J_p with a of the y_i's, b of the z_i's, for $1 \le i < p-1$, and $c = 1$ if $z_{p-1} \in J_p(a,b,c)$, $c = 0$ otherwise, for $0 \le a \le p-1$ and $0 \le b \le p-2$. Since the y_i's, and the z_i's, except for z_{p-1}, are indistinguishable, these subgraphs $J_p(a,b,c)$ are *all* of the induced subgraphs of J_p, up to isomorphism.

We can thus reduce our graph of subgraphs $\mathcal{G}$ to the graph $\mathcal{G}'$ containing only these $2p^2-2p$ vertices. We re-weight the edges from $J_p(a,b,c)$ where a, b or c is reduced by one, by multiplying by a, b or c, respectively. This accounts for the a, b or c choices of vertex to remove. Write $\ell(a,b,c)$ for the value of $u_{k-1}+k-1$ upon reaching $J_p(a,b,c)$, that is, $\ell(a,b,c)$ is the number of vertices and edges that must be deleted from $J_p(p-1,p-2,1)$ to give $J_p(a,b,c)$. Then (7.2) gives the weight of the edge from $J_p(a,b,c)$ to $J_p(a-1,b,c)$ in terms of a,b,c and ℓ:

$$a\,(b+c)!\binom{\ell(a,b,c)+b+c}{b+c} \;=\; a\,(b+c)!\binom{\ell(a-1,b,c)-1}{\ell(a-1,b,c)-\ell(a,b,c)-1}. \tag{7.3}$$

The equations for the edges from $J_p(a,b,c)$ to $J_p(a,b-1,c)$ and $J_p(a,b,c-1)$ are the same up to a cyclic permutation of (a,b,c). The total number of edges in J_p is $(p-1)^2+(p-2) = p^2-p-1$. The number of edges in $J_p(a,b,c)$ is $ab+ac+bc$, and we reach $J_p(a,b,c)$ by deleting $(p-1-a)+(p-2-b)+(1-c)$ vertices. We then calculate:

$$\begin{aligned}\ell(a,b,c) \;&=\; p^2-p-1-(ab+(a+b)c)+(p-1-a)+(p-2-b)+(1-c)\\ &\equiv\; (a+2)(p-b-2)+(c-1)(a+b+2) \mod p.\end{aligned}$$

Lemma 7.3. *When $c=1$, $(a+2)(p-b-2) > p$ and $(p-a-2)(b+2) > p$, every path in $\mathcal{G}'$ that visits $J_p(a,b,c)$ has weight zero modulo p.*

Proof. We argue by induction on $(2p-3)-(a+b)$, that is, on the distance in $\mathcal{G}'$ from $J_p(p-1,p-2,1)$ to $J_p(a,b,c)$. When $a=p-1$, $b=p-2$, $c=1$, the conditions of the lemma are not met, and the statement is true vacuously.

Now suppose that a,b,c satisfy the conditions in this lemma. Then a path that visits $J_p(a,b,c)$ must come from either $J_p(a+1,b,c)$ or $J_p(a,b+1,c)$. If the values $a+1,b,c$ satisfy the conditions in this lemma, we can then apply the induction hypothesis to show that every path through $J_p(a+1,b,c)$ has weight 0 modulo p. In particular, a path that includes the edge from $J_p(a+1,b,c)$ to $J_p(a,b,c)$ has weight 0 modulo p.

On the other hand, suppose that $a+1,b,c$ do not satisfy the conditions in this lemma. Then $(a+3)(p-b-2) > (a+2)(p-b-2) > p$, so we must have $(p-a-3)(b+2) \le p$. Note that if a or b is greater than or equal to $p-2$, either $(a+2)(p-b-2) \le 0$ or $(p-a-2)(b+2) \le 0$. We thus have $a,b < p-2$, so that $(p-a-3)(b+2) = p$ is impossible.

However, when $(p-a-3)(b+2) < p$, since $b < p-2$, we have $(p-a-3)(b-2) >_p (p-a-2)(b+2)$. Thus, $\ell(a+1,b,c) >_p \ell(a,b,c)$, and so by (7.3), the edge from $J_p(a+1,b,c)$ to $J_p(a,b,c)$ has weight 0 modulo p. The argument for the edge from $J_p(a,b+1,c)$ to $J_p(a,b,c)$ is the same by symmetry. □

Lemma 7.4. *Given a,b with $(b+2)(p-a-2) \le p$ the edge from $J_p(a,b,1)$ to $J_p(a,b,0)$ has weight 0 unless $a=p-3$ and $b=0$, $a=p-2$ and $b=0$ or 1, or $a=p-1$ with b arbitrary.*

Similarly, given a, b with $(a+2)(p-b-2) \le p$, the edge from $J_p(a,b,1)$ to $J_p(a,b,0)$ has weight 0 unless $b = p-3$ and $a = 0$, or $b = p-2$ and $a = 0$ or 1.

Proof. We give the proof of the first statement, since the proof of the second is essentially identical. Permuting (a,b,c) in (7.3) to find the weight of the edge from $J_p(a,b,1)$ to $J_p(a,b,0)$, we note that we must have $a+b < p$ and $a+b <_p a+b+\ell(a,b,1)$. Since $\ell(a,b,1) \equiv (b+2)(p-a-2) \mod p$, this gives:

$$a+b+(b+2)(p-a-2) \,<\, p.$$

This implies that

$$p < a+1+\tfrac{3}{b+1} \le a+4.$$

We conclude that $a > p-4$, and the rest of the lemma follows by elementary case analysis. □

Proof of Lemma 7.1. Note that the edges from $J_p(p-1,p-2,1)$ to $J_p(p-1,p-3,1)$ and $J_p(p-1,p-2,0)$ have weight 0 modulo p. Combining this with the previous two lemmas, we conclude that every path in $\mathcal{G}'$ has weight 0 modulo p unless it visits either $J_p(p-2,1,1)$ or $J_p(1,p-2,1)$. We now complete the desired calculation, through repeated applications of (7.3), symmetry, and Wilson's theorem:

$$\begin{aligned}
\boldsymbol{e}(J_p(p-1,p-2,1)) \;&\equiv\; (p-1)(p-1)!\,\boldsymbol{e}\big(J_p(p-2,p-2,1)\big)\\
&\equiv\; (p-2)!\,(-1)^{p-3}\Big[\boldsymbol{e}(J_p(p-2,1,1)) + \boldsymbol{e}\big(J_p(p-2,0,1)\big)\Big]\\
&\equiv\; 2\,\boldsymbol{e}\big(J_p(p-2,1,1)\big)\\
&\equiv\; 2\,(p-1)!\Big[\boldsymbol{e}\big(J_p(p-2,1,0)\big) + \boldsymbol{e}\big(J_p(p-2,0,1)\big)\Big]\\
&\equiv\; -4\,\boldsymbol{e}\big(J_p(p-2,0,1)\big)\\
&\equiv\; -4\,(p-2)!\,\boldsymbol{e}\big(J_p(p-2,0,0)\big) \;-\; 4\,(p-2)\,\boldsymbol{e}\big(J_p(p-3,0,1)\big)\\
&\equiv\; -4\,\boldsymbol{e}\big(J_p(p-2,0,0)\big) \;+\; 8\binom{p-1}{2}\boldsymbol{e}\big(J_p(p-3,0,0)\big)\\
&\equiv\; -4\,(p-2)! \;+\; 4\,(p-1)(p-2)(p-3)!\\
&\equiv\; -8 \mod p.
\end{aligned}$$

This completes the proof. □

8. Final remarks

8.1. Lee and Skipper [LS17] report:

> [In personal communication] *"Brightwell and Winkler asserted that: (i) the complexity for the general height-2 case is still open; (ii) there seems to be no work on counting linear extensions of incidence posets; (iii) there is no compelling reason to believe that the case of incidence posets should be easier than general height-2 posets."*

Now that we proved that both results are #P-complete, this finally settles the debate. Arguably, our proof of Theorem 1.1 could have been obtained 27 years ago when [BW91] appeared. On the other hand, our proof of Theorem 1.3 was only made possible with advances in computer algebra and computer technology.

8.2. The equations in Appendix A are nonhomogeneous polynomials in 5 variables, with a maximum total degree of 5. The coefficients are nonnegative integers ≤ 400. Before inserting parameters, the gates were permutations of length 8, so there were $8! = 40320$ possibilities. In fact, the requirement that the variables be in strictly decreasing order restricts the possibilities significantly. After some experimentation, we added the further restriction that the first variable be in the first position of the permutation. After these restrictions, only 96 possibilities remain.

For each gate, we generated the system of 12 polynomials in `C++`, for each of these 96 possible permutations. We then computed which systems had solutions over $\mathbb{C}$; the systems were tested with `Macaulay2`.[1] Generating the systems took 314.4 seconds, or an average of 3.3 seconds per system. Testing all 96 systems took less than ten seconds for each of the three gates. If we had needed to extend our search to 6 variables, the cost in computing time would have increased significantly, as shown in Figure 8.

Here is the result of our computation. For each gate, at least one of the of the 96 possible permutations produced systems of equations with nontrivial solutions over $\mathbb{C}$. To be precise:

- ◇ For the Swap gate, this worked for 2 of the 96 possible permutations.
- ◇ For the AndOr gate, this worked for 47 of the 96 possible permutations.
- ◇ For the TestEq gate, this worked for 4 of the 96 possible permutations.

Variables	Candidate gates	Computation time per candidate gate (sec.)
4	6	≤ 0.1
5	96	3.3
6	1200	1618 ($\sim$ 27 minutes)

Figure 8. Candidate permutations and computation time.

8.3. Let us quickly mention complexity implications of our results for people unfamiliar with modern Complexity Theory. Roughly, when a counting problem is #P-complete, this is an extremely strong evidence against it being computable in polynomial time, much stronger than $\mathsf{P} \neq \mathsf{NP}$, for example. Indeed, otherwise Toda's theorem $\mathsf{PH} \subseteq \mathsf{P}^{\#\mathsf{P}}$ implies that every problem in *polynomial hierarchy* PH can be solved in polynomial time.

Another interesting question is about complexity of computing #D2LE mod p. Note that #P-completeness does not automatically imply the hardness of all such problems, e.g. $\mathrm{Per}(A) \bmod 2$ of an integer matrix A can be computed in polynomial time. While our proof works implies hardness only for primes ≥ 11, an early version of the proof works modulo 2, i.e. proves that #D2LE mod 2 is $\oplus$P-complete, see [Dit18+]. In fact, we are confident that a larger version of our construction would prove the result for the remaining primes $p \in \{3, 5, 7\}$.

8.4. Motivated by probabilistic applications, Montúfar and Rauh [MR16] recently define the *polytope of modes* $\mathbf{M}(G, X)$, for every simple graph $G = (V, E)$ and independent subset of vertices $X \subset V$. They prove that

$$\operatorname{vol} \mathbf{M}(G, X) \,=\, \frac{\operatorname{vol}(\Delta^n)}{n!}\, \boldsymbol{e}\big(P_{G,X}\big),$$

[1] Computations were made with an `Intel`® `Core`™ `i7-3610QM CPU with 2.30GHz, 4 cores and 8Gb of RAM`.

where $n = |V|$, $\mathrm{vol}(\Delta^n) = \sqrt{n}/(n-1)!$, and $P_{G,X}$ is a height-2 poset with vertices in X on one level and $V \smallsetminus X$ on the other [MR16, Prop. 3] (see also [Sta86] for a strongly related *order polytope*). The authors then discuss the problem of computing $\boldsymbol{e}\big(P_{G,X}\big)$.

The following result follows easily from our Theorem 1.2. Curiously, we learned about this problem after the paper has been written.

Corollary 8.1. *The problem of computing $\boldsymbol{e}\big(P_{G,X}\big)$ is $\#\mathsf{P}$-complete.*

Proof. For a simple graph $H = (V, E)$, let G be the *medial graph* $G = M(H)$, defined as a graph on the set of vertices $V \cup E$ with edges given by adjacencies. This is a bipartite graph, so V is an independent set. By construction, $P_{G,X} = I_G$ is the incidence poset, which implies the result. □

8.5. Let us mention some interesting open problems. First, there is a long tradition in Probabilistic Combinatorics to study properties of *random posets*, see survey [Bri93]. In fact, there are several interesting models for random posets P and for some of them rather sharp results on the number $\boldsymbol{e}(P)$ of their linear extensions (see e.g. [BB97] and references therein). It would be interesting to see if $\boldsymbol{e}(P)$ can be computed in polynomial time w.h.p. We would be especially curious about complexity of computing $\boldsymbol{e}(P)$ for random height-2 posets, and of $\boldsymbol{e}(P_\sigma)$ for random $\sigma \in S_n$.

We are also curious about variations on Theorem 1.4. For example, is computing the size of the principal ideal of the *strong Bruhat order* $\#\mathsf{P}$-complete? What about other finite Coxeter groups? We refer [BjB05] for definitions and the background.

Finally, we conjecture that computing the number $R(\sigma)$ of reduced factorizations of a permutation $\sigma \in S_n$ into adjacent transpositions is $\#\mathsf{P}$-complete. Recall that $R(\sigma)$ can be computed in polynomial time in several special cases, see e.g. [MPP17]. Note that such factorizations can be viewed as saturated chains $1 \to \sigma$ in the weak Bruhat order $B_n = (S_n, \leq)$.

8.6. In his 1984 short survey paper [Riv84], Rival wrote:

> *"Counting is hard! And counting the linear extensions of an ordered set is no exception. This counting problem is tractable only for some special and very simple classes of ordered sets."*

While we now know many more classes of posets for which one can compute the number of linear extensions, the sentiment continues to hold.

Acknowledgements. We are grateful to Greg Kuperberg, Laci M. Lovász, Alejandro Morales, Greta Panova, Bruce Rothschild, Pete Winkler and Damir Yeliussizov for helpful conversations and remarks on the subject. We are thankful to Jon Lee for telling us about incidence posets and bringing [LS17] to our attention. Guido Montúfar kindly showed us [MR16] and explained the problem discussed in §8.4. We are especially grateful to Anton Leykin for his insights into computer algebra and his help programming in `Macaulay2`, and to MSRI for hosting us to make such conversations possible.

Finally, the second author owes a debt of gratitude to Ivan Rival, who went out of his way to meet us during his travels to the Soviet Union, when we were still an undergraduate interested in combinatorics. Ivan encouraged us to work on posets, an advice we didn't adhere until now.

The second author was partially supported by MSRI and the NSF.

References

[Atk89] M. D. Atkinson, The complexity of orders, in *Algorithms and order* (I. Rival, Ed.), Kluwer, Dordrecht, 1989, 195–230.

[BGHP10] J. Banks, S. Garrabrant, M. L. Huber and A. Perizzolo, Using TPA to count linear extensions, `arXiv:1010.4981`.

[BjB05] A. Björner and F. Brenti, *Combinatorics of Coxeter groups*, Springer, New York, 2005.

[BjW91] A. Björner and M. Wachs, Permutation statistics and linear extensions of posets, *J. Combin. Theory* **A 58** (1991), 85–114.

[BB97] B. Bollobás and G. Brightwell, The structure of random graph orders, *SIAM J. Discrete Math.* **10** (1997), 318–335.

[BBS99] B. Bollobás, G. Brightwell and A. Sidorenko, Geometrical techniques for estimating numbers of linear extensions, *European J. Combin.* **20** (1999), 329–335.

[Bri93] G. Brightwell, Models of random partial orders, in *Surveys in combinatorics*, Cambridge Univ. Press, Cambridge, 1993, 53–83.

[BW91] G. Brightwell and P. Winkler, Counting linear extensions, *Order* **8** (1991), 225–247; extended abstract in *Proc. 23rd STOC* (1991), 175–181.

[BD99] R. Bubley and M. Dyer, Faster random generation of linear extensions, *Discrete Math.* **201** (1999), 81–88; extended abstract in *Proc. 9th SODA* (1998), 350–354.

[BDGJ99] R. Bubley and M. Dyer, C. Greenhill and M. Jerrum, On approximately counting colorings of small degree graphs, *SIAM J. Comput.* **29** (1999), 387–400; extended abstract in *Proc. 9th SODA* (1998), 355–363.

[CH93] N. Creignou and M. Hermann, On #P-completeness of some counting problems, Research Report **2144** (1993), INRIA, 11 pp.

[CRS09] S. Caracciolo, E. Rinaldi and A. Sportiello, Exact sampling of corrugated surfaces, *J. Stat. Mech. Theory. Exp.* (2009), P02049, 13 pp.

[Dit18+] S. Dittmer, UCLA Ph.D. thesis in prepation.

[EGKO16] E. Eiben, R. Ganian, K. Kangas and S. Ordyniak, Counting linear extensions: parametrizations by treewidth, in *Proc. 24th ESA* (2016), Art. 39, 18 pp.

[FM14] S. Felsner and T. Manneville, Linear extensions of N-free orders, *Order* **32** (2014), 147–155.

[FW97] S. Felsner and L. Wernisch, Markov chains for linear extensions, the two-dimensional case, in *Proc. 8th SODA* (1997), 239–247.

[Hub06] M. Huber, Fast perfect sampling from linear extensions, *Discrete Math.* **306** (2006), 420–428.

[Hub14] M. Huber, Near-linear time simulation of linear extensions of a height-2 poset with bounded interaction, *Chicago J. Theoret. Comput. Sci.* (2014), Art. 3, 16 pp.

[KL91] J. Kahn and N. Linial, Balancing extensions via Brunn–Minkowski, *Combinatorica* **11** (1991), 363–368.

[KHNK16] K. Kangas, T. Hankala, T. Niinimäki and M. Koivisto, Counting linear extensions of sparse posets, in *Proc. 25th IJCAI* (2016), 603–609.

[K+17] L. Kari, S. Kopecki, P.-É. Meunier, M. J. Patitz and S. Seki, Binary pattern tile set synthesis is NP-hard, *Algorithmica* **78** (2017), 1–46.

[KK91] A. Karzanov and L. Khachiyan, On the conductance of order Markov chains, *Order* **8** (1991), 7–15.

[Kha93] L. Khachiyan, Complexity of polytope volume computation, in *New trends in discrete and computational geometry*, Springer, Berlin, 1993, 91–101.

[LS17] J. Lee and D. Skipper, Volume computations for sparse boolean quadric relaxations; `arXiv:1703.02444`.

[Mac01] D. MacKenzie, *Mechanizing proof. Computing, risk, and trust*, MIT Press, Cambridge, MA, 2001.

[Mat91] P. Matthews, Generating a random linear extension of a partial order, *Ann. Probab.* **19** (1991), 1367–1392.

[MR16] G. Montúfar and J. Rauh, Mode poset probability polytopes, *J. Algebr. Stat.* **7** (2016), 1–13.

[MM11] C. Moore and S. Mertens, *The nature of computation*, Oxford Univ. Press, Oxford, 2011.

[MPP17] A. Morales, I. Pak and G. Panova, Hook formulas for skew shapes III. Multivariate and product formulas, `arXiv:1707.00931`.

[MPP18] A. Morales, I. Pak and G. Panova, Asymptotics of the number of standard Young tableaux of skew shape, *European J. Combin.* **70** (2018), 26–49.

[MR08] W. Mulzer and G. Rote, Minimum-weight triangulations is NP-hard, *J. ACM* **55** (2008), no. 2, Art. 11, 29 pp.; extended abstract in *Proc. 22nd SOCG* (2006), 1–10.

[Möh89] R. H. Möhring, Computationally tractable classes of ordered sets, in *Algorithms and order* (I. Rival, Ed.), Kluwer, Dordrecht, 1989, 105–193.

[Pap94] C. H. Papadimitriou, *Computational Complexity*, Addison-Wesley, Reading, MA, 1994.

[PB83] J. S. Provan and M. O. Ball, The complexity of counting cuts and of computing the probability that a graph is connected, *SIAM J. Comput.* **12** (1983), 777–788.

[PR94] G. Pruesse and F. Ruskey, Generating linear extensions fast, *SIAM J. Comput.* **23** (1994), 373–386.

[Reu96] K. Reuter, Linear extensions of posets as abstract convex sets, *Hamburger Beiträge zur Mathematik* **56** (1996), 9 pp.; available electronically at `https://tinyurl.com/ycnvbcak`

[Riv84] I. Rival, Linear extensions of finite ordered sets, in *Orders: description and roles*, North-Holland, Amsterdam, 1984, 355–370.

[Sta86] R. P. Stanley, Two poset polytopes, *Discrete Comput. Geom.* **1** (1986), 9–23.

[Sta97] R. P. Stanley, *Enumerative Combinatorics*, Vol. 1, Cambridge Univ. Press, Cambridge, MA, 1997.

[TNK17] T. Talvitie, T. Niinimäki and M. Koivisto, The mixing of Markov chains on linear extensions in practice, in *Proc. 26th IJCAI* (2017), 524–530.

[Tro92] W. T. Trotter, *Combinatorics and Partially Ordered Sets: Dimension Theory*, Johns Hopkins Univ. Press, Baltimore, MD, 1992.

[Tro95] W. T. Trotter, Partially ordered sets, in *Handbook of combinatorics*, Vol. 1, Elsevier, Amsterdam, 1995, 433–480.

[TGF92] W. T. Trotter, W. G. Gehrlein and P. C. Fishburn, Balance theorems for height-2 posets, *Order* **9** (1992), 43–53.

[TW14] W. T. Trotter and R. Wang, Incidence posets and cover graphs, *Order* **31** (2014), 279–287.

[Val79] L. G. Valiant, The complexity of enumeration and reliability problems, *SIAM J. Comput.* **8** (1979), 410–421.

[Zwi02] U. Zwick, Computer assisted proof of optimal approximability results, in *Proc. 13th SODA* (2002), 496–505.

Appendix A. Gate equations

We print here the systems of polynomial equations for the parametrized Swap, AndOr, and TestEq gates given in Lemma 4.7. For each of these gates, there are six equations from Lemma 4.3 and six equations from the requirements for the logical operation of the gate itself, for a total of twelve equations.

A.1. **Swap gate.**

(1) $|\phi| - 2 \equiv 0 \mod p^3$:
$z_1 + z_2 + z_3 + z_4 + z_5 + 4 = 0$

(2) $\boldsymbol{e}\big(\phi \rtimes (11,11)\big) \equiv 1 \mod p$:
$2z_2z_5^3+2z_1z_5^3+4z_5^3+6z_2z_4z_5^2+6z_1z_4z_5^2+12z_4z_5^2+3z_2z_3z_5^2+3z_1z_3z_5^2+6z_3z_5^2+3z_2^2z_5^2+3z_1z_2z_5^2+15z_2z_5^2+6z_1z_5^2+15z_5^2+6z_2z_4^2z_5+6z_1z_4^2z_5+12z_4^2z_5+6z_2z_3z_4z_5+6z_1z_3z_4z_5+12z_3z_4z_5+6z_2^2z_4z_5+6z_1z_2z_4z_5+30z_2z_4z_5+12z_1z_4z_5+30z_4z_5+3z_2z_3z_5+3z_1z_3z_5+6z_3z_5+3z_2^2z_5+3z_1z_2z_5+13z_2z_5+4z_1z_5+11z_5 = 6$

(3) $\boldsymbol{e}\big(\phi \rtimes (10,01)\big) \equiv 1 \mod p$:
$2z_5^3 + 6z_4z_5^2 + 3z_3z_5^2 + 3z_2z_5^2 + 12z_5^2 + 6z_4^2z_5 + 6z_3z_4z_5 + 6z_2z_4z_5 + 24z_4z_5 + 9z_3z_5 + 9z_2z_5 + 16z_5 + 6z_4^2 + 6z_3z_4 + 6z_2z_4 + 12z_4 = 6$

(4) $\boldsymbol{e}\big(\phi \rtimes (10,10)\big) \equiv 0 \mod p$:
$2z_5^3 + 6z_4z_5^2 + 3z_3z_5^2 + 3z_2z_5^2 + 12z_5^2 + 6z_4^2z_5 + 6z_3z_4z_5 + 6z_2z_4z_5 + 24z_4z_5 + 9z_3z_5 + 9z_2z_5 + 22z_5 + 6z_4^2 + 6z_3z_4 + 6z_2z_4 + 18z_4 + 6z_3 + 6z_2 + 12 = 0$

(5) $\boldsymbol{e}\big(\phi \rtimes (01,10)\big) \equiv 1 \mod p$:
$2z_2z_5^3 + 2z_1z_5^3 + 2z_5^3 + 6z_2z_4z_5^2 + 6z_1z_4z_5^2 + 6z_4z_5^2 + 3z_2z_3z_5^2 + 3z_1z_3z_5^2 + 3z_3z_5^2 + 3z_2^2z_5^2 + 3z_1z_2z_5^2 + 18z_2z_5^2 + 12z_1z_5^2 + 15z_5^2 + 6z_2z_4^2z_5 + 6z_1z_4^2z_5 + 6z_4^2z_5 + 6z_2z_3z_4z_5 + 6z_1z_3z_4z_5 + 6z_3z_4z_5 + 6z_2^2z_4z_5 + 6z_1z_2z_4z_5 + 36z_2z_4z_5 + 24z_1z_4z_5 + 30z_4z_5 + 9z_2z_3z_5 + 9z_1z_3z_5 + 9z_3z_5 + 9z_2^2z_5 + 9z_1z_2z_5 + 40z_2z_5 + 22z_1z_5 + 31z_5 + 6z_2z_4^2 + 6z_1z_4^2 + 6z_4^2 + 6z_2z_3z_4 + 6z_1z_3z_4 + 6z_3z_4 + 6z_2^2z_4 + 6z_1z_2z_4 + 30z_2z_4 + 18z_1z_4 + 24z_4 + 6z_2z_3 + 6z_1z_3 + 6z_3 + 6z_2^2 + 6z_1z_2 + 24z_2 + 12z_1 + 18 = 6$

(6) $\boldsymbol{e}(\phi \rtimes (01,01)\big) \equiv 0 \mod p$:
$2z_2z_5^3+2z_1z_5^3+2z_5^3+6z_2z_4z_5^2+6z_1z_4z_5^2+6z_4z_5^2+3z_2z_3z_5^2+3z_1z_3z_5^2+3z_3z_5^2+3z_2^2z_5^2+3z_1z_2z_5^2+18z_2z_5^2+12z_1z_5^2+15z_5^2+6z_2z_4^2z_5+6z_1z_4^2z_5+6z_4^2z_5+6z_2z_3z_4z_5+6z_1z_3z_4z_5+6z_3z_4z_5+6z_2^2z_4z_5+6z_1z_2z_4z_5+36z_2z_4z_5+24z_1z_4z_5+30z_4z_5+9z_2z_3z_5+9z_1z_3z_5+9z_3z_5+9z_2^2z_5+9z_1z_2z_5+34z_2z_5+16z_1z_5+25z_5+6z_2z_4^2+6z_1z_4^2+6z_4^2+6z_2z_3z_4+6z_1z_3z_4+6z_3z_4+6z_2^2z_4+6z_1z_2z_4+24z_2z_4+12z_1z_4+18z_4 = 0$

(7) $\boldsymbol{e}(\phi \rtimes (00,00)\big) \equiv 1 \mod p$:
$2z_5^3 + 6z_4z_5^2 + 3z_3z_5^2 + 3z_2z_5^2 + 18z_5^2 + 6z_4^2z_5 + 6z_3z_4z_5 + 6z_2z_4z_5 + 36z_4z_5 + 15z_3z_5 + 15z_2z_5 + 40z_5 + 12z_4^2 + 12z_3z_4 + 12z_2z_4 + 36z_4 + 6z_3 + 6z_2 + 15 = 3$

(8) $-2\boldsymbol{e}\big(M(\phi_\circ) \rtimes (10,11)\big) + \boldsymbol{e}\big(L(\phi_\circ) \rtimes (10,11)\big) + \boldsymbol{e}\big(R(\phi_\circ) \rtimes (10,11)\big) \equiv 0 \mod p$:
$2z_5^4+8z_4z_5^3+5z_3z_5^3+5z_2z_5^3+2z_1z_5^3+12z_5^3+12z_4^2z_5^2+15z_3z_4z_5^2+15z_2z_4z_5^2+6z_1z_4z_5^2+36z_4z_5^2+3z_3^2z_5^2+6z_2z_3z_5^2+3z_1z_3z_5^2+18z_3z_5^2+3z_2^2z_5^2+3z_1z_2z_5^2+18z_2z_5^2+6z_1z_5^2+22z_5^2+6z_4^3z_5+12z_3z_4^2z_5+12z_2z_4^2z_5+6z_1z_4^2z_5+30z_4^2z_5+6z_3^2z_4z_5+12z_2z_3z_4z_5+6z_1z_3z_4z_5+33z_3z_4z_5+6z_2^2z_4z_5+6z_1z_2z_4z_5+33z_2z_4z_5+12z_1z_4z_5+40z_4z_5+3z_3^2z_5+6z_2z_3z_5+3z_1z_3z_5+13z_3z_5+3z_2^2z_5+3z_1z_2z_5+13z_2z_5+4z_1z_5+12z_5 = 0$

(9) $-2\boldsymbol{e}\big(M(\phi_\circ) \rtimes (01,11)\big) + \boldsymbol{e}\big(L(\phi_\circ) \rtimes (01,11)\big) + \boldsymbol{e}\big(R(\phi_\circ) \rtimes (01,11)\big) \equiv 0 \mod p$:
$2z_2z_5^4 + 2z_1z_5^4 + 2z_5^4 + 8z_2z_4z_5^3 + 8z_1z_4z_5^3 + 8z_4z_5^3 + 5z_2z_3z_5^3 + 5z_1z_3z_5^3 + 5z_3z_5^3 + 5z_2^2z_5^3 + 7z_1z_2z_5^3 + 22z_2z_5^3 + 2z_1^2z_5^3 + 16z_1z_5^3 + 17z_5^3 + 12z_2z_4^2z_5^2 + 12z_1z_4^2z_5^2 + 12z_4^2z_5^2 + 15z_2z_3z_4z_5^2 + 15z_1z_3z_4z_5^2 + 15z_3z_4z_5^2 + 15z_2^2z_4z_5^2 + 21z_1z_2z_4z_5^2 + 66z_2z_4z_5^2 + 6z_1^2z_4z_5^2 + 48z_1z_4z_5^2 + 51z_4z_5^2 + 3z_2z_3^2z_5^2 + 3z_1z_3^2z_5^2 + 3z_3^2z_5^2 + 6z_2^2z_3z_5^2 + 9z_1z_2z_3z_5^2 + 30z_2z_3z_5^2 + 3z_1^2z_3z_5^2 + 24z_1z_3z_5^2 + 24z_3z_5^2 + 3z_2^3z_5^2 + 6z_1z_2^2z_5^2 + 27z_2^2z_5^2 + 3z_1^2z_2z_5^2 + 33z_1z_2z_5^2 + 67z_2z_5^2 + 6z_1^2z_5^2 + 37z_1z_5^2 + 43z_5^2 + 6z_2z_4^3z_5 + 6z_1z_4^3z_5 + 6z_4^3z_5 + 12z_2z_3z_4^2z_5 + 12z_1z_3z_4^2z_5 + 12z_3z_4^2z_5 + 12z_2^2z_4^2z_5 + 18z_1z_2z_4^2z_5 + 54z_2z_4^2z_5 + 6z_1^2z_4^2z_5 + 42z_1z_4^2z_5 + 42z_4^2z_5 + 6z_2z_3^2z_4z_5 + 6z_1z_3^2z_4z_5 + 6z_3^2z_4z_5 + 12z_2^2z_3z_4z_5 + 18z_1z_2z_3z_4z_5 + 57z_2z_3z_4z_5 + 6z_1^2z_3z_4z_5 + 45z_1z_3z_4z_5 + 45z_3z_4z_5 + 6z_2^3z_4z_5 + 12z_1z_2^2z_4z_5 + 51z_2^2z_4z_5 + 6z_1^2z_2z_4z_5 + 63z_1z_2z_4z_5 + 124z_2z_4z_5 + 12z_1^2z_4z_5 + 70z_1z_4z_5 + 79z_4z_5 + 3z_2z_3^2z_5 + 3z_1z_3^2z_5 + 3z_3^2z_5 + 6z_2^2z_3z_5 + 9z_1z_2z_3z_5 + 25z_2z_3z_5 + 3z_1^2z_3z_5 + 19z_1z_3z_5 + 19z_3z_5 + 3z_2^3z_5 + 6z_1z_2^2z_5 + 22z_2^2z_5 + 3z_1^2z_2z_5 + 26z_1z_2z_5 + 47z_2z_5 + 4z_1^2z_5 + 23z_1z_5 + 28z_5 = 0$

(10) $-2\boldsymbol{e}\big(M(\phi_\circ) \rtimes (00,01)\big) + \boldsymbol{e}\big(L(\phi_\circ) \rtimes (00,01)\big) + \boldsymbol{e}\big(R(\phi_\circ) \rtimes (00,01)\big) \equiv 0 \mod p$:
$2z_5^4 + 8z_4z_5^3 + 5z_3z_5^3 + 5z_2z_5^3 + 2z_1z_5^3 + 20z_5^3 + 12z_4^2z_5^2 + 15z_3z_4z_5^2 + 15z_2z_4z_5^2 + 6z_1z_4z_5^2 + 60z_4z_5^2 + 3z_3^2z_5^2 + 6z_2z_3z_5^2 + 3z_1z_3z_5^2 + 33z_3z_5^2 + 3z_2^2z_5^2 + 3z_1z_2z_5^2 + 33z_2z_5^2 + 12z_1z_5^2 + 64z_5^2 + 6z_4^3z_5 + 12z_3z_4^2z_5 + 12z_2z_4^2z_5 + 6z_1z_4^2z_5 + 54z_4^2z_5 + 6z_3^2z_4z_5 + 12z_2z_3z_4z_5 + 6z_1z_3z_4z_5 + 63z_3z_4z_5 + 6z_2^2z_4z_5 + 6z_1z_2z_4z_5 +$

$63z_2z_4z_5 + 24z_1z_4z_5 + 124z_4z_5 + 9z_3^2z_5 + 18z_2z_3z_5 + 9z_1z_3z_5 + 52z_3z_5 + 9z_2^2z_5 + 9z_1z_2z_5 + 52z_2z_5 + 16z_1z_5 + 64z_5 + 6z_4^3 + 12z_3z_4^2 + 12z_2z_4^2 + 6z_1z_4^2 + 36z_4^2 + 6z_3^2z_4 + 12z_2z_3z_4 + 6z_1z_3z_4 + 36z_3z_4 + 6z_2^2z_4 + 6z_1z_2z_4 + 36z_2z_4 + 12z_1z_4 + 48z_4 = 0$

(11) $-2\boldsymbol{e}\big(M(\phi_\circ) \rtimes (00,10)\big) + \boldsymbol{e}\big(L(\phi_\circ) \rtimes (00,10)\big) + \boldsymbol{e}\big(R(\phi_\circ) \rtimes (00,10)\big) \equiv 0 \mod p$:
$2z_5^4 + 8z_4z_5^3 + 5z_3z_5^3 + 5z_2z_5^3 + 2z_1z_5^3 + 20z_5^3 + 12z_4^2z_5^2 + 15z_3z_4z_5^2 + 15z_2z_4z_5^2 + 6z_1z_4z_5^2 + 60z_4z_5^2 + 3z_3^2z_5^2 + 6z_2z_3z_5^2 + 3z_1z_3z_5^2 + 33z_3z_5^2 + 3z_2^2z_5^2 + 3z_1z_2z_5^2 + 33z_2z_5^2 + 12z_1z_5^2 + 70z_5^2 + 6z_4^3z_5 + 12z_3z_4^2z_5 + 12z_2z_4^2z_5 + 6z_1z_4^2z_5 + 54z_4^2z_5 + 6z_3^2z_4z_5 + 12z_2z_3z_4z_5 + 6z_1z_3z_4z_5 + 63z_3z_4z_5 + 6z_2^2z_4z_5 + 6z_1z_2z_4z_5 + 63z_2z_4z_5 + 24z_1z_4z_5 + 136z_4z_5 + 9z_3^2z_5 + 18z_2z_3z_5 + 9z_1z_3z_5 + 64z_3z_5 + 9z_2^2z_5 + 9z_1z_2z_5 + 64z_2z_5 + 22z_1z_5 + 100z_5 + 6z_4^3 + 12z_3z_4^2 + 12z_2z_4^2 + 6z_1z_4^2 + 42z_4^2 + 6z_3^2z_4 + 12z_2z_3z_4 + 6z_1z_3z_4 + 48z_3z_4 + 6z_2^2z_4 + 6z_1z_2z_4 + 48z_2z_4 + 18z_1z_4 + 84z_4 + 6z_3^2 + 12z_2z_3 + 6z_1z_3 + 36z_3 + 6z_2^2 + 6z_1z_2 + 36z_2 + 12z_1 + 48 = 0$

(12) $2\boldsymbol{e}\big(M^2(\phi_\circ) \rtimes (00,11)\big) - 4\boldsymbol{e}\big(LM(\phi_\circ) \rtimes (00,11)\big) - 4\boldsymbol{e}\big(RM(\phi_\circ) \rtimes (00,11)\big) +$
$\boldsymbol{e}\big(L^2(\phi_\circ) \rtimes (00,11)\big) + 2\boldsymbol{e}\big(LR(\phi_\circ) \rtimes (00,11)\big) + \boldsymbol{e}\big(R^2(\phi_\circ) \rtimes (00,11)\big) \equiv 0 \mod p$:
$2z_5^5 + 10z_4z_5^4 + 7z_3z_5^4 + 7z_2z_5^4 + 4z_1z_5^4 + 20z_5^4 + 20z_4^2z_5^3 + 28z_3z_4z_5^3 + 28z_2z_4z_5^3 + 16z_1z_4z_5^3 + 80z_4z_5^3 + 8z_3^2z_5^3 + 16z_2z_3z_5^3 + 10z_1z_3z_5^3 + 50z_3z_5^3 + 8z_2^2z_5^3 + 10z_1z_2z_5^3 + 50z_2z_5^3 + 2z_1^2z_5^3 + 26z_1z_5^3 + 70z_5^3 + 18z_4^3z_5^2 + 39z_3z_4^2z_5^2 + 39z_2z_4^2z_5^2 + 24z_1z_4^2z_5^2 + 114z_4^2z_5^2 + 24z_3^2z_4z_5^2 + 48z_2z_3z_4z_5^2 + 30z_1z_3z_4z_5^2 + 147z_3z_4z_5^2 + 24z_2^2z_4z_5^2 + 30z_1z_2z_4z_5^2 + 147z_2z_4z_5^2 + 6z_1^2z_4z_5^2 + 78z_1z_4z_5^2 + 206z_4z_5^2 + 3z_3^3z_5^2 + 9z_2z_3^2z_5^2 + 6z_1z_3^2z_5^2 + 33z_3^2z_5^2 + 9z_2^2z_3z_5^2 + 12z_1z_2z_3z_5^2 + 66z_2z_3z_5^2 + 3z_1^2z_3z_5^2 + 39z_1z_3z_5^2 + 107z_3z_5^2 + 3z_2^3z_5^2 + 6z_1z_2^2z_5^2 + 33z_2^2z_5^2 + 3z_1^2z_2z_5^2 + 39z_1z_2z_5^2 + 107z_2z_5^2 + 6z_1^2z_5^2 + 50z_1z_5^2 + 100z_5^2 + 6z_4^4z_5 + 18z_3z_4^3z_5 + 18z_2z_4^3z_5 + 12z_1z_4^3z_5 + 54z_4^3z_5 + 18z_3^2z_4^2z_5 + 36z_2z_3z_4^2z_5 + 24z_1z_3z_4^2z_5 + 111z_3z_4^2z_5 + 18z_2^2z_4^2z_5 + 24z_1z_2z_4^2z_5 + 111z_2z_4^2z_5 + 6z_1^2z_4^2z_5 + 66z_1z_4^2z_5 + 160z_4^2z_5 + 6z_3^3z_4z_5 + 18z_2z_3^2z_4z_5 + 12z_1z_3^2z_4z_5 + 60z_3^2z_4z_5 + 18z_2^2z_3z_4z_5 + 24z_1z_2z_3z_4z_5 + 120z_2z_3z_4z_5 + 6z_1^2z_3z_4z_5 + 72z_1z_3z_4z_5 + 185z_3z_4z_5 + 6z_2^3z_4z_5 + 12z_1z_2^2z_4z_5 + 60z_2^2z_4z_5 + 6z_1^2z_2z_4z_5 + 72z_1z_2z_4z_5 + 185z_2z_4z_5 + 12z_1^2z_4z_5 + 92z_1z_4z_5 + 172z_4z_5 + 3z_3^3z_5 + 9z_2z_3^2z_5 + 6z_1z_3^2z_5 + 25z_3^2z_5 + 9z_2^2z_3z_5 + 12z_1z_2z_3z_5 + 50z_2z_3z_5 + 3z_1^2z_3z_5 + 29z_1z_3z_5 + 64z_3z_5 + 3z_2^3z_5 + 6z_1z_2^2z_5 + 25z_2^2z_5 + 3z_1^2z_2z_5 + 29z_1z_2z_5 + 64z_2z_5 + 4z_1^2z_5 + 28z_1z_5 + 48z_5 = 0$

SOLUTION: $(z_1, z_2, z_3, z_4, z_5) = (-1, -2, 0, 1, -2)$.

A.2. **AndOr gate.**

(1) $|\phi| - 2 \equiv 0 \mod p^3$:
$z_1 + z_2 + z_3 + z_4 + z_5 + 4 = 0$

(2) $\boldsymbol{e}\big(\phi \rtimes (11,11)\big) \not\equiv 0 \mod p$:
$z_3z_5^3 + z_2z_5^3 + z_1z_5^3 + 2z_5^3 + 2z_3z_4z_5^2 + 2z_2z_4z_5^2 + 2z_1z_4z_5^2 + 4z_4z_5^2 + 2z_3^2z_5^2 + 3z_2z_3z_5^2 + 2z_1z_3z_5^2 + 9z_3z_5^2 + z_2^2z_5^2 + z_1z_2z_5^2 + 6z_2z_5^2 + 3z_1z_5^2 + 8z_5^2 + z_3z_4^2z_5 + z_2z_4^2z_5 + z_1z_4^2z_5 + 2z_4^2z_5 + 2z_3^2z_4z_5 + 3z_2z_3z_4z_5 + 2z_1z_3z_4z_5 + 9z_3z_4z_5 + z_2^2z_4z_5 + z_1z_2z_4z_5 + 6z_2z_4z_5 + 3z_1z_4z_5 + 8z_4z_5 + z_3^3z_5 + 2z_2z_3^2z_5 + z_1z_3^2z_5 + 7z_3^2z_5 + z_2^2z_3z_5 + z_1z_2z_3z_5 + 8z_2z_3z_5 + 3z_1z_3z_5 + 14z_3z_5 + z_2^2z_5 + z_1z_2z_5 + 6z_2z_5 + 2z_1z_5 + 8z_5 \neq 0$

(3) $\boldsymbol{e}\big(\phi \rtimes (10,01)\big) \not\equiv 0 \mod p$:
$z_5^3 + 2z_4z_5^2 + 2z_3z_5^2 + z_2z_5^2 + 4z_5^2 + z_4^2z_5 + 2z_3z_4z_5 + z_2z_4z_5 + 4z_4z_5 + z_3^2z_5 + z_2z_3z_5 + 4z_3z_5 + z_2z_5 + 3z_5 \neq 0$

(4) $\boldsymbol{e}\big(\phi \rtimes (10,10)\big) \equiv 0 \mod p$:
$z_5^3 + 2z_4z_5^2 + 2z_3z_5^2 + z_2z_5^2 + 6z_5^2 + z_4^2z_5 + 2z_3z_4z_5 + z_2z_4z_5 + 7z_4z_5 + z_3^2z_5 + z_2z_3z_5 + 7z_3z_5 + 3z_2z_5 + 11z_5 + z_4^2 + 2z_3z_4 + z_2z_4 + 5z_4 + z_3^2 + z_2z_3 + 5z_3 + 2z_2 + 6 = 0$

(5) $\boldsymbol{e}\big(\phi \rtimes (01,10)\big) \equiv 0 \mod p$:
$z_3z_5^3 + z_2z_5^3 + z_1z_5^3 + z_5^3 + 2z_3z_4z_5^2 + 2z_2z_4z_5^2 + 2z_1z_4z_5^2 + 2z_4z_5^2 + 2z_3^2z_5^2 + 3z_2z_3z_5^2 + 2z_1z_3z_5^2 + 10z_3z_5^2 + z_2^2z_5^2 + z_1z_2z_5^2 + 8z_2z_5^2 + 6z_1z_5^2 + 8z_5^2 + z_3z_4^2z_5 + z_2z_4^2z_5 + z_1z_4^2z_5 + z_4^2z_5 + 2z_3^2z_4z_5 + 3z_2z_3z_4z_5 + 2z_1z_3z_4z_5 + 11z_3z_4z_5 + z_2^2z_4z_5 + z_1z_2z_4z_5 + 9z_2z_4z_5 + 7z_1z_4z_5 + 9z_4z_5 + z_3^3z_5 + 2z_2z_3^2z_5 + z_1z_3^2z_5 + 10z_3^2z_5 + z_2^2z_3z_5 + z_1z_2z_3z_5 + 13z_2z_3z_5 + 7z_1z_3z_5 + 28z_3z_5 + 3z_2^2z_5 + 3z_1z_2z_5 + 18z_2z_5 + 11z_1z_5 + 19z_5 + z_3z_4^2 + z_2z_4^2 + z_1z_4^2 + z_4^2 + 2z_3^2z_4 + 3z_2z_3z_4 + 2z_1z_3z_4 + 9z_3z_4 + z_2^2z_4 + z_1z_2z_4 + 7z_2z_4 + 5z_1z_4 + 7z_4 + z_3^3 + 2z_2z_3^2 + z_1z_3^2 + 8z_3^2 + z_2^2z_3 + z_1z_2z_3 + 10z_2z_3 + 5z_1z_3 + 19z_3 + 2z_2^2 + 2z_1z_2 + 11z_2 + 6z_1 + 12 = 0$

(6) $\boldsymbol{e}(\phi \rtimes (01,01)\big) \not\equiv 0 \mod p$:
$z_3z_5^3 + z_2z_5^3 + z_1z_5^3 + z_5^3 + 2z_3z_4z_5^2 + 2z_2z_4z_5^2 + 2z_1z_4z_5^2 + 2z_4z_5^2 + 2z_3^2z_5^2 + 3z_2z_3z_5^2 + 2z_1z_3z_5^2 + 8z_3z_5^2 + z_2^2z_5^2 + z_1z_2z_5^2 + 6z_2z_5^2 + 4z_1z_5^2 + 6z_5^2 + z_3z_4^2z_5 + z_2z_4^2z_5 + z_1z_4^2z_5 + z_4^2z_5 + 2z_3^2z_4z_5 + 3z_2z_3z_4z_5 + 2z_1z_3z_4z_5 + 8z_3z_4z_5 + z_2^2z_4z_5 + z_1z_2z_4z_5 + 6z_2z_4z_5 + 4z_1z_4z_5 + 6z_4z_5 + z_3^3z_5 + 2z_2z_3^2z_5 + z_1z_3^2z_5 + 7z_3^2z_5 + z_2^2z_3z_5 + z_1z_2z_3z_5 + 8z_2z_3z_5 + 4z_1z_3z_5 + 14z_3z_5 + z_2^2z_5 + z_1z_2z_5 + 6z_2z_5 + 3z_1z_5 + 8z_5 \neq 0$

(7) $\boldsymbol{e}\big(\phi \rtimes (00,00)\big) \not\equiv 0 \mod p$:
$2z_5^3+4z_4z_5^2+4z_3z_5^2+2z_2z_5^2+12z_5^2+2z_4^2z_5+4z_3z_4z_5+2z_2z_4z_5+13z_4z_5+2z_3^2z_5+2z_2z_3z_5+13z_3z_5+4z_2z_5+18z_5+z_4^2+2z_3z_4+z_2z_4+6z_4+z_3^2+z_2z_3+6z_3+2z_2+8\neq 0$

(8) $-2\boldsymbol{e}\big(M(\phi_\circ)\rtimes(10,11)\big)+\boldsymbol{e}\big(L(\phi_\circ)\rtimes(10,11)\big)+\boldsymbol{e}\big(R(\phi_\circ)\rtimes(10,11)\big)\equiv 0 \mod p$:
$z_5^4+3z_4z_5^3+3z_3z_5^3+2z_2z_5^3+z_1z_5^3+6z_5^3+3z_4^2z_5^2+6z_3z_4z_5^2+4z_2z_4z_5^2+2z_1z_4z_5^2+12z_4z_5^2+3z_3^2z_5^2+4z_2z_3z_5^2+2z_1z_3z_5^2+12z_3z_5^2+z_2^2z_5^2+z_1z_2z_5^2+7z_2z_5^2+3z_1z_5^2+11z_5^2+z_4^3z_5+3z_3z_4^2z_5+2z_2z_4^2z_5+z_1z_4^2z_5+6z_4^2z_5+3z_3^2z_4z_5+4z_2z_3z_4z_5+2z_1z_3z_4z_5+12z_3z_4z_5+z_2^2z_4z_5+z_1z_2z_4z_5+7z_2z_4z_5+3z_1z_4z_5+11z_4z_5+z_3^3z_5+2z_2z_3^2z_5+z_1z_3^2z_5+6z_3^2z_5+z_2^2z_3z_5+z_1z_2z_3z_5+7z_2z_3z_5+3z_1z_3z_5+11z_3z_5+z_2^2z_5+z_1z_2z_5+5z_2z_5+2z_1z_5+6z_5=0$

(9) $-2\boldsymbol{e}\big(M(\phi_\circ)\rtimes(01,11)\big)+\boldsymbol{e}\big(L(\phi_\circ)\rtimes(01,11)\big)+\boldsymbol{e}\big(R(\phi_\circ)\rtimes(01,11)\big)\equiv 0 \mod p$:
$z_3z_5^4+z_2z_5^4+z_1z_5^4+z_5^4+3z_3z_4z_5^3+3z_2z_4z_5^3+3z_1z_4z_5^3+3z_4z_5^3+3z_3^2z_5^3+5z_2z_3z_5^3+4z_1z_3z_5^3+12z_3z_5^3+2z_2^2z_5^3+3z_1z_2z_5^3+10z_2z_5^3+z_1^2z_5^3+8z_1z_5^3+9z_5^3+3z_3z_4^2z_5^2+3z_2z_4^2z_5^2+3z_1z_4^2z_5^2+3z_4^2z_5^2+6z_3^2z_4z_5^2+10z_2z_3z_4z_5^2+8z_1z_3z_4z_5^2+24z_3z_4z_5^2+4z_2^2z_4z_5^2+6z_1z_2z_4z_5^2+20z_2z_4z_5^2+2z_1^2z_4z_5^2+16z_1z_4z_5^2+18z_4z_5^2+3z_3^3z_5^2+7z_2z_3^2z_5^2+5z_1z_3^2z_5^2+21z_3^2z_5^2+5z_2^2z_3z_5^2+7z_1z_2z_3z_5^2+31z_2z_3z_5^2+2z_1^2z_3z_5^2+21z_1z_3z_5^2+44z_3z_5^2+z_2^3z_5^2+2z_1z_2^2z_5^2+10z_2^2z_5^2+z_1^2z_2z_5^2+13z_1z_2z_5^2+30z_2z_5^2+3z_1^2z_5^2+19z_1z_5^2+26z_5^2+z_3z_4^3z_5+z_2z_4^3z_5+z_1z_4^3z_5+z_4^3z_5+3z_3^2z_4^2z_5+5z_2z_3z_4^2z_5+4z_1z_3z_4^2z_5+12z_3z_4^2z_5+2z_2^2z_4^2z_5+3z_1z_2z_4^2z_5+10z_2z_4^2z_5+z_1^2z_4^2z_5+8z_1z_4^2z_5+9z_4^2z_5+3z_3^3z_4z_5+7z_2z_3^2z_4z_5+5z_1z_3^2z_4z_5+21z_3^2z_4z_5+5z_2^2z_3z_4z_5+7z_1z_2z_3z_4z_5+31z_2z_3z_4z_5+2z_1^2z_3z_4z_5+21z_1z_3z_4z_5+44z_3z_4z_5+z_2^3z_4z_5+2z_1z_2^2z_4z_5+10z_2^2z_4z_5+z_1^2z_2z_4z_5+13z_1z_2z_4z_5+30z_2z_4z_5+3z_1^2z_4z_5+19z_1z_4z_5+26z_4z_5+z_3^4z_5+3z_2z_3^3z_5+2z_1z_3^3z_5+10z_3^3z_5+3z_2^2z_3^2z_5+4z_1z_2z_3^2z_5+21z_2z_3^2z_5+z_1^2z_3^2z_5+13z_1z_3^2z_5+35z_3^2z_5+z_2^3z_3z_5+2z_1z_2^2z_3z_5+12z_2^2z_3z_5+z_1^2z_2z_3z_5+15z_1z_2z_3z_5+44z_2z_3z_5+3z_1^2z_3z_5+25z_1z_3z_5+50z_3z_5+z_2^3z_5+2z_1z_2^2z_5+9z_2^2z_5+z_1^2z_2z_5+11z_1z_2z_5+26z_2z_5+2z_1^2z_5+14z_1z_5+24z_5=0$

(10) $-2\boldsymbol{e}\big(M(\phi_\circ)\rtimes(00,01)\big)+\boldsymbol{e}\big(L(\phi_\circ)\rtimes(00,01)\big)+\boldsymbol{e}\big(R(\phi_\circ)\rtimes(00,01)\big)\equiv 0 \mod p$:
$z_5^4+3z_4z_5^3+3z_3z_5^3+2z_2z_5^3+z_1z_5^3+8z_5^3+3z_4^2z_5^2+6z_3z_4z_5^2+4z_2z_4z_5^2+2z_1z_4z_5^2+16z_4z_5^2+3z_3^2z_5^2+4z_2z_3z_5^2+2z_1z_3z_5^2+16z_3z_5^2+z_2^2z_5^2+z_1z_2z_5^2+9z_2z_5^2+4z_1z_5^2+19z_5^2+z_4^3z_5+3z_3z_4^2z_5+2z_2z_4^2z_5+z_1z_4^2z_5+8z_4^2z_5+3z_3^2z_4z_5+4z_2z_3z_4z_5+2z_1z_3z_4z_5+16z_3z_4z_5+z_2^2z_4z_5+z_1z_2z_4z_5+9z_2z_4z_5+4z_1z_4z_5+19z_4z_5+z_3^3z_5+2z_2z_3^2z_5+z_1z_3^2z_5+8z_3^2z_5+z_2^2z_3z_5+z_1z_2z_3z_5+9z_2z_3z_5+4z_1z_3z_5+19z_3z_5+z_2^2z_5+z_1z_2z_5+7z_2z_5+3z_1z_5+12z_5=0$

(11) $-2\boldsymbol{e}\big(M(\phi_\circ)\rtimes(00,10)\big)+\boldsymbol{e}\big(L(\phi_\circ)\rtimes(00,10)\big)+\boldsymbol{e}\big(R(\phi_\circ)\rtimes(00,10)\big)\equiv 0 \mod p$:
$z_5^4+3z_4z_5^3+3z_3z_5^3+2z_2z_5^3+z_1z_5^3+10z_5^3+3z_4^2z_5^2+6z_3z_4z_5^2+4z_2z_4z_5^2+2z_1z_4z_5^2+21z_4z_5^2+3z_3^2z_5^2+4z_2z_3z_5^2+2z_1z_3z_5^2+21z_3z_5^2+z_2^2z_5^2+z_1z_2z_5^2+13z_2z_5^2+6z_1z_5^2+35z_5^2+z_4^3z_5+3z_3z_4^2z_5+2z_2z_4^2z_5+z_1z_4^2z_5+12z_4^2z_5+3z_3^2z_4z_5+4z_2z_3z_4z_5+2z_1z_3z_4z_5+24z_3z_4z_5+z_2^2z_4z_5+z_1z_2z_4z_5+15z_2z_4z_5+7z_1z_4z_5+44z_4z_5+z_3^3z_5+2z_2z_3^2z_5+z_1z_3^2z_5+12z_3^2z_5+z_2^2z_3z_5+z_1z_2z_3z_5+15z_2z_3z_5+7z_1z_3z_5+44z_3z_5+3z_2^2z_5+3z_1z_2z_5+25z_2z_5+11z_1z_5+50z_5+z_4^3+3z_3z_4^2+2z_2z_4^2+z_1z_4^2+9z_4^2+3z_3^2z_4+4z_2z_3z_4+2z_1z_3z_4+18z_3z_4+z_2^2z_4+z_1z_2z_4+11z_2z_4+5z_1z_4+26z_4+z_3^3+2z_2z_3^2+z_1z_3^2+9z_3^2+z_2^2z_3+z_1z_2z_3+11z_2z_3+5z_1z_3+26z_3+2z_2^2+2z_1z_2+14z_2+6z_1+24=0$

(12) $2\boldsymbol{e}\big(M^2(\phi_\circ)\rtimes(00,11)\big)-4\boldsymbol{e}\big(LM(\phi_\circ)\rtimes(00,11)\big)-4\boldsymbol{e}\big(RM(\phi_\circ)\rtimes(00,11)\big)+$
$\boldsymbol{e}\big(L^2(\phi_\circ)\rtimes(00,11)\big)+2\boldsymbol{e}\big(LR(\phi_\circ)\rtimes(00,11)\big)+\boldsymbol{e}\big(R^2(\phi_\circ)\rtimes(00,11)\big)\equiv 0 \mod p$:
$z_5^5+4z_4z_5^4+4z_3z_5^4+3z_2z_5^4+2z_1z_5^4+10z_5^4+6z_4^2z_5^3+12z_3z_4z_5^3+9z_2z_4z_5^3+6z_1z_4z_5^3+30z_4z_5^3+6z_3^2z_5^3+9z_2z_3z_5^3+6z_1z_3z_5^3+30z_3z_5^3+3z_2^2z_5^3+4z_1z_2z_5^3+21z_2z_5^3+z_1^2z_5^3+13z_1z_5^3+35z_5^3+4z_4^3z_5^2+12z_3z_4^2z_5^2+9z_2z_4^2z_5^2+6z_1z_4^2z_5^2+30z_4^2z_5^2+12z_3^2z_4z_5^2+18z_2z_3z_4z_5^2+12z_1z_3z_4z_5^2+60z_3z_4z_5^2+6z_2^2z_4z_5^2+8z_1z_2z_4z_5^2+42z_2z_4z_5^2+2z_1^2z_4z_5^2+26z_1z_4z_5^2+70z_4z_5^2+4z_3^3z_5^2+9z_2z_3^2z_5^2+6z_1z_3^2z_5^2+30z_3^2z_5^2+6z_2^2z_3z_5^2+8z_1z_2z_3z_5^2+42z_2z_3z_5^2+2z_1^2z_3z_5^2+26z_1z_3z_5^2+70z_3z_5^2+z_2^3z_5^2+2z_1z_2^2z_5^2+12z_2^2z_5^2+z_1^2z_2z_5^2+15z_1z_2z_5^2+44z_2z_5^2+3z_1^2z_5^2+25z_1z_5^2+50z_5^2+1(z_4^4)z_5+4z_3z_4^3z_5+3z_2z_4^3z_5+2z_1z_4^3z_5+10z_4^3z_5+6z_3^2z_4^2z_5+9z_2z_3z_4^2z_5+6z_1z_3z_4^2z_5+30z_3z_4^2z_5+3z_2^2z_4^2z_5+4z_1z_2z_4^2z_5+21z_2z_4^2z_5+z_1^2z_4^2z_5+13z_1z_4^2z_5+35z_4^2z_5+4z_3^3z_4z_5+9z_2z_3^2z_4z_5+6z_1z_3^2z_4z_5+30z_3^2z_4z_5+6z_2^2z_3z_4z_5+8z_1z_2z_3z_4z_5+42z_2z_3z_4z_5+2z_1^2z_3z_4z_5+26z_1z_3z_4z_5+70z_3z_4z_5+z_2^3z_4z_5+2z_1z_2^2z_4z_5+12z_2^2z_4z_5+z_1^2z_2z_4z_5+15z_1z_2z_4z_5+44z_2z_4z_5+3z_1^2z_4z_5+25z_1z_4z_5+50z_4z_5+z_3^4z_5+3z_2z_3^3z_5+2z_1z_3^3z_5+10z_3^3z_5+3z_2^2z_3^2z_5+4z_1z_2z_3^2z_5+21z_2z_3^2z_5+z_1^2z_3^2z_5+13z_1z_3^2z_5+35z_3^2z_5+z_2^3z_3z_5+2z_1z_2^2z_3z_5+12z_2^2z_3z_5+z_1^2z_2z_3z_5+15z_1z_2z_3z_5+44z_2z_3z_5+3z_1^2z_3z_5+25z_1z_3z_5+50z_3z_5+z_2^3z_5+2z_1z_2^2z_5+9z_2^2z_5+z_1^2z_2z_5+11z_1z_2z_5+26z_2z_5+2z_1^2z_5+14z_1z_5+24z_5=0$

SOLUTION: $(z_1,z_2,z_3,z_4,z_5)=(-2,1,-3,1,-1)$.
The nonzero values $\boldsymbol{e}(\sigma\rtimes(v,v'))$ takes are 2 and 4.

A.3. **TestEq gate.**

(1) $|\phi| - 2 \equiv 0 \mod p^3$:
$z_1 + z_2 + z_3 + z_4 + z_5 + 4 = 0$

(2) $\boldsymbol{e}\big(\phi \rtimes (11,11)\big) \not\equiv 0 \mod p$:
$2z_3z_5^3 + 2z_2z_5^3 + 2z_1z_5^3 + 4z_5^3 + 6z_3z_4z_5^2 + 6z_2z_4z_5^2 + 6z_1z_4z_5^2 + 12z_4z_5^2 + 6z_3^2z_5^2 + 9z_2z_3z_5^2 + 6z_1z_3z_5^2 + 24z_3z_5^2 + 3z_2^2z_5^2 + 3z_1z_2z_5^2 + 15z_2z_5^2 + 6z_1z_5^2 + 18z_5^2 + 6z_3z_4^2z_5 + 6z_2z_4^2z_5 + 6z_1z_4^2z_5 + 12z_4^2z_5 + 12z_3^2z_4z_5 + 18z_2z_3z_4z_5 + 12z_1z_3z_4z_5 + 48z_3z_4z_5 + 6z_2^2z_4z_5 + 6z_1z_2z_4z_5 + 30z_2z_4z_5 + 12z_1z_4z_5 + 36z_4z_5 + 6z_3^3z_5 + 12z_2z_3^2z_5 + 6z_1z_3^2z_5 + 36z_3^2z_5 + 6z_2^2z_3z_5 + 6z_1z_2z_3z_5 + 39z_2z_3z_5 + 12z_1z_3z_5 + 58z_3z_5 + 3z_2^2z_5 + 3z_1z_2z_5 + 19z_2z_5 + 4z_1z_5 + 26z_5 \neq 0$

(3) $\boldsymbol{e}\big(\phi \rtimes (10,01)\big) \equiv 0 \mod p$:
$2z_5^3 + 6z_4z_5^2 + 6z_3z_5^2 + 3z_2z_5^2 + 12z_5^2 + 6z_4^2z_5 + 12z_3z_4z_5 + 6z_2z_4z_5 + 24z_4z_5 + 6z_3^2z_5 + 6z_2z_3z_5 + 24z_3z_5 + 9z_2z_5 + 16z_5 + 6z_4^2 + 12z_3z_4 + 6z_2z_4 + 12z_4 + 6z_3^2 + 6z_2z_3 + 12z_3 = 0$

(4) $\boldsymbol{e}\big(\phi \rtimes (10,10)\big) \equiv 0 \mod p$:
$2z_5^3 + 6z_4z_5^2 + 6z_3z_5^2 + 3z_2z_5^2 + 12z_5^2 + 6z_4^2z_5 + 12z_3z_4z_5 + 6z_2z_4z_5 + 24z_4z_5 + 6z_3^2z_5 + 6z_2z_3z_5 + 24z_3z_5 + 9z_2z_5 + 22z_5 + 6z_4^2 + 12z_3z_4 + 6z_2z_4 + 18z_4 + 6z_3^2 + 6z_2z_3 + 18z_3 + 6z_2 + 12 = 0$

(5) $\boldsymbol{e}\big(\phi \rtimes (01,10)\big) \equiv 0 \mod p$:
$2z_3z_5^3 + 2z_2z_5^3 + 2z_1z_5^3 + 2z_5^3 + 6z_3z_4z_5^2 + 6z_2z_4z_5^2 + 6z_1z_4z_5^2 + 6z_4z_5^2 + 6z_3^2z_5^2 + 9z_2z_3z_5^2 + 6z_1z_3z_5^2 + 24z_3z_5^2 + 3z_2^2z_5^2 + 3z_1z_2z_5^2 + 18z_2z_5^2 + 12z_1z_5^2 + 18z_5^2 + 6z_3z_4^2z_5 + 6z_2z_4^2z_5 + 6z_1z_4^2z_5 + 6z_4^2z_5 + 12z_3^2z_4z_5 + 18z_2z_3z_4z_5 + 12z_1z_3z_4z_5 + 48z_3z_4z_5 + 6z_2^2z_4z_5 + 6z_1z_2z_4z_5 + 36z_2z_4z_5 + 24z_1z_4z_5 + 36z_4z_5 + 6z_3^3z_5 + 12z_2z_3^2z_5 + 6z_1z_3^2z_5 + 42z_3^2z_5 + 6z_2^2z_3z_5 + 6z_1z_2z_3z_5 + 51z_2z_3z_5 + 24z_1z_3z_5 + 88z_3z_5 + 9z_2^2z_5 + 9z_1z_2z_5 + 46z_2z_5 + 22z_1z_5 + 52z_5 + 6z_3z_4^2 + 6z_2z_4^2 + 6z_1z_4^2 + 6z_4^2 + 12z_3^2z_4 + 18z_2z_3z_4 + 12z_1z_3z_4 + 42z_3z_4 + 6z_2^2z_4 + 6z_1z_2z_4 + 30z_2z_4 + 18z_1z_4 + 30z_4 + 6z_3^3 + 12z_2z_3^2 + 6z_1z_3^2 + 36z_3^2 + 6z_2^2z_3 + 6z_1z_2z_3 + 42z_2z_3 + 18z_1z_3 + 66z_3 + 6z_2^2 + 6z_1z_2 + 30z_2 + 12z_1 + 36 = 0$

(6) $\boldsymbol{e}(\phi \rtimes (01,01)\big) \equiv 0 \mod p$:
$2z_3z_5^3 + 2z_2z_5^3 + 2z_1z_5^3 + 2z_5^3 + 6z_3z_4z_5^2 + 6z_2z_4z_5^2 + 6z_1z_4z_5^2 + 6z_4z_5^2 + 6z_3^2z_5^2 + 9z_2z_3z_5^2 + 6z_1z_3z_5^2 + 24z_3z_5^2 + 3z_2^2z_5^2 + 3z_1z_2z_5^2 + 18z_2z_5^2 + 12z_1z_5^2 + 18z_5^2 + 6z_3z_4^2z_5 + 6z_2z_4^2z_5 + 6z_1z_4^2z_5 + 6z_4^2z_5 + 12z_3^2z_4z_5 + 18z_2z_3z_4z_5 + 12z_1z_3z_4z_5 + 48z_3z_4z_5 + 6z_2^2z_4z_5 + 6z_1z_2z_4z_5 + 36z_2z_4z_5 + 24z_1z_4z_5 + 36z_4z_5 + 6z_3^3z_5 + 12z_2z_3^2z_5 + 6z_1z_3^2z_5 + 42z_3^2z_5 + 6z_2^2z_3z_5 + 6z_1z_2z_3z_5 + 51z_2z_3z_5 + 24z_1z_3z_5 + 82z_3z_5 + 9z_2^2z_5 + 9z_1z_2z_5 + 40z_2z_5 + 16z_1z_5 + 46z_5 + 6z_3z_4^2 + 6z_2z_4^2 + 6z_1z_4^2 + 6z_4^2 + 12z_3^2z_4 + 18z_2z_3z_4 + 12z_1z_3z_4 + 36z_3z_4 + 6z_2^2z_4 + 6z_1z_2z_4 + 24z_2z_4 + 12z_1z_4 + 24z_4 + 6z_3^3 + 12z_2z_3^2 + 6z_1z_3^2 + 30z_3^2 + 6z_2^2z_3 + 6z_1z_2z_3 + 30z_2z_3 + 12z_1z_3 + 42z_3 + 6z_2 + 18 = 0$

(7) $\boldsymbol{e}(\phi \rtimes (00,00)\big) \not\equiv 0 \mod p$:
$2z_5^3 + 6z_4z_5^2 + 6z_3z_5^2 + 3z_2z_5^2 + 18z_5^2 + 6z_4^2z_5 + 12z_3z_4z_5 + 6z_2z_4z_5 + 36z_4z_5 + 6z_3^2z_5 + 6z_2z_3z_5 + 36z_3z_5 + 15z_2z_5 + 40z_5 + 12z_4^2 + 24z_3z_4 + 12z_2z_4 + 36z_4 + 12z_3^2 + 12z_2z_3 + 36z_3 + 6z_2 + 15 \neq 0$

(8) $-2\boldsymbol{e}\big(M(\phi_\circ) \rtimes (10,11)\big) + \boldsymbol{e}\big(L(\phi_\circ) \rtimes (10,11)\big) + \boldsymbol{e}\big(R(\phi_\circ) \rtimes (10,11)\big) \equiv 0 \mod p$:
$2z_5^4 + 8z_4z_5^3 + 8z_3z_5^3 + 5z_2z_5^3 + 2z_1z_5^3 + 12z_5^3 + 12z_4^2z_5^2 + 24z_3z_4z_5^2 + 15z_2z_4z_5^2 + 6z_1z_4z_5^2 + 36z_4z_5^2 + 12z_3^2z_5^2 + 15z_2z_3z_5^2 + 6z_1z_3z_5^2 + 36z_3z_5^2 + 3z_2^2z_5^2 + 3z_1z_2z_5^2 + 18z_2z_5^2 + 6z_1z_5^2 + 22z_5^2 + 6z_4^3z_5 + 18z_3z_4^2z_5 + 12z_2z_4^2z_5 + 6z_1z_4^2z_5 + 30z_4^2z_5 + 18z_3^2z_4z_5 + 24z_2z_3z_4z_5 + 12z_1z_3z_4z_5 + 60z_3z_4z_5 + 6z_2^2z_4z_5 + 6z_1z_2z_4z_5 + 33z_2z_4z_5 + 12z_1z_4z_5 + 40z_4z_5 + 6z_3^3z_5 + 12z_2z_3^2z_5 + 6z_1z_3^2z_5 + 30z_3^2z_5 + 6z_2^2z_3z_5 + 6z_1z_2z_3z_5 + 33z_2z_3z_5 + 12z_1z_3z_5 + 40z_3z_5 + 3z_2^2z_5 + 3z_1z_2z_5 + 13z_2z_5 + 4z_1z_5 + 12z_5 = 0$

(9) $-2\boldsymbol{e}\big(M(\phi_\circ) \rtimes (01,11)\big) + \boldsymbol{e}\big(L(\phi_\circ) \rtimes (01,11)\big) + \boldsymbol{e}\big(R(\phi_\circ) \rtimes (01,11)\big) \equiv 0 \mod p$:
$2z_3z_5^4 + 2z_2z_5^4 + 2z_1z_5^4 + 2z_5^4 + 8z_3z_4z_5^3 + 8z_2z_4z_5^3 + 8z_1z_4z_5^3 + 8z_4z_5^3 + 8z_3^2z_5^3 + 13z_2z_3z_5^3 + 10z_1z_3z_5^3 + 28z_3z_5^3 + 5z_2^2z_5^3 + 7z_1z_2z_5^3 + 22z_2z_5^3 + 2z_1^2z_5^3 + 16z_1z_5^3 + 20z_5^3 + 12z_3z_4^2z_5^2 + 12z_2z_4^2z_5^2 + 12z_1z_4^2z_5^2 + 12z_4^2z_5^2 + 24z_3^2z_4z_5^2 + 39z_2z_3z_4z_5^2 + 30z_1z_3z_4z_5^2 + 84z_3z_4z_5^2 + 15z_2^2z_4z_5^2 + 21z_1z_2z_4z_5^2 + 66z_2z_4z_5^2 + 6z_1^2z_4z_5^2 + 48z_1z_4z_5^2 + 60z_4z_5^2 + 12z_3^3z_5^2 + 27z_2z_3^2z_5^2 + 18z_1z_3^2z_5^2 + 72z_3^2z_5^2 + 18z_2^2z_3z_5^2 + 24z_1z_2z_3z_5^2 + 99z_2z_3z_5^2 + 6z_1^2z_3z_5^2 + 60z_1z_3z_5^2 + 130z_3z_5^2 + 3z_2^3z_5^2 + 6z_1z_2^2z_5^2 + 27z_2^2z_5^2 + 3z_1^2z_2z_5^2 + 33z_1z_2z_5^2 + 76z_2z_5^2 + 6z_1^2z_5^2 + 40z_1z_5^2 + 70z_5^2 + 6z_3z_4^3z_5 + 6z_2z_4^3z_5 + 6z_1z_4^3z_5 + 6z_4^3z_5 + 18z_3^2z_4^2z_5 + 30z_2z_3z_4^2z_5 + 24z_1z_3z_4^2z_5 + 66z_3z_4^2z_5 + 12z_2^2z_4^2z_5 + 18z_1z_2z_4^2z_5 + 54z_2z_4^2z_5 + 6z_1^2z_4^2z_5 + 42z_1z_4^2z_5 + 48z_4^2z_5 + 18z_3^3z_4z_5 + 42z_2z_3^2z_4z_5 + 30z_1z_3^2z_4z_5 + 114z_3^2z_4z_5 + 30z_2^2z_3z_4z_5 + 42z_1z_2z_3z_4z_5 + 165z_2z_3z_4z_5 + 12z_1^2z_3z_4z_5 + 108z_1z_3z_4z_5 + 214z_3z_4z_5 + 6z_2^3z_4z_5 + 12z_1z_2^2z_4z_5 + 51z_2^2z_4z_5 + 6z_1^2z_2z_4z_5 + 63z_1z_2z_4z_5 + 136z_2z_4z_5 + 12z_1^2z_4z_5 + 76z_1z_4z_5 + 118z_4z_5 + 6z_3^4z_5 + 18z_2z_3^3z_5 + 12z_1z_3^3z_5 + 54z_3^3z_5 + 18z_2^2z_3^2z_5 + 24z_1z_2z_3^2z_5 + 111z_2z_3^2z_5 + 6z_1^2z_3^2z_5 + 66z_1z_3^2z_5 + 166z_3^2z_5 + 6z_2^3z_3z_5 + 12z_1z_2^2z_3z_5 + 60z_2^2z_3z_5 + 6z_1^2z_2z_3z_5 + 72z_1z_2z_3z_5 + 194z_2z_3z_5 + 12z_1^2z_3z_5 + 98z_1z_3z_5 + 206z_3z_5 + 3z_2^3z_5 + 6z_1z_2^2z_5 + 28z_2^2z_5 + 3z_1^2z_2z_5 + 32z_1z_2z_5 + 86z_2z_5 + 4z_1^2z_5 + 38z_1z_5 + 88z_5 = 0$

(10) $-2\boldsymbol{e}\big(M(\phi_\circ) \rtimes (00,01)\big) + \boldsymbol{e}\big(L(\phi_\circ) \rtimes (00,01)\big) + \boldsymbol{e}\big(R(\phi_\circ) \rtimes (00,01)\big) \equiv 0 \mod p$:
$2z_5^4 + 8z_4z_5^3 + 8z_3z_5^3 + 5z_2z_5^3 + 2z_1z_5^3 + 20z_5^3 + 12z_4^2z_5^2 + 24z_3z_4z_5^2 + 15z_2z_4z_5^2 + 6z_1z_4z_5^2 + 60z_4z_5^2 + 12z_3^2z_5^2 + 15z_2z_3z_5^2 + 6z_1z_3z_5^2 + 60z_3z_5^2 + 3z_2^2z_5^2 + 3z_1z_2z_5^2 + 33z_2z_5^2 + 12z_1z_5^2 + 64z_5^2 + 6z_4^3z_5 + 18z_3z_4^2z_5 + 12z_2z_4^2z_5 + 6z_1z_4^2z_5 + 54z_4^2z_5 + 18z_3^2z_4z_5 + 24z_2z_3z_4z_5 + 12z_1z_3z_4z_5 + 108z_3z_4z_5 + 6z_2^2z_4z_5 + 6z_1z_2z_4z_5 + 63z_2z_4z_5 + 24z_1z_4z_5 + 124z_4z_5 + 6z_3^3z_5 + 12z_2z_3^2z_5 + 6z_1z_3^2z_5 + 54z_3^2z_5 + 6z_2^2z_3z_5 + 6z_1z_2z_3z_5 + 63z_2z_3z_5 + 24z_1z_3z_5 + 124z_3z_5 + 9z_2^2z_5 + 9z_1z_2z_5 + 52z_2z_5 + 16z_1z_5 + 64z_5 + 6z_4^3 + 18z_3z_4^2 + 12z_2z_4^2 + 6z_1z_4^2 + 36z_4^2 + 18z_3^2z_4 + 24z_2z_3z_4 + 12z_1z_3z_4 + 72z_3z_4 + 6z_2^2z_4 + 6z_1z_2z_4 + 36z_2z_4 + 12z_1z_4 + 48z_4 + 6z_3^3 + 12z_2z_3^2 + 6z_1z_3^2 + 36z_3^2 + 6z_2^2z_3 + 6z_1z_2z_3 + 36z_2z_3 + 12z_1z_3 + 48z_3 = 0$

(11) $-2\boldsymbol{e}\big(M(\phi_\circ) \rtimes (00,10)\big) + \boldsymbol{e}\big(L(\phi_\circ) \rtimes (00,10)\big) + \boldsymbol{e}\big(R(\phi_\circ) \rtimes (00,10)\big) \equiv 0 \mod p$:
$2z_5^4 + 8z_4z_5^3 + 8z_3z_5^3 + 5z_2z_5^3 + 2z_1z_5^3 + 20z_5^3 + 12z_4^2z_5^2 + 24z_3z_4z_5^2 + 15z_2z_4z_5^2 + 6z_1z_4z_5^2 + 60z_4z_5^2 + 12z_3^2z_5^2 + 15z_2z_3z_5^2 + 6z_1z_3z_5^2 + 60z_3z_5^2 + 3z_2^2z_5^2 + 3z_1z_2z_5^2 + 33z_2z_5^2 + 12z_1z_5^2 + 70z_5^2 + 6z_4^3z_5 + 18z_3z_4^2z_5 + 12z_2z_4^2z_5 + 6z_1z_4^2z_5 + 54z_4^2z_5 + 18z_3^2z_4z_5 + 24z_2z_3z_4z_5 + 12z_1z_3z_4z_5 + 108z_3z_4z_5 + 6z_2^2z_4z_5 + 6z_1z_2z_4z_5 + 63z_2z_4z_5 + 24z_1z_4z_5 + 136z_4z_5 + 6z_3^3z_5 + 12z_2z_3^2z_5 + 6z_1z_3^2z_5 + 54z_3^2z_5 + 6z_2^2z_3z_5 + 6z_1z_2z_3z_5 + 63z_2z_3z_5 + 24z_1z_3z_5 + 136z_3z_5 + 9z_2^2z_5 + 9z_1z_2z_5 + 64z_2z_5 + 22z_1z_5 + 100z_5 + 6z_4^3 + 18z_3z_4^2 + 12z_2z_4^2 + 6z_1z_4^2 + 42z_4^2 + 18z_3^2z_4 + 24z_2z_3z_4 + 12z_1z_3z_4 + 84z_3z_4 + 6z_2^2z_4 + 6z_1z_2z_4 + 48z_2z_4 + 18z_1z_4 + 84z_4 + 6z_3^3 + 12z_2z_3^2 + 6z_1z_3^2 + 42z_3^2 + 6z_2^2z_3 + 6z_1z_2z_3 + 48z_2z_3 + 18z_1z_3 + 84z_3 + 6z_2^2 + 6z_1z_2 + 36z_2 + 12z_1 + 48 = 0$

(12) $2\boldsymbol{e}\big(M^2(\phi_\circ) \rtimes (00,11)\big) - 4\boldsymbol{e}\big(LM(\phi_\circ) \rtimes (00,11)\big) - 4\boldsymbol{e}\big(RM(\phi_\circ) \rtimes (00,11)\big) +$
$\boldsymbol{e}\big(L^2(\phi_\circ) \rtimes (00,11)\big) + 2\boldsymbol{e}\big(LR(\phi_\circ) \rtimes (00,11)\big) + \boldsymbol{e}\big(R^2(\phi_\circ) \rtimes (00,11)\big) \equiv 0 \mod p$:
$2z_5^5 + 10z_4z_5^4 + 10z_3z_5^4 + 7z_2z_5^4 + 4z_1z_5^4 + 20z_5^4 + 20z_4^2z_5^3 + 40z_3z_4z_5^3 + 28z_2z_4z_5^3 + 16z_1z_4z_5^3 + 80z_4z_5^3 + 20z_3^2z_5^3 + 28z_2z_3z_5^3 + 16z_1z_3z_5^3 + 80z_3z_5^3 + 8z_2^2z_5^3 + 10z_1z_2z_5^3 + 50z_2z_5^3 + 2z_1^2z_5^3 + 26z_1z_5^3 + 70z_5^3 + 18z_4^3z_5^2 + 54z_3z_4^2z_5^2 + 39z_2z_4^2z_5^2 + 24z_1z_4^2z_5^2 + 114z_4^2z_5^2 + 54z_3^2z_4z_5^2 + 78z_2z_3z_4z_5^2 + 48z_1z_3z_4z_5^2 + 228z_3z_4z_5^2 + 24z_2^2z_4z_5^2 + 30z_1z_2z_4z_5^2 + 147z_2z_4z_5^2 + 6z_1^2z_4z_5^2 + 78z_1z_4z_5^2 + 206z_4z_5^2 + 18z_3^3z_5^2 + 39z_2z_3^2z_5^2 + 24z_1z_3^2z_5^2 + 114z_3^2z_5^2 + 24z_2^2z_3z_5^2 + 30z_1z_2z_3z_5^2 + 147z_2z_3z_5^2 + 6z_1^2z_3z_5^2 + 78z_1z_3z_5^2 + 206z_3z_5^2 + 3z_2^3z_5^2 + 6z_1z_2^2z_5^2 + 33z_2^2z_5^2 + 3z_1^2z_2z_5^2 + 39z_1z_2z_5^2 + 107z_2z_5^2 + 6z_1^2z_5^2 + 50z_1z_5^2 + 100z_5^2 + 6(z_4^4)z_5 + 24z_3z_4^3z_5 + 18z_2z_4^3z_5 + 12z_1z_4^3z_5 + 54z_4^3z_5 + 36z_3^2z_4^2z_5 + 54z_2z_3z_4^2z_5 + 36z_1z_3z_4^2z_5 + 162z_3z_4^2z_5 + 18z_2^2z_4^2z_5 + 24z_1z_2z_4^2z_5 + 111z_2z_4^2z_5 + 6z_1^2z_4^2z_5 + 66z_1z_4^2z_5 + 160z_4^2z_5 + 24z_3^3z_4z_5 + 54z_2z_3^2z_4z_5 + 36z_1z_3^2z_4z_5 + 162z_3^2z_4z_5 + 36z_2^2z_3z_4z_5 + 48z_1z_2z_3z_4z_5 + 222z_2z_3z_4z_5 + 12z_1^2z_3z_4z_5 + 132z_1z_3z_4z_5 + 320z_3z_4z_5 + 6z_2^3z_4z_5 + 12z_1z_2^2z_4z_5 + 60z_2^2z_4z_5 + 6z_1^2z_2z_4z_5 + 72z_1z_2z_4z_5 + 185z_2z_4z_5 + 12z_1^2z_4z_5 + 92z_1z_4z_5 + 172z_4z_5 + 6z_3^4z_5 + 18z_2z_3^3z_5 + 12z_1z_3^3z_5 + 54z_3^3z_5 + 18z_2^2z_3^2z_5 + 24z_1z_2z_3^2z_5 + 111z_2z_3^2z_5 + 6z_1^2z_3^2z_5 + 66z_1z_3^2z_5 + 160z_3^2z_5 + 6z_2^3z_3z_5 + 12z_1z_2^2z_3z_5 + 60z_2^2z_3z_5 + 6z_1^2z_2z_3z_5 + 72z_1z_2z_3z_5 + 185z_2z_3z_5 + 12z_1^2z_3z_5 + 92z_1z_3z_5 + 172z_3z_5 + 3z_2^3z_5 + 6z_1z_2^2z_5 + 25z_2^2z_5 + 3z_1^2z_2z_5 + 29z_1z_2z_5 + 64z_2z_5 + 4z_1^2z_5 + 28z_1z_5 + 48z_5 = 0$

SOLUTION: $(z_1, z_2, z_3, z_4, z_5) = (-2, -\frac{8}{3}, \frac{5}{3}, -3, 2)$.
The nonzero values $\boldsymbol{e}(\sigma \rtimes (v, v'))$ takes are $\frac{7}{3}$ and $\frac{-8}{3}$.