Add/XNOR Chains: An NP-Complete Boolean–Arithmetic Problem at the Meet-in-the-Middle Barrier

Ping Wang[1], Mengxin Lv[1], Zishen Shen[2] and Fangguo Zhang[3]

[1] College of Electronics and Information Engineering,
Shenzhen University, Shenzhen 518060, China
wangping@szu.edu.cn, 2510044047@mail.szu.edu.cn
[2] Department of Mathematics,
University of California, Berkeley, California 94720, USA
zishen@berkeley.edu
[3] School of Computer Science and Engineering,
Sun Yat-sen University, Guangzhou 510006, China
isszhfg@mail.sysu.edu.cn
November 2, 2026

Abstract. We introduce the Add/XNOR decision problem: given n words of n bits, decide whether they can be combined into zero by a left-associative chain in which each operator is either addition modulo 2^n or bitwise XNOR. We prove NP-completeness by a deterministic reduction from Subset-Sum in which four-bit guard fields and a parity normalization force every gadget to be all-additions or all-XNORs; hardness persists for every width that is a fixed rational power of the operand count, and 3-SAT reduces with linear parameter blow-up. The problem is a minimal model of Boolean–arithmetic mixing, and we give an exact theory of its linear and differential behaviour: a carry decomposition, closed forms for every single-bit linear correlation (for addition runs $z_1, \ldots, z_r$ the bit-1 correlation is $\prod_j 2^{-z_j/2} \cos(z_j \pi/4)$, averaging exactly $(n+1)2^{-n}$), and an exact transposition differential with collision probability $2(5/8)^{m-1} - 2^{-(m-1)}$, all confirmed experimentally. Both operations are triangular, which yields a low-bit dissection algorithm with worst-case time $O^*(2^{n/2})$ and space $O^*(2^{n/4})$ plus an explicit balance parameter; no technique we know of improves the exponent $1/2$. We explain why the representation technique that recently broke the $2^{n/2}$ barrier for Subset-Sum is not available to the natural formulation of non-commutative chains, report that CDCL solvers scale as roughly $2^{0.7n}$ in our experiments, and prove in an ideal chain model, where each operation is an independent random permutation, that planted search requires $\Omega(2^{(n-1)/2})$ oracle queries. We place the problem in the current fine-grained landscape, in which the 3SUM, APSP and (strong) exponential time hypotheses have been refuted: the conditional lower bound that the 3-SAT reduction would otherwise yield is void, and the exponential complexity of Add/XNOR is open in both directions.

Keywords: NP-completeness, Add/XNOR, Boolean–arithmetic mixing, meet-in-the-middle, dissection, representation technique, generic lower bounds, fine-grained complexity

1 Introduction

Addition modulo 2^m and bitwise XOR are the two cheapest operations on machine words, and mixing them is the oldest way of destroying algebraic structure: it underlies ARX ciphers and most hash functions in use. This paper introduces the ADD/XNOR problem, which isolates the decision version of this mixing in its simplest form: a chain of n words is combined left to right, at each of the $n-1$ positions one may choose addition or XNOR, and the question is whether the result can be made zero. Every operand enters once, so the only freedom is the operator string, a certificate of $n-1$ bits for an input of n^2 bits.

Definition 1 (ADD/XNOR). *For an input parameter $n \geq 2$, let $M = 2^n$. Given an ordered sequence $A_1, \ldots, A_n \in \{0, \ldots, M-1\}$, each written in exactly n bits with leading zeros allowed, decide whether there exist $O_1, \ldots, O_{n-1} \in \{+, \odot\}$ such that*

$$E = (((A_1\, O_1\, A_2)\, O_2\, A_3) \cdots)\, O_{n-1}\, A_n = 0. \tag{1}$$

Here $+$ is addition modulo M, and $a \odot b = (M-1) \oplus a \oplus b$ is bitwise XNOR, where $\oplus$ denotes XOR. Evaluation is from left to right and all operators are free choices.

We call this the balanced version: n is both the operand count and the word width. The two-parameter version has $N \geq 2$ operands of $m \geq 1$ bits, modulus 2^m, and $q = N-1$ free operators. Bit positions are numbered from 0 at the least significant bit. Write $x_i = 0$ for addition and $x_i = 1$ for XNOR, so that an operator string is a word $x \in \{0,1\}^q$. The *state sequence* of x on $A = (A_1, \ldots, A_N)$ is $S_1 = A_1$ and $S_{i+1} = S_i + A_{i+1} \bmod 2^m$ if $x_i = 0$, $S_{i+1} = S_i \odot A_{i+1}$ if $x_i = 1$; we write $E_A(x) = S_N$, or $E(x)$, for the value of the expression. A general target in place of zero changes nothing below.

Observation 1 (Reachable states) *The two-parameter problem is decidable in $O(Nm2^m)$ bit operations; in particular the restriction $m = O(\log N)$ is in P.*

The proof, a dynamic program over reachable values, is given in Section 5.1; the bound is exponential in the word width, not in the input length. Fixing the first operator and merging the first two operands reduces search to decision with q oracle calls, so hardness may be stated for either.

1.1 Why Add/XNOR

Operator dichotomy. The hardness of the chain depends entirely on which two operations compete (Table 1). Binary choice is not the source of hardness, carries

Table 1. Chains with two competing operations on N words of m bits ($q = N - 1$ choices).

Operations	Structure	Status	Where
$\{\oplus, \odot\}$	closed form; parity only	$O(Nm)$	eq. (2)
$\{+, -\}$	signed sum, commutative	NP-complete; $O^*(2^{(1/2-\varepsilon)q})$ claimed	[20,37,29]
$\{+, \oplus\}$	carries, no complement	open	—
$\{+, \odot\}$	carries and complement, non-commutative	NP-complete; $O^*(2^{q/2})$	this paper

are. With $\{\oplus, \odot\}$ the value is

$$E = \bigoplus_{i=1}^{N} A_i \ \oplus \ \Big(\bigoplus_{i=1}^{q} x_i\Big)(2^m - 1), \tag{2}$$

so a zero output exists iff $\bigoplus_i A_i$ is zero or all ones. With $\{+, -\}$ the chain is $A_1 + \sum_{i \geq 2} \sigma_i A_i$ with free signs, i.e. Partition modulo 2^m: NP-complete but commutative, so it inherits every Subset-Sum algorithm, including the recent ones below $2^{q/2}$. With $\{+, \oplus\}$ our guard does not transfer (the mixed pattern `+X+X` on the operands of Lemma 5 returns the field to its initial value) and the status is open. Only $\{+, \odot\}$ combines a carry-propagating operation with a bitwise one so that no commutative structure remains.

Mixing, witnesses, and a non-commutative cousin. Addition and XNOR are both *triangular*: output bit j depends only on input bits $0, \ldots, j$. Such maps are the T-functions of Klimov and Shamir [21], and the linear [36] and differential [23] properties of addition are the standard tools of ARX cryptanalysis. An Add/XNOR chain is a keyed permutation with one key bit per round, and the decision problem is key recovery from the single pair $(A_1, 0)$, where the generic attack on a cascade of q keyed permutations is meet-in-the-middle at $2^{q/2}$; Add/XNOR asks whether the algebra of carries allows anything better, and Section 6 makes "generic" precise. (The chain is also a one-way function candidate of unusual simplicity; its average-case security is not analysed here.) A balanced instance has $n - 1$ free bits but a CNF encoding with $\Theta(n^2)$ variables, so, like Subset-Sum, Knapsack and TSP, its exponential parameter is the witness length, and the recent subexponential k-SAT algorithms give nothing for it. Among these problems it is the non-commutative cousin of Subset-Sum: partial solutions of Subset-Sum form an abelian group with a homomorphic key, whereas partial solutions of Add/XNOR are words acting on states—which is exactly what the recent Subset-Sum algorithms exploit (Section 5.3).

1.2 Results

1. **NP-completeness** (Section 3). Balanced ADD/XNOR is NP-complete (Theorem 1); hardness extends to every width $m = \lceil N^\delta \rceil$ with fixed rational $\delta > 0$ (Corollary 2), and 3-SAT with v variables and c clauses reduces to balanced ADD/XNOR with $n = O(v + c)$ (Theorem 2).
2. **Triangularity** (Section 2). Reduction modulo 2^p commutes with forward and backward evaluation (Lemma 3); a common power of two dividing the operands strips the width (Lemma 4).
3. **Exact structure** (Section 4). The chain is the XOR of its operands corrected by the operator parity and the accumulated carries (Lemma 6); for an operator string with addition runs $z_1, \ldots, z_r$ the bit-1 correlation with its linear model is exactly $\prod_j 2^{-z_j/2} \cos(z_j \pi/4)$ (Theorem 3), averaging exactly $(N+1)2^{-N}$ (Theorem 4); every bit position has an explicitly diagonalizable transfer operator, with bit 1 extremal (Theorem 5); and transposing an adjacent $(+, \odot)$ pair collides with probability exactly $2(5/8)^{m-1} - 2^{-(m-1)}$ (Theorem 6).
4. **Algorithms and the barrier** (Section 5). A low-bit dissection runs in worst-case time $O^*(2^{q/2})$ and space $O^*(2^{q/4} + \mu)$ for an explicit, efficiently computable balance parameter μ (Theorem 7). In the prefix/suffix formulation a solution has exactly one representation across every cut (Lemma 9), which is why the representation technique behind the recent SUBSET-SUM algorithms does not transfer in its standard form.
5. **A generic lower bound** (Section 6). In the ideal chain model, where each step's two options are independent random permutations, any T-query algorithm solves planted search with probability at most $3T/2^{\lceil q/2 \rceil} + 10T^2/2^m + 2^{-q}$ (Theorem 8): for $m \geq q$ this is $\Omega(2^{q/2})$ queries, so meet-in-the-middle is optimal among algorithms that only evaluate and invert steps.
6. **Experiments** (Section 7). Every closed form is validated, several exhaustively; solution counts are close to a Poisson law of mean $1/2$ with a measured over-dispersion; and CDCL solvers scale as roughly $2^{0.7n}$ over the measured range $n \leq 30$.

1.3 The current fine-grained landscape

The natural conditional statement to attach to Theorem 2 would be that, under the Exponential Time Hypothesis (ETH) [18], balanced ADD/XNOR has no $2^{o(n)}$-time algorithm, and the natural benchmark for Proposition 5 would be the $2^{n/2}$ barrier for SUBSET-SUM. Both have moved. Table 2 records the status, at the time of writing, of the hypotheses and barriers relevant to this paper.

Three points matter here. First, the three pillars of fine-grained complexity in the sense of [34]—SETH, 3SUM and APSP—have all recently been refuted: [3] gives polynomial improvements over the textbook algorithms for 3SUM and APSP, formalized in Lean [4], and the manuscript [27] counts the satisfying assignments of any k-CNF in subexponential time, refuting ETH and hence SETH;

Table 2. Hypotheses and barriers relevant to ADD/XNOR, status at the time of writing ("Lean": a machine-checked formalization has been released).

Hypothesis / barrier	Status	Verification
3SUM, $n^{2-o(1)}$	refuted: deterministic $O(n^{1.9992})$ [3]	Lean [4]
APSP, $n^{3-o(1)}$	refuted: deterministic $O(n^{2.9995})$ [3]	Lean [4]
ETH, #ETH	refuted: #k-SAT in time $\mathrm{poly}(n+m)\,2^{O_k(n \log\log n/\sqrt{\log n})}$, polynomial space (manuscript dated 25 Oct. 2026) [27]; independent manuscript [5]	Lean for a weaker bound [28]; external verification described in [26]
SETH, Gap-ETH, NSETH	refuted as consequences [27]	—
SUBSET-SUM, $2^{(1/2-\varepsilon)n}$	claimed: randomized word-RAM $O(2^{0.49n})$ [29]; independently, randomized $O(2^{(1/2-1/450000)n})$ for n integers of $\mathrm{poly}(n)$ bits [37, Thm. 4.1]	none listed
Orthogonal Vectors, $n^{2-o(1)}$	not refuted by the above (implied by SETH, not equivalent)	—
ADD/XNOR, $2^{(1/2-\varepsilon)q}$	open; no technique known (this paper)	—

for the latter we distinguish the manuscript's theorem, the weaker bound covered by the released formalization [28], and the external verification described in [26]. None of our theorems depends on this background. Second, the refutation of ETH does not transfer to ADD/XNOR: subexponential bounds are preserved by reductions with linear blow-up in the relevant parameter, superlinear blow-up must be assessed by composing the bounds, and the CNF encoding of a balanced instance has $\Theta(n^2)$ variables (Remark 1), so the new algorithms run in time $2^{O(n^2 \log\log n/\sqrt{\log n})}$ on it, far above $2^{n/2}$; here n is the operator parameter and $\Theta(n^2)$ the input length. The subexponential status of ADD/XNOR is open in both directions. Third, the comparison with SUBSET-SUM has reversed: the exponent 1/2 of Horowitz and Sahni [16] had resisted fifty years of attacks [11], and two recent preprints claim exponential improvements—a randomized word-RAM algorithm in time $O(2^{0.49n})$ [29] and, independently, time $O(2^{(1/2-1/450000)n})$ for n integers of polynomially many bits [37, Thm. 4.1], by the representation technique in the worst-case framework of [30] combined with a new partial-match algorithm. That mechanism relies on the additive structure of partial solutions, which the natural formulation of ADD/XNOR lacks (Section 5.3); this makes ADD/XNOR a test case for whether the meet-in-the-middle barrier survives without commutativity. We do not promote this into a named hypothesis—recent events are a sufficient warning about the half-life of such hypotheses—but state the barrier as an open question with the evidence we have.

Related work. Exact algorithms for Subset-Sum—meet-in-the-middle [16], the Schroeppel–Shamir tradeoff [31] and its Orthogonal-Vectors improvement [25], the representation technique for random instances [17,7,9] and its worst-case uses [24,30,6], lattice attacks on low-density instances [22,12], and the generalized birthday algorithm [35]—are surveyed in [37,30]. Dissection for compositions of permutations and for knapsacks is due to [14]. Structurally, Add/XNOR is a known-plaintext key-recovery problem for a cascade of q one-bit-keyed permutations: meet-in-the-middle on double encryption goes back to [13], lower bounds in the ideal cipher model to [2,15], and our ideal chain model follows the methodology of the generic group model [32]. In the pseudo-polynomial regime, the SETH-based lower bound of [1] that framed Bringmann's algorithm [10] is void after [27].

Conventions. Running times count bit operations on a deterministic multitape Turing machine unless stated otherwise; $O^*(\cdot)$ suppresses factors polynomial in N and m. For an m-bit word z, $\overline{z} = z \oplus \mathbf{1}$ is its complement, where $\mathbf{1} = 2^m - 1$; z_j is bit j of z; $\mathsf{wt}(x)$ is the Hamming weight of a bit string; $\rho_p : \mathbb{Z}_{2^m} \to \mathbb{Z}_{2^p}$ is reduction modulo 2^p, i.e. the low p bits. For a finite set S, $s \leftarrow S$ denotes uniform sampling, and a *uniform instance* is $A \leftarrow (\{0,1\}^m)^N$.

2 Preliminaries

Lemma 1 (Step maps). *For every $a \in \{0,1\}^m$, $\sigma_a : s \mapsto s + a$ and $\tau_a : s \mapsto s \odot a$ are permutations of $\{0,1\}^m$, with $\sigma_a^{-1}(t) = t - a$ and $\tau_a^{-1} = \tau_a$; for every fixed s, the maps $a \mapsto s + a$ and $a \mapsto s \odot a$ are permutations of $\{0,1\}^m$ as well.*

Proof. σ_a is translation in $\mathbb{Z}_{2^m}$ and $\tau_a(s) = s \oplus \bar{a}$ is translation in $\mathbb{F}_2^m$; bijectivity in a is symmetric. □

Hence the chain can be evaluated backward from any target, a unique last operand completes any prefix to any target, and $E_A(x)$ is uniform for every fixed x over a uniform instance. The model of Section 6 grants exactly this power and nothing more.

2.1 Normalization by XNOR parity

Lemma 2 (Normalization). *Let E be the current value of a left-associative computation on m-bit words and $\varepsilon \in \{0,1\}$ the parity of the number of XNORs performed so far. Define $Y = E$ if $\varepsilon = 0$ and $Y = \overline{E}$ if $\varepsilon = 1$. On processing the next operand a,*

$$(Y, \varepsilon) \longmapsto \begin{cases} (Y + (-1)^{\varepsilon} a \bmod 2^m,\ \varepsilon), & \text{for } +, \\ (Y \oplus a,\ \varepsilon \oplus 1), & \text{for } \odot\,. \end{cases} \tag{3}$$

Consequently, if 2^p divides a, the lowest p bits of Y are unchanged; a zero operand leaves Y unchanged.

Proof. For $\varepsilon = 1$ use $\overline{\overline{Y} + a} = Y - a \pmod{2^m}$; for XNOR, $E' = \overline{E \oplus a}$ gives $Y' = Y \oplus a$ with the parity flipped. The last assertion follows by reducing modulo 2^p. □

In normalized form a chain is a sequence of signed additions and XORs, and between two consecutive XNOR positions every operator is an addition with the same sign. Hence the value is determined by the set of XNOR positions $\{j_1 < \cdots < j_t\}$: writing Σ_s for the sum of the operands strictly between positions j_s and j_{s+1},

$$Y = \Big(\cdots \big((A_1 + \Sigma_0) \oplus A_{j_1+1} - \Sigma_1 \big) \oplus A_{j_2+1} + \Sigma_2 \cdots \Big). \tag{4}$$

Within a run of additions there is nothing to choose; the only choice is where the runs end.

2.2 Triangularity

Lemma 3 (Projection). *Let $p \le m$. For all $z, a \in \mathbb{Z}_{2^m}$, $\rho_p(z \pm a) = \rho_p(z) \pm \rho_p(a)$ and $\rho_p(z \odot a) = \rho_p(z) \odot_p \rho_p(a)$, with arithmetic modulo 2^p and $\odot_p$ XNOR on p-bit words. Consequently, for every instance I, every operator string x and every prefix length, the low p bits of the state equal the state of the projected instance $\rho_p(I) = (\rho_p(A_1), \ldots, \rho_p(A_N))$ under the same x; the same holds for backward evaluation.*

Proof. Carries and borrows propagate from low to high positions only, and XNOR is positionwise; induct on the prefix length. □

Corollary 1. *$E(x) \equiv 0 \pmod{2^p}$ if and only if x solves the width-p instance $\rho_p(I)$; the solution set of I is contained in that of $\rho_p(I)$ for every p. In particular $E(x) \bmod 2 = \bigoplus_{i=1}^{N} (A_i \bmod 2) \oplus \bigoplus_{i=1}^{q} x_i$, so every solution has the same XNOR parity.*

The projection lemma is the one place where ADD/XNOR is *not* a generic composition of permutations; Section 5.2 uses it for space.

Lemma 4 (Stripping a common power of two). *Let $s < m$ and suppose that 2^s divides each of $A_2, \ldots, A_N$. Write $A_i = 2^s A_i'$ for $i \ge 2$ and $A_1 = 2^s A_1' + c$ with $0 \le c < 2^s$, and let $I' = (A_1', \ldots, A_N')$, an instance of width $m - s$. Then I is a yes-instance if and only if $c = 0$ and I' has a solution with an even number of XNORs, or $c = 2^s - 1$ and I' has a solution with an odd number of XNORs.*

Proof. By Lemma 2 the low s bits of Y equal c throughout and no carry or borrow enters bit s, so the high bits of Y evolve as the normalized state of I'. The final value is Y or $\overline{Y}$ according to ε, and it is zero iff ($\varepsilon = 0$, $Y = 0$) or ($\varepsilon = 1$, $Y = 2^m - 1$); read off the low and high parts. □

A parity side condition is enforced by restricting enumerations to one parity class, so the effective width of an instance is $m - s$.

2.3 Boolean encoding

Remark 1 (Carry constraints). With $a_{i,j}$ bit j of A_i, $e_{i,j}$ bit j after operation i ($e_{0,j} = a_{1,j}$) and $c_{i,j}$ the carry into position j ($c_{i,0} = 0$), an exact encoding over $\mathbb{F}_2$ is

$$\begin{aligned} c_{i,j+1} &= (1 + x_i)\big(e_{i-1,j}a_{i+1,j} + e_{i-1,j}c_{i,j} + a_{i+1,j}c_{i,j}\big), \\ e_{i,j} &= e_{i-1,j} + a_{i+1,j} + x_i + (1 + x_i)c_{i,j}, \end{aligned} \tag{5}$$

for $1 \le i \le q$, $0 \le j < m$, with $e_{q,j} = 0$: a CNF with $O(Nm)$ auxiliary variables and clauses, i.e. $\Theta(n^2)$ variables for $n - 1$ free ones in the balanced case.

2.4 Random and planted instances

Proposition 1 (Solution counting). *On a uniform instance,* $\Pr[E(x) = 0] = 2^{-m}$ *for every fixed* $x \in \{0,1\}^q$*; the number* Z *of solutions has* $\mathbb{E}[Z] = 2^{q-m}$ *and* $\Pr[Z \ge 1] \le 2^{q-m}$*. In the balanced case* $\mathbb{E}[Z] = 1/2$*.*

Proof. Condition on $A_2, \dots, A_N$: $E(x)$ is a composition of permutations applied to the uniform word A_1 (Lemma 1). Sum over x and apply Markov's inequality. □

Proposition 1 gives neither a positive lower bound on $\Pr[Z \ge 1]$ nor a uniqueness statement: the values $\{E_A(x)\}_x$ are not independent (strings at Hamming distance one are never both solutions, by Corollary 1), and even a Poisson(1/2) law would give $\Pr[Z = 1 \mid Z \ge 1] \approx 0.77$. Experimentally Z is close to Poisson(2^{q-m}) at $m = n \le 14$, with a measurable over-dispersion (Table 5). The *planted* distribution samples $x^* \leftarrow \{0,1\}^q$ and $A_1, \dots, A_{N-1} \leftarrow \{0,1\}^m$ and chooses the unique A_N with $E_A(x^*) = 0$; decision is trivial for it, and the *planted search problem* asks for any x with $E_A(x) = 0$. It is size-biased by the number of solutions:

Proposition 2 (Size bias of the planted ensemble). *For every instance* A*,* $\Pr_{\text{plant}}[A] = 2^{m-q}\, Z(A)\, \Pr_{\text{unif}}[A]$.

Proof. A is produced exactly by the pairs $(x^*, (A_1, \dots, A_{N-1}))$ with $E_A(x^*) = 0$, each of probability $2^{-q}2^{-m(N-1)}$; there are $Z(A)$ of them, and $\Pr_{\text{unif}}[A] = 2^{-mN}$. □

Statements about one ensemble therefore do not transfer to the other without an explicit argument.

3 The Add/XNOR problem is NP-complete

Our source problem is binary Subset-Sum: for positive integers $s_1, \dots, s_r$ and a target T, decide whether $\sum_i y_i s_i = T$ for some $y \in \{0,1\}^r$; it is NP-complete [20]. The reduction is deterministic, polynomial-time and many-one, its output has exactly n operands of n bits, and every operator remains free. The main issue is soundness: four free operators admit sixteen patterns, and all but two must be excluded.

Table 3. All four-step patterns for the guard in Lemma 5. Here X denotes XNOR, and y is the normalized four-bit value.

Pattern	Final y	Pattern	Final y
`++++`	1	`X+++`	5
`+++X`	9	`X++X`	15
`++X+`	5	`X+X+`	3
`++XX`	15	`X+XX`	11
`+X++`	9	`XX++`	15
`+X+X`	11	`XX+X`	15
`+XX+`	7	`XXX+`	15
`+XXX`	7	`XXXX`	1

Theorem 1. *The* ADD/XNOR *decision problem of Definition 1 is NP-complete under deterministic polynomial-time many-one reductions.*

Throughout this section, $M = 2^n$.

Lemma 5 (Four-bit guard). *Consider the updates in* (3) *modulo* 16*, starting from* $(y, \varepsilon) = (1, 0)$*, with operands* $(3, 13, 3, 13)$ *in that order. The final value of* y *is* 1 *if and only if all four operators are additions or all four are XNORs. Both patterns end with* $\varepsilon = 0$*.*

Proof. The sixteen possibilities are listed in Table 3. Each entry is obtained using addition or subtraction modulo 16 according to the current parity, and XOR together with a parity flip for an XNOR operation. For example, the pattern `X+X+`, where X denotes XNOR, gives

$$(1, 0) \longmapsto (2, 1) \longmapsto (5, 1) \longmapsto (6, 0) \longmapsto (3, 0).$$

Only `++++` and `XXXX` return y to 1; their numbers of XNOR operations are zero and four, respectively. □

Proof (Theorem 1). **Membership in NP.** A certificate specifies the $n-1$ operators. Both modular addition and XNOR on n-bit words take $O(n)$ bit operations, so the expression can be verified in $O(n^2)$ time, polynomial in its input length.

Construction. Let $(s_1, \ldots, s_r; T)$ be a SUBSET-SUM instance with $r \geq 1$, positive integers s_i, and $0 \leq T \leq S := \sum_i s_i$; other inputs are trivial and are mapped to fixed two-bit yes- or no-instances.

Set

$$h = \lceil \log_2(S + 1) \rceil, \qquad p_i = 2 + 4(i - 1), \quad 1 \leq i \leq r, \tag{6}$$
$$K = 4r + 2, \qquad n = K + h + 1.$$

Define

$$\begin{aligned} C &= 1 + \sum_{i=1}^{r} 2^{p_i}, & P &= C + 2^{K+1}T, \\ U_i &= 3 \cdot 2^{p_i}, & V_i &= s_i 2^K - U_i, \\ M &= 2^n, & D &= M - P. \end{aligned} \tag{7}$$

The output sequence is

$$\left(C,\ U_1, V_1, U_1, V_1,\ \ldots,\ U_r, V_r, U_r, V_r,\quad \underbrace{0, \ldots, 0}_{h+1 \text{ operands}}\ ,\ D \right). \tag{8}$$

All its operands are represented by exactly n bits, allowing leading zeros, and every operator is unrestricted.

There are $1 + 4r + (h+1) + 1 = n$ operands. Moreover, $C < 2^K$, $S < 2^h$, and

$$0 < P \le C + 2^{K+1} S < 2^{K+h+1} = M.$$

Since $p_i \le K - 4$, we have $0 < U_i < 2^K$ and $0 < V_i = s_i 2^K - U_i < s_i 2^K < M$. Also $0 < D < M$. Thus (8) is a valid balanced instance.

The lowest two bits serve as control bits. For each i, bits $p_i, \ldots, p_i + 3$ form a separate guard field, initially equal to 1 in C. Within this field, the four operands of gadget i are $(3, 13, 3, 13)$, because

$$\frac{U_i}{2^{p_i}} = 3, \qquad \frac{V_i}{2^{p_i}} = s_i 2^{K - p_i} - 3 \equiv 13 \pmod{16}.$$

Completeness. Suppose that $I \subseteq \{1, \ldots, r\}$ satisfies $\sum_{i \in I} s_i = T$. For each $i \in I$, use four additions in gadget i, whose net effect is

$$E \longmapsto E + 2(U_i + V_i) = E + 2^{K+1} s_i \pmod{M}. \tag{9}$$

For $i \notin I$, use four XNORs. This is the identity map, since the four complements cancel and

$$E \oplus U_i \oplus V_i \oplus U_i \oplus V_i = E.$$

Use addition on every zero operand and on D. Immediately before D the value is $C + 2^{K+1}T = P$, so the final value is $P + D = M \equiv 0 \pmod{M}$.

Soundness: the last operation and the target state. Now consider an arbitrary operator sequence whose final value is zero. Before D, every operand after C is divisible by 4. Since $C \bmod 4 = 1$, addition leaves the two control bits unchanged, and XNOR complements them. Hence the actual state immediately before D is congruent to 1 or 2 modulo 4, according to whether the XNOR parity is even or odd.

As $D \bmod 4 = 3$, an XNOR with D preserves those two bits and cannot produce zero. The last operator must therefore be addition. Its success requires

the preceding state to be congruent to 1 modulo 4, so the preceding parity is even. Since $D = M - P$, the state immediately before D satisfies

$$\varepsilon = 0, \qquad E = Y = P. \tag{10}$$

In particular, every normalized guard field must have final value 1.

Soundness: gadgets are forced. For $j > i$, both U_j and V_j are divisible by 2^{p_j} with $p_j \geq p_i + 4$, so by Lemma 2 all later gadgets and all zero operands preserve the lowest $p_i + 4$ bits of Y: after gadget i its normalized guard field cannot change before D. (This concerns Y, not E; later XNORs do complement the actual field, which is why normalization is needed.) We induct on i. Before gadget 1, $\varepsilon = 0$ and its guard field is 1; the operands of the gadget have their lowest p_1 bits zero, so no carry or borrow enters the field and Lemma 5 applies; since the field must end at 1 by (10) and cannot be repaired, gadget 1 is `++++` or `XXXX`. If gadgets $1, \ldots, i-1$ have these patterns, each added a multiple of 2^{K+1} or acted as the identity, with an even number of XNORs, so before gadget i the parity is zero and the lowest K bits still equal those of C; the same argument forces gadget i.

Soundness: recovery of a subset. Let I consist of the gadgets using four additions. Zero operands preserve Y, so the normalized value before D is

$$Y \equiv C + 2^{K+1} \sum_{i \in I} s_i \pmod{M}.$$

Combining this with (10) yields

$$2^{K+1} \left(\sum_{i \in I} s_i - T \right) \equiv 0 \pmod{2^{K+h+1}},$$

and therefore

$$\sum_{i \in I} s_i - T \equiv 0 \pmod{2^h}.$$

Both the subset sum and T lie in $[0, S]$, with $S < 2^h$. Their difference has absolute value less than 2^h, so it must be zero. Thus I solves the original Subset-Sum instance; the zero padding introduces no false positives, since it preserves Y and the control bits enforce even parity before D.

Polynomial size and running time. Let L be the binary encoding length of the Subset-Sum input. Then $r = O(L)$ and $h = O(\max_i \text{bitlength}(s_i) + \log r) = O(L)$. Consequently, $n = 4r + h + 3 = O(L)$ and the output length is $\Theta(n^2) = O(L^2)$. The constants are computed using additions, subtractions, and binary shifts on $O(L)$-bit words, so the reduction runs in polynomial time. Completeness and soundness establish

$$\text{Subset-Sum} \leq_m^p \text{Add/XNOR}.$$

Together with membership in NP, this proves the theorem. □

Remark 2 (Solution multiplicity). For a nontrivial source instance used in the construction, let Z_{SS} count indexed subsets with sum T, and let Z_{AX} count satisfying operator strings of (8). Then

$$Z_{\mathrm{AX}} = 2^h Z_{\mathrm{SS}}.$$

Indeed, soundness uniquely fixes every gadget from the chosen subset and forces the final addition. Among the $h+1$ zero operands, exactly 2^h operator strings have even XNOR parity; each preserves Y and leaves the actual state equal to P. Every successful string arises in this way. The reduction is therefore not parsimonious: 2^h is a known instance-dependent factor, not a universal constant. In particular, a unique source solution does not yield a unique target solution.

3.1 Varying operand count and word width

Corollary 2 (Polynomial relations between the two parameters). *For every fixed positive rational δ, the two-parameter* ADD/XNOR *problem restricted to $m = \lceil N^\delta \rceil$ is NP-complete.*

Proof. Write $\delta = a/b$ with fixed positive integers a, b. Given a SUBSET-SUM instance as above, put $t = 4r + h + 3$, $N = t^b$ and $m = t^a$, so that $m = N^\delta$ and $N, m \geq t$. Keep K, C, P, U_i, V_i from the construction, replace the modulus by 2^m, set $D = 2^m - P$, and place $Q = N - 4r - 2 \geq h + 1$ zero operands before D. The guard and control-bit arguments are unchanged: in any successful evaluation each gadget has one of its two forced patterns, the zero segment preserves Y with even parity, and the final operator is addition, whence $\sum_{i \in I} s_i - T \equiv 0 \pmod{2^{m-K-1}}$ with $m - K - 1 \geq h$, which forces equality; conversely a source solution works with additions throughout. The output has $Nm = t^{a+b}$ bits, polynomial in the source length since a, b are fixed. □

Together with Observation 1, this separates logarithmic word widths from every fixed positive rational power; for $m = O((\log N)^k)$ with $k > 1$ the dynamic program is quasipolynomial, and the intermediate regimes are not classified here.

3.2 A linear-parameter reduction from 3-SAT

Theorem 2. *There is a deterministic polynomial-time reduction that maps a 3-CNF formula with v variables and c clauses to an equivalent balanced* ADD/XNOR *instance with $n = O(v + c)$. Consequently, for every $\eta > 0$ there is a constant C_η such that 3-SAT on v variables reduces, in time $O^*(2^{\eta v})$, to the disjunction of at most $2^{\eta v}$ balanced* ADD/XNOR *instances with $n \leq C_\eta v$ each.*

Proof. Remove repeated literals and tautological clauses. The usual digit construction of SUBSET-SUM has $r = 2v + 2c$ positive integers with $v + c$ decimal positions. For each variable use two numbers, one for each truth value: both have digit 1 in that variable's position, and digit 1 in exactly those clause positions

satisfied by that literal. For each clause add two slack numbers having respectively digits 1 and 2 in that clause position and zeros elsewhere. The target has variable digits 1 and clause digits 4.

No column sum can exceed 6, even if every number is selected, so there are no decimal carries. Variable columns force one truth value per variable. A clause with $k \in \{0, 1, 2, 3\}$ selected true literals can reach digit 4 with the slack numbers exactly when $k \geq 1$. Thus the formula is satisfiable exactly when the Subset-Sum instance is satisfiable. Its sum S has $O(v + c)$ binary bits, whence $r = O(v + c)$ and $h = O(v + c)$, and Theorem 1 produces an equivalent Add/XNOR instance with $n = 4r+h+3 = O(v+c)$. For the second statement, the sparsification lemma [19] expresses an arbitrary 3-CNF as a disjunction of at most $2^{\eta v}$ formulas with at most $c_\eta v$ clauses each, in $O^*(2^{\eta v})$ time; apply the first statement to each. □

Remark 3. Under ETH, Theorem 2 would exclude $2^{o(n)}$-time algorithms for balanced Add/XNOR; with ETH refuted [27] the implication is vacuous. What survives is that Add/XNOR is at least as hard as 3-SAT in the subexponential sense (a $2^{o(n)}$ algorithm for Add/XNOR would refute ETH independently), while the converse offers no help, since the CNF encoding of Add/XNOR has $\Theta(n^2)$ variables (Remark 1). Whether balanced Add/XNOR admits a $2^{o(n)}$ algorithm is open in both directions, as it now is for Subset-Sum.

The obstruction is mechanistic as well as quantitative. The algorithm of [27] writes the count of satisfying assignments as a sum of clause-factor products and arranges, through parameter-dependent replacements of the variables, that the variables fall into small groups coupled only through a few shared integer linear measurements, so that the sum factorizes group by group and the exact count is recovered by interpolation; its exponent is measured in the number of CNF variables, and the method needs the total degree of the clause product to be linear in that number. In an Add/XNOR chain the operator variables of two consecutive blocks interact only through the carries, that is, through the full m-bit intermediate state, so the analogue of a measurement between blocks is the state itself. Grouping the chain with full-state measurements recovers the reachable-state dynamic program of Section 5.1, and grouping it into two blocks with one measurement recovers meet-in-the-middle; triangularity lets the measurement shrink to its low bits, which is the dissection of Section 5.2 and saves space, not time. The same holds for width-based methods: the natural CNF, read along the chain, has pathwidth $O(m)$, and dynamic programming over that decomposition is again the $2^{O(m)}$ reachable-state algorithm. The feature that protects Add/XNOR from the refutation is thus the one this paper is about: its q free variables are coupled by an m-bit carry state, not by sparse local constraints.

4 Linear and differential structure of the chain

This section isolates where the hardness of the chain lives and gives exact closed forms for its dominant linear and differential behaviour, in the spirit of the ARX

analyses of [36,23]. Every formula is confirmed in Section 7, several exhaustively. Throughout, $x \in \{0,1\}^q$ has state sequence $S_1, \ldots, S_N$ on A and $E_A(x) = S_N$.

4.1 The carry decomposition

We next isolate where the hardness lives. For an addition step define its *carry word* $C := (s + a) \oplus s \oplus a$; note $C_0 = 0$ always, and $C_{j+1} = \mathrm{maj}(s_j, a_j, C_j)$ is the familiar carry recursion.

Lemma 6 (Carry decomposition). *Fix A and x, and let $S_1, \ldots, S_N$ be the state sequence of Section 1. For each i with $x_i = 0$ let $C^{(i)} := S_{i+1} \oplus S_i \oplus A_{i+1}$ be the carry word of step i. Then*

$$E_A(x) \;=\; \bigoplus_{i=1}^{N} A_i \;\oplus\; \big(\mathsf{wt}(x) \bmod 2\big) \cdot \mathbf{1} \;\oplus \bigoplus_{i:\, x_i = 0} C^{(i)}.$$

Proof. Each addition step satisfies $S_{i+1} = S_i \oplus A_{i+1} \oplus C^{(i)}$ by definition of $C^{(i)}$, and each XNOR step satisfies $S_{i+1} = S_i \oplus A_{i+1} \oplus \mathbf{1}$. Composing from $S_1 = A_1$ and collecting terms, every A_i appears exactly once, every XNOR step contributes one $\mathbf{1}$, and every addition step contributes its carry word. □

Since the XOR part is trivial to control, all of the computational content of the problem resides in the carry accumulation $\bigoplus_{i:x_i=0} C^{(i)}$. One consequence is immediate; the rest of this section quantifies the carries.

Corollary 3 (Parity invariant). *$E_A(x)_0 = \bigoplus_{i=1}^{N} A_{i,0} \oplus (\mathsf{wt}(x) \bmod 2)$, as in Corollary 1. Hence: (i) every solution of an instance A satisfies $\mathsf{wt}(x) \equiv \beta(A) := \bigoplus_i A_{i,0} \pmod 2$; (ii) any two solutions of the same instance lie at even Hamming distance; (iii) any algorithm may restrict its search to one parity class, a speedup of exactly one bit.*

Proof. Carry words have zero LSB, so bit 0 of Lemma 6 reads as stated. (ii): if $\mathsf{wt}(x) \equiv \mathsf{wt}(x')$ then $\mathsf{wt}(x \oplus x') = \mathsf{wt}(x) + \mathsf{wt}(x') - 2\mathsf{wt}(x \wedge x')$ is even. □

4.2 Exact linear correlations of the chain

Linear cryptanalysis asks how well $E_A(x)$ is approximated by an affine function of the inputs. Bit 0 is *exactly* affine (Corollary 3); the first nontrivial position is bit 1, whose natural linear model, suggested by Lemma 6 with carries dropped, is

$$\ell_A(x) \;:=\; \bigoplus_{i=1}^{N} A_{i,1} \;\oplus\; \big(\mathsf{wt}(x) \bmod 2\big).$$

Define the correlation of a fixed operator string x as

$$\mathsf{Cor}(x) \;:=\; \mathbb{E}_{A \leftarrow (\{0,1\}^m)^N}\Big[(-1)^{\,E_A(x)_1 \,\oplus\, \ell_A(x)}\Big].$$

By Lemma 6, $E_A(x)_1 \oplus \ell_A(x) = \bigoplus_{i:x_i=0} C_1^{(i)}$, the accumulated carry parity at position 1, so $\mathsf{Cor}(x)$ quantifies exactly the visibility of the carries at the lowest nonlinear position. The structure of x that matters is its *run profile*: let $z_1, \ldots, z_r \geq 1$ be the lengths of the maximal runs of 0s (addition runs) in x, read left to right.

Theorem 3 (Run-product formula). *For all $N \geq 2$, $m \geq 2$, and every $x \in \{0,1\}^q$ with addition-run profile $(z_1, \ldots, z_r)$,*

$$\mathsf{Cor}(x) \;=\; \prod_{j=1}^{r} c(z_j), \qquad \textit{where} \qquad c(z) \;=\; 2^{-z/2} \cos\frac{z\pi}{4}.$$

In particular $|c(z)| \leq 2^{-z/2}$, $c(z) = 0$ iff $z \equiv 2 \pmod 4$, and $|\mathsf{Cor}(x)| \leq 2^{-\frac{1}{2}\sum_j z_j} = 2^{-\mathsf{wt}(\bar{x})/2}$: the correlation decays exponentially in the total number of additions, at a rate of exactly half a bit per addition.

Proof. Step 1: reduction to LSB counts. Consider addition run j, entered in state $u^{(j)}$ and consuming operands $B_1^{(j)}, \ldots, B_{z_j}^{(j)}$ (consecutive elements of A). Modular addition is associative, so the run computes $S_{\text{out}} = u^{(j)} + \sum_k B_k^{(j)} \bmod 2^m$, while the per-step carry words XOR-telescope: $\bigoplus_{\text{steps in run } j} C = S_{\text{out}} \oplus u^{(j)} \oplus \bigoplus_k B_k^{(j)}$. Taking bit 1 and using the schoolbook identity for multi-operand addition—bit 1 of an integer sum equals the XOR of the operands' bits 1, XORed with the carry $\lfloor L/2 \rfloor \bmod 2$ produced at position 0, where L is the integer sum of the operands' LSBs—we get

$$T_j \;:=\; \bigoplus_{\text{steps in run } j} C_1 \;=\; \Big\lfloor L_j/2 \Big\rfloor \bmod 2, \qquad L_j \;=\; u_0^{(j)} + \sum_{k=1}^{z_j} \big(B_k^{(j)}\big)_0,$$

and $E_A(x)_1 \oplus \ell_A(x) = \bigoplus_{j=1}^{r} T_j$. (XNOR steps contribute no carries; their effect on bit 1 is affine and already absorbed into ℓ_A.)

Step 2: independence across runs. The operand LSBs $(B_k^{(j)})_0$ are LSBs of distinct inputs, hence i.i.d. uniform. For the entering bits, designate $D_1 := A_1$ and, for $j \geq 2$, let D_j be the last separator element between runs $j-1$ and j (at least one XNOR step separates maximal runs). Since XNOR and XOR steps act on the LSB as $u \mapsto u \oplus a_0 \oplus 1$, we have $u_0^{(j)} = (D_j)_0 \oplus g_j$, where g_j is a function of input LSBs of index strictly smaller than that of D_j, i.e., of run operands of runs $< j$, earlier separators, and constants. The elements $D_1, \ldots, D_r$ are distinct from each other and from all run operands, so conditioned on all run-operand LSBs, the map $((D_j)_0)_j \mapsto (u_0^{(j)})_j$ is a bijection (triangular with unit diagonal), making $u_0^{(1)}, \ldots, u_0^{(r)}$ i.i.d. uniform and independent of all run-operand LSBs. Consequently the triples defining $T_1, \ldots, T_r$ are mutually independent, and $\mathsf{Cor}(x) = \mathbb{E}[(-1)^{\oplus_j T_j}] = \prod_j \mathbb{E}[(-1)^{T_j}]$.

Step 3: one run. For a run of length z, L is a sum of $N := z+1$ i.i.d. uniform bits, so with $\omega_k := (-1)^{\lfloor k/2 \rfloor}$,

$$\mathbb{E}\big[(-1)^T\big] = 2^{-N} \sum_{k=0}^{N} \binom{N}{k} \omega_k.$$

Since $\omega_k = \mathrm{Re}\big((1-i)\, i^k\big)$, the sum equals

$$\mathrm{Re}\big((1-i)(1+i)^N\big) = \mathrm{Re}\big(2^{(N+1)/2} e^{i(N-1)\pi/4}\big) = 2^{(N+1)/2} \cos \tfrac{(N-1)\pi}{4},$$

giving $\mathbb{E}[(-1)^T] = 2^{(1-N)/2} \cos \frac{(N-1)\pi}{4} = 2^{-z/2} \cos \frac{z\pi}{4} = c(z)$. □

The carry signal at bit 1 thus decays at exactly half a bit of correlation per addition, oscillating and vanishing on runs of length 2 mod 4. Averaging over the operator string gives a second exact identity.

Theorem 4 (Exact average correlation). *For every $N \geq 2$ and $m \geq 2$,*

$$\mathbb{E}_{x \leftarrow \{0,1\}^q} \big[\mathsf{Cor}(x)\big] \;=\; (N+1)\, 2^{-N}.$$

Proof. Average first over A and then over x step by step. Track the joint evolution of $u \in \{0,1\}$ (the LSB of the running state) and $t \in \{0,1\}$ (the accumulated bit-1 carry parity) under one chain step with a uniform operator bit b and a fresh uniform operand LSB a: if $b = 1$ then $(u,t) \mapsto (u \oplus a \oplus 1,\ t)$; if $b = 0$ then $(u,t) \mapsto (u \oplus a,\ t \oplus (u \wedge a))$. Encode the state as the signed measure $\sigma(u) := \Pr[u, t{=}0] - \Pr[u, t{=}1]$, so that the desired average correlation is $\sigma(0) + \sigma(1)$ after q steps. Averaging the update over $(b,a) \in \{0,1\}^2$ gives the linear map

$$\begin{pmatrix} \sigma'(0) \\ \sigma'(1) \end{pmatrix} = M \begin{pmatrix} \sigma(0) \\ \sigma(1) \end{pmatrix}, \qquad M = \begin{pmatrix} 1/2 & 0 \\ 1/2 & 1/2 \end{pmatrix}.$$

(Only the addition branch with $u = 1, a = 1$ flips the sign, which yields the stated entries.) $M = \frac{1}{2}(I + N)$ with $N^2 = 0$, so $M^k = 2^{-k}(I + kN)$. The initial state has $t = 0$ and $u = A_{1,0}$ uniform, i.e. $(\sigma(0), \sigma(1)) = (\frac{1}{2}, \frac{1}{2})$, whence after $k = q$ steps $\sigma(0) + \sigma(1) = 2^{-k}\big(1 + \frac{k}{2}\big) = (N+1)2^{-N}$. □

The transfer matrix is defective (a Jordan block at $1/2$), which is why the average is $\Theta(N)2^{-N}$ rather than geometric. Both theorems are confirmed to within sampling error (Table 4). Three limits should be kept in mind: these are signed averages under uniform operands, which do not control absolute correlations, the best linear mask, or algorithms combining several constraints; they concern one explicit family of approximations; and they are statements about the uniform ensemble—the planted distribution is size-biased (Proposition 2), and already at $N = m = 2$ the same statistic is $3/4$ under uniform (A, x) but $1/2$ when A_2 is forced. We present the identities as structural results and draw no distinguishing or pruning bound from them.

4.3 Beyond bit 1: the spectrum of single-bit approximations

The higher bits are governed by a transfer operator whose spectrum admits a closed form. Fix $1 \le j \le m-1$, write $N := 2^j$, and for a run of z additions with independent uniform operands applied to a uniform state let $c_j(z) := \mathbb{E}[(-1)^{\kappa_1 \oplus \cdots \oplus \kappa_z}]$, where κ_t is the carry into bit j at the t-th addition—the correlation of bit j of the run's output with the XOR of bit j of its inputs.

Theorem 5 (Spectrum of single-bit approximations). *Let M_j be the $N \times N$ matrix, $N = 2^j$, with entries $M_j[s', s] = (-1)^{[s'<s]}$ for $s, s' \in \{0, \ldots, N-1\}$, and let*

$$\lambda_k \;=\; 1 - i\cot\theta_k, \qquad \theta_k = \frac{(2k+1)\pi}{2^{\,j+1}}, \qquad k = 0, \ldots, N-1.$$

Then:

(i) For any operator string x with addition runs $z_1, \ldots, z_r$, the correlation of bit j of $E_A(x)$ with $\bigoplus_i A_{i,j} \oplus (\mathsf{wt}(x) \bmod 2)$ equals $\prod_t c_j(z_t)$.

(ii) M_j is diagonalizable with eigenvalues $\lambda_0, \ldots, \lambda_{N-1}$, and for every $z \ge 0$

$$c_j(z) \;=\; 2^{-j(z+2)} \sum_{k=0}^{N-1} |\lambda_k|^2 \, \lambda_k^z.$$

(iii) With $\rho_j := 1/\big(2^j \sin(\pi/2^{j+1})\big)$, $|c_j(z)| \le \rho_j^z$ for all $z \ge 0$.

(iv) $\rho_1 = 2^{-1/2} > \rho_2 > \rho_3 > \cdots$ with $\lim_j \rho_j = 2/\pi$; hence $|c_j(z)| \le 2^{-z/2}$ for every $j \ge 1$ and $z \ge 0$, with equality of the decay rate only at $j = 1$, where $|c_1(z)| = 2^{-z/2}$ at every $z \equiv 0 \pmod 4$.

Proof. (i) As in Theorem 3: bit j of an XNOR step is $s_j \oplus a_j \oplus 1$, exactly linear, and a fresh uniform XNOR operand maps the low j bits of the state to a uniform value independent of everything prior, so the carry processes of distinct runs are independent and the correlations multiply.

(ii) The carry into bit j of $s + a$ depends only on $\sigma := s \bmod 2^j$ and $\alpha := a \bmod 2^j$, namely $\kappa = [\sigma + \alpha \ge 2^j]$; writing $\sigma' := (\sigma + \alpha) \bmod 2^j$ for the new low state, $\kappa = [\sigma' < \sigma]$, and as α ranges uniformly over $\{0, \ldots, N-1\}$, so does σ'. For the signed measure $\mu(\sigma) := \Pr[\sigma, \text{parity } 0] - \Pr[\sigma, \text{parity } 1]$, one addition acts as $\mu \mapsto 2^{-j} M_j \mu$, so with $\mu_0 = N^{-1}\mathbf{1}$ we get $c_j(z) = 2^{-jz} N^{-1}\, \mathbf{1}^\top M_j^z \mathbf{1}$, which we compute spectrally. For $r \in \mathbb{C}$ with $r^N = -1$, the geometric vector $v(r)_s := r^s$ satisfies

$$(M_j v)_{s'} \;=\; \sum_{s \le s'} r^s - \sum_{s > s'} r^s \;=\; \frac{2r^{s'+1} - 2 - (r^N - 1)}{r - 1} \;=\; \frac{2r}{r-1}\, r^{s'},$$

because $r^N - 1 = -2$. Hence each of the N distinct roots $r_k = e^{i(2k+1)\pi/N}$ yields an eigenvector with eigenvalue $\lambda_k = 2r_k/(r_k - 1) = 1 - i\cot(\theta_k)$ where $\theta_k =$

$\frac{1}{2}\arg r_k = (2k+1)\pi/(2N)$; as the λ_k are distinct, M_j is diagonalizable. Expanding $\mathbf{1} = \sum_k \beta_k v(r_k)$ by discrete Fourier interpolation gives $\beta_k = 2/\big(N(1-r_k^{-1})\big)$, and $\mathbf{1}^\top v(r_k) = (r_k^N - 1)/(r_k - 1) = -2/(r_k - 1)$, so

$$\beta_k \cdot \mathbf{1}^\top v(r_k) = \frac{-4}{N(1-r_k^{-1})(r_k-1)} = \frac{2}{N(1-\cos\arg r_k)} = \frac{1}{N\sin^2\theta_k} = \frac{|\lambda_k|^2}{N},$$

using $|\lambda_k|^2 = 1 + \cot^2\theta_k = 1/\sin^2\theta_k$. Therefore

$$\mathbf{1}^\top M_j^z \mathbf{1} = N^{-1}\sum_k |\lambda_k|^2 \lambda_k^z \qquad \text{and} \qquad c_j(z) = 2^{-j(z+2)}\sum_k |\lambda_k|^2\lambda_k^z$$

as claimed.

(iii) Setting $z = 0$ in (ii) and using $c_j(0) = 1$ yields the identity $\sum_k |\lambda_k|^2 = 4^j$ (equivalently, $\sum_k \cot^2\theta_k = N(N-1)$). Hence

$$|c_j(z)| \le 2^{-j(z+2)}\Big(\max_k |\lambda_k|\Big)^z \sum_k |\lambda_k|^2 = 2^{-j(z+2)}\big(2^j\rho_j\big)^z 4^j = \rho_j^z,$$

since $\max_k |\lambda_k| = 1/\sin\theta_0 = 2^j\rho_j$.

(iv) With $x := 2^j$ and $u := \pi/(2x) \in (0, \pi/4]$, one has $1/\rho_j = x\sin(\pi/(2x))$ and $\frac{d}{dx}\big[x\sin(\pi/(2x))\big] = \sin u - u\cos u > 0$ because $\tan u > u$ on $(0,\pi/2)$; so ρ_j is strictly decreasing in j, with $\rho_1 = 1/(2\sin(\pi/4)) = 2^{-1/2}$ and $x\sin(\pi/(2x)) \to \pi/2$, i.e., $\rho_j \downarrow 2/\pi$. Then $|c_j(z)| \le \rho_j^z \le \rho_1^z = 2^{-z/2}$ for every $j \ge 1$. For $j = 1$, (ii) reduces to the formula of Theorem 3: $\lambda_{0,1} = 1 \mp i$, so $c_1(z) = 2^{-(z+2)}\cdot 2\,\big(2\,\mathrm{Re}(1-i)^z\big) = 2^{-z/2}\cos(z\pi/4)$, whose absolute value equals $2^{-z/2}$ at $z \equiv 0 \pmod 4$. □

Thus, within this family, every bit's correlation decays at least as fast as $2^{-1/2}$ per addition, bit 1 is the unique extremal position, and no bit decays slower than $2/\pi$ per addition. The dominance is asymptotic, not run-by-run (at $z = 2$, $c_1(2) = 0$ while $c_2(2) = -\frac{1}{4}$), and averaged over operator strings the higher bits are far weaker (at $N = 14$: 9.2×10^{-4}, 2.5×10^{-6}, 2.6×10^{-8} for bits $1, 2, 3$). All statements were verified numerically (Section 7).

4.4 Differential structure and solution-pair geometry

Differential analysis asks how $E_A(x)$ and $E_A(x')$ relate for nearby operator strings; since all step maps are bijections sharing operands, this is governed by how XOR differences propagate.

Lemma 7 (Difference propagation). *Let $T, A, \delta \in \{0,1\}^m$ with T, A uniform and independent.*

(i) XNOR steps propagate XOR differences unchanged: $(T\oplus\delta)\odot A = (T\odot A)\oplus\delta$ always.

(ii) The MSB difference commutes with addition: $(T\oplus 2^{m-1}) + A = \big((T+A)\big)\oplus 2^{m-1}$ always (addition of 2^{m-1} modulo 2^m is XOR by 2^{m-1}).

(iii) For general δ,

$$\Pr_{T,A}\Big[(T \oplus \delta) + A = (T + A) \oplus \delta\Big] \;=\; 2^{-\mathsf{wt}(\delta \wedge (2^{m-1}-1))},$$

i.e., each active bit of δ below the MSB independently halves the propagation probability, and the MSB is free.

Proof. (i) is immediate from $s \odot a = s \oplus a \oplus \mathbf{1}$. (ii) holds because $2^{m-1} \equiv -2^{m-1} \pmod{2^m}$, so adding 2^{m-1} flips the MSB and nothing else. For (iii), write $C := C(T, A)$ and $C' := C(T \oplus \delta, A)$ for the two carry words; the event is $C' = C$. Compare the recursions $c_{k+1} = \mathrm{maj}(T_k, A_k, c_k)$ and $c'_{k+1} = \mathrm{maj}(T_k \oplus \delta_k, A_k, c'_k)$ upward from $c_0 = c'_0 = 0$, maintaining $c'_k = c_k$: at a bit with $\delta_k = 0$ equality is preserved automatically; at a bit with $\delta_k = 1$ and $k < m-1$, $\mathrm{maj}(\overline{T_k}, A_k, c_k) = \mathrm{maj}(T_k, A_k, c_k)$ holds iff $A_k = c_k$ (in which case both majorities equal A_k), an event of probability exactly $\frac{1}{2}$ since A_k is uniform and independent of c_k (which is determined by lower-indexed bits); at $k = m-1$ the carry out of the word is discarded, so nothing is required. Multiplying the conditional probabilities over active bits below the MSB gives the claim. For the general theory of XOR differentials through modular addition see [23]; the identical-operand case is the one the chain needs. □

Proposition 3 (Operator-flip differences). *Flipping a single operator bit x_i changes the state after step i by the XOR difference*

$$D_i \;=\; (S + A) \oplus (S \odot A) \;=\; C(S, A) \oplus \mathbf{1},$$

the complemented carry word of that step. Since carry words have LSB 0, $D_i \neq 0$ and no two solutions are at Hamming distance 1, and $\mathbb{E}[\mathsf{wt}(D_i)] = \frac{m+2}{2} - 2^{-m}$. (A difference of weight w is preserved by a subsequent addition with probability 2^{-w} by Lemma 7(iii); the preservation probability of D_i is $\mathbb{E}[2^{-\mathsf{wt}(D_i)}]$, not $2^{-\mathbb{E}\mathsf{wt}(D_i)}$, and we do not evaluate it.)

Proof. $S + A = S \oplus A \oplus C$ and $S \odot A = S \oplus A \oplus \mathbf{1}$; XOR the two. $\mathbb{E}[\mathsf{wt}(C)] = \sum_{k=1}^{m-1} \Pr[C_k = 1] = \sum_{k=1}^{m-1} (\frac{1}{2} - 2^{-k-1}) = \frac{m-2}{2} + 2^{-m}$, and $\mathsf{wt}(D) = m - \mathsf{wt}(C)$. □

Since single flips never cancel, solution *pairs* are governed by double flips, and here the chain has an exactly solvable differential: transposing an adjacent $(+, \odot)$ pair of operators.

Theorem 6 (Transposition differential). *Fix any position $1 \le p \le q-1$ and any $x \in \{0,1\}^q$ with $(x_p, x_{p+1}) = (0, 1)$, and let x' agree with x everywhere except $(x'_p, x'_{p+1}) = (1, 0)$. Then, over uniform A,*

$$\Pr\big[E_A(x) = E_A(x')\big] \;=\; 2\left(\frac{5}{8}\right)^{m-1} \;-\; 2^{-(m-1)}.$$

The same holds for any common prefix and suffix; the collision probability is position-independent.

Proof. Let T be the state after the common prefix (T is uniform: it is a bijective image of A_1; and it is independent of the two window operands $A := A_{p+1}$, $B := A_{p+2}$). Because the common suffix applies the same bijections to both chains, $E_A(x) = E_A(x')$ iff the states agree at the window exit, i.e., iff $(T + A) \odot B = (T \odot A) + B$. Writing $C := C(T, A)$, we have $(T+A)\odot B = T \oplus A \oplus C \oplus B \oplus \mathbf{1}$ and $(T \odot A) + B = T \oplus A \oplus \mathbf{1} \oplus B \oplus C'$ with $C' := C(T \odot A,\, B)$; the exit states agree iff $C' = C$. Condition on (T, A), so that C and $V := T \odot A$ are fixed, and count the B with $C(V, B) = C$ via the carry recursion $c_{k+1} = \mathrm{maj}(V_k, B_k, c_k)$ with all c_k forced to C_k: at bit $k \le m-2$, if $V_k = C_k$ the majority ignores B_k and the required next carry must equal V_k (two choices of B_k if $C_{k+1} = V_k$, none otherwise); if $V_k \neq C_k$ the majority equals B_k, forcing one choice. Bit $m-1$ is free. Averaging over (T_k, A_k) per bit turns this into a weighted two-state chain over $c_k \in \{0, 1\}$: from $c = 0$, the operand bits $(0,1), (1,0)$ give $V_k = 0 = c$ with consistent next carry 0 (weight 1 each) and $(0,0)$ gives $V_k = 1 \neq c$ (weight $\frac{1}{2}$, next carry 0), totalling $W_{0\to0} = \frac{5}{8}$, while $(1,1)$ gives weight $\frac{1}{2}$ with next carry 1: $W_{0\to1} = \frac{1}{8}$; from $c = 1$, $(0,1), (1,0)$ give $V_k = 0 \neq c$ (weight $\frac{1}{2}$, next carry 1) and $(1,1)$ gives $V_k = 1 = c$ with consistent next carry 1 (weight 1): $W_{1\to1} = \frac{1}{2}$, and $(0,0)$ is infeasible: $W_{1\to0} = 0$. With $W = \left(\begin{smallmatrix} 5/8 & 0 \\ 1/8 & 1/2 \end{smallmatrix}\right)$ lower triangular and $c_0 = 0$, the total mass after the $m-1$ constrained bits is

$$\Pr \;=\; \mathbf{1}^{\top} W^{\,m-1} e_0 \;=\; \left(\tfrac{5}{8}\right)^{m-1} + \left[\left(\tfrac{5}{8}\right)^{m-1} - \left(\tfrac{1}{2}\right)^{m-1}\right] \;=\; 2\left(\tfrac{5}{8}\right)^{m-1} - 2^{-(m-1)}.$$

□

Transposition pairs thus collide with probability $\Theta((5/8)^m)$, an enhancement of $\approx 2(5/4)^{m-1}$ over the even-distance baseline $2^{-(m-1)}$ (53× measured at $m = 16$); other double flips are generic or mildly enhanced. This is consistent with the measured over-dispersion of the solution count (Table 5), but a second-moment analysis summing all pair families is left open.

5 Classical exact algorithms and the barrier

Throughout, the instance has N operands of m bits, $q = N - 1$ free operators, $h = \lceil q/2 \rceil$, and $k = \lfloor q/4 \rfloor$.

5.1 Reachable states and meet-in-the-middle

Proof (Observation 1). Let $R_1 = \{A_1\}$ and $R_i = \{(z + A_i) \bmod 2^m : z \in R_{i-1}\} \cup \{z \odot A_i : z \in R_{i-1}\}$; accept iff $0 \in R_N$. Stored as Boolean arrays, the addition image is a cyclic shift and the XNOR image a permutation by XOR with $\overline{A_i}$, implementable in m passes, so each layer costs $O(m2^m)$. □

With sorted lists the program is output-sensitive, costing $O^*(\max_i |R_i|)$, which can be far below 2^m.

Proposition 4 (Small operands). *If $A_2, \ldots, A_N < 2^s$, then $|R_i| \le 2(2i-1)2^s$ for all i, so the instance is decidable in $O^*(2^s)$ time.*

Proof. Write $Y = Y_{\mathrm{hi}}2^s + Y_{\mathrm{lo}}$. A signed addition of an operand below 2^s changes Y_{hi} by at most ± 1 and an XOR leaves it unchanged, so after $i-1$ operations Y_{hi} lies in an interval of $2i-1$ residues; the actual state is Y or $\overline{Y}$. □

This is the ADD/XNOR analogue of the pseudo-polynomial SUBSET-SUM algorithm, with operand magnitude in the role of the target: hard instances need operands of magnitude $2^{\Omega(q)}$. Whether $O^*(2^s)$ can be improved to $\tilde{O}(N+2^s)$, as [10] does for SUBSET-SUM, is open and no longer has a conditional lower bound attached to it.

Proposition 5 (Meet-in-the-middle). ADD/XNOR *has a deterministic decision and witness-recovery algorithm using* $O^*(2^h)$ *time and space.*

Proof. Enumerate the 2^h prefixes x^L with the states they reach from A_1, and the 2^{q-h} suffixes x^R with the states from which they reach zero, applying the inverses of Lemma 1 in reverse order; sort both lists and look for a common state. A match is exactly a solution $x = (x^L, x^R)$, and every solution is split by the cut. □

By Corollary 3 the prefix enumeration may be restricted to one parity class. The attack is a claw-finding problem between the forward and backward maps on a range of size $2^m \geq 2^h$; golden-collision search with distinguished points [33] finds the claw with heuristic time $O(2^{3h/2}/\sqrt{w})$ and memory 2^w, for any $w \leq h$, interpolating from $O(2^{3q/4})$ at polynomial memory down to Proposition 5.

5.2 Low-bit dissection: the same time with less space

Schroeppel and Shamir [31] reduce the space of meet-in-the-middle for SUBSET-SUM to $O^*(2^{q/4})$ by writing each half-sum as a sum of two quarter-sums; that uses commutativity. Dissection [14] obtains the same tradeoff for compositions of permutations by guessing an intermediate state, which pays only when the per-stage key space exceeds the state space; for ADD/XNOR at $m = n$ the key space 2^q and the state space 2^m coincide up to one bit, so guessing a full state costs as much as the whole search. For knapsacks, however, [14] guess only a partial sum modulo a small number, and triangularity provides the ADD/XNOR analogue: the low bits of the middle state.

Split the operator positions into four consecutive blocks Q_1, Q_2, Q_3, Q_4 of lengths $k, k, k, q-3k$. A prefix $u \in \{0,1\}^{2k}$ has *forward state* $S(u)$, the value after operands $A_1, \ldots, A_{2k+1}$; a suffix $v \in \{0,1\}^{q-2k}$ has *backward state* $R(v)$, the unique value from which v on operands $A_{2k+2}, \ldots, A_N$ reaches zero. (u, v) is a solution iff $S(u) = R(v)$. For $p \leq k$ and $g \in \mathbb{Z}_{2^p}$ let $F_g = \{u : \rho_p(S(u)) = g\}$ and $B_g = \{v : \rho_p(R(v)) = g\}$, so that every solution lies in $F_g \times B_g$ for exactly one g, and define the *balance parameter* $\mu_p = \max_g \min(|F_g|, |B_g|)$.

Lemma 8 (Enumerating a low-bit class). *For each* g*, the list* $\{(u, S(u)) : u \in F_g\}$ *can be produced in* $O^*(2^k + |F_g|)$ *time with* $O^*(2^k)$ *working space, and all counts* $|F_g|$ *can be computed in* $O^*(2^p)$ *time and space; the same holds for* B_g*.*

Proof. Write $u = u_1 u_2$ with $u_1, u_2 \in \{0,1\}^k$. Enumerate u_1, compute the full state $S_1(u_1)$ after block Q_1, and sort the pairs by the key $\rho_p(S_1(u_1))$. Enumerate u_2 and, on the projected instance $\rho_p(I)$, apply the inverses of u_2 backwards from g to obtain $r(u_2) \in \mathbb{Z}_{2^p}$; by Lemma 3, $\rho_p(S(u_1u_2)) = g$ iff $\rho_p(S_1(u_1)) = r(u_2)$. For each hit output $(u_1u_2, S(u_1u_2))$, computing the full state by applying u_2 forward from $S_1(u_1)$. The counts come from the reachable-state program of Section 5.1 run on $\rho_p(I)$ with a counter per state. For B_g exchange the roles of forward and backward. □

Theorem 7 (Low-bit dissection). *For every $p \le k$,* ADD/XNOR *can be decided, and a witness produced, by a deterministic algorithm with worst-case time $O^*(2^{q/2})$ and space $O^*(2^{q/4} + \mu_p)$, where μ_p is computed by the algorithm in $O^*(2^{q/4})$ time before any list is stored. On uniform instances $\mathbb{E}|F_g| = 2^{2k-p}$ and $\mathbb{E}|B_g| = 2^{q-2k-p}$ for every single g; for $p = k$ both equal $O^*(2^{q/4})$.*

Proof. Compute all counts by Lemma 8. For each g, enumerate the smaller of F_g, B_g into a list sorted by full state, then enumerate the other class and test each full state against the list; a hit is a solution, and every solution is found at $g = \rho_p(S(u))$. The time is $\sum_g O^*(2^k + |F_g| + |B_g|) = O^*(2^{p+k} + 2^{2k} + 2^{q-2k}) = O^*(2^{q/2})$, since $\sum_g |F_g| = 2^{2k}$, $\sum_g |B_g| = 2^{q-2k}$ and $p \le k$; the space is $O^*(2^k)$ plus the stored class. For the expectations, $S(u)$ is uniform over A_1 given the other operands, and $R(v)$ is uniform over A_{2k+2} given the others (Lemma 1). □

The time bound is unconditional. The space bound depends on μ_p, which the algorithm computes before storing anything, and we stress that the per-class expectation does not bound $\mathbb{E}[\mu_p]$: the prefixes share operands, the class sizes are strongly dependent, and uniform marginals are compatible with a large maximum. We therefore make no claim about typical space; the balance parameter is part of the statement. μ_p can be as large as 2^{2k}, for instance when 2^p divides every operand after the first, but such instances reduce by Lemma 4 to width $m - p$, and below width $q/4$ the dynamic program runs within $O^*(2^{q/4})$; the algorithm can also choose $p \le k$ to minimize μ_p. A concentration bound for μ_p, or a worst-case algorithm with $O^*(2^{q/4})$ space, is left open.

5.3 Why the Subset-Sum barrier-breakers do not transfer

The algorithms that improve on $2^{q/2}$ for SUBSET-SUM, in the average case [17,7,9] and recently in the worst case [37,29], share the representation technique. Instead of writing a solution $\mathbf{c} \in \{0,1\}^r$ as $\mathbf{a} + \mathbf{b}$ with $\mathbf{a}, \mathbf{b}$ supported on the two halves, one chooses larger sets, e.g. $\mathbf{a} \in \{0,1\}^r$ and $\mathbf{b} \in \{-1,0,1\}^r$, so that every solution has N_{rep} representations with distinct keys $\mathbf{a} \cdot \mathbf{z}$; a random modular filter on the key keeps a $1/N_{\mathrm{rep}}$ fraction of each list while keeping one representation of each solution, the filtered lists are generated by a meet-in-the-middle *inside* each set, which is possible because the key is a homomorphism, and the survivors are sifted by a compatibility test (Orthogonal Vectors in [25], partial match in [37]; the few-sums dichotomy of [6] covers inputs whose keys do

not mix). The ingredients are thus (i) an abelian group of partial objects with a homomorphic key, (ii) exponentially many representations of each solution, and (iii) a compatibility test.

ADD/XNOR has a partial analogue of (i): the low-bit key $\rho_p(S(u))$ is not a homomorphism, but Lemma 8 generates the class of prefixes with a prescribed key in time $O^*(2^k + |F_g|)$, which is what a modular filter needs and what the dissection exploits. For (ii) the natural formulation—partial objects are prefixes and suffixes, keys are the states they reach—offers nothing.

Lemma 9 (Unique cut representation). *Fix an instance and a cut position h. Every solution x has exactly one pair (x^L, x^R) with $|x^L| = h$ realizing it, namely its prefix and suffix, and the meeting state is determined by x^L alone.*

Proof. The state after h steps is a function of x^L, and the operators are position-bound: the two operations neither commute nor associate with each other, so there is no reordering freedom. □

This is a property of the formulation, not an impossibility result: SUBSET-SUM restricted to two fixed halves also has unique decompositions, and its improved algorithms deliberately enlarge the partial-solution spaces. We can only report that the enlargements we know how to analyse fail here, and why. The key of a concatenation is $S(uv) = f_v(S(u))$, depending on the key of u and on *all* of v, so matching is separable only across disjoint intervals, and a word has only $q + 1$ such factorizations. Letting partial objects act on all positions interleaves their actions along the chain and leaves neither with a key computable from itself alone. Enlarging the key space instead, by writing the middle state as $s = (s - t) + t$ for t in a set T, gives $|T|$ representations, but each surviving object (u, t) still requires its own prefix to be enumerated, so the lists do not shrink. The only commutative substructure, runs of additions (4), contains nothing to choose. The remaining SUBSET-SUM tools meet the same obstacle in their standard forms: Schroeppel–Shamir and [25] need half-keys that are sums of quarter-keys; Wagner's algorithm [35] needs a group law for merging partial results; lattice attacks [22,12] need a $\mathbb{Z}$-linear relation, whereas by Lemma 6 the runs are coupled through carries, linear over neither $\mathbb{Z}_{2^m}$ nor $\mathbb{F}_2$, and no density parameter governs the problem. (For $m \geq q + \omega(1)$ a uniform instance has $2^{q-m} = o(1)$ expected solutions, so the interesting regimes are $m \leq q$.) A lower bound covering all representation-style algorithms would require a defined algorithmic model; the ideal chain model of Section 6 covers those that only evaluate and invert steps. Finally, if ADD/XNOR with N operands reduced to SUBSET-SUM with $N + o(N)$ integers, the new SUBSET-SUM algorithms would beat Proposition 5; the only reductions we know go through the Boolean encoding and produce $\Theta(Nm)$ integers.

Lifting, linearization, SAT, local search. By Corollary 1 the solutions modulo 2^p are those of the projected width-p instance, and a Hensel-style procedure lifts them one bit at a time; its cost is the number of solutions at each level, 2^{q-p} in expectation on uniform inputs, so its expected total $\Theta(2^q)$ is that of

exhaustive search, and combining lifting on the low $q/2$ bits with meet-in-the-middle on the strings reproduces $2^{q/2}$. For the Boolean system of Remark 1 we have no analysis of linearization or Gröbner-basis methods and know of none below $2^{q/2}$. The CNF encoding (ripple-carry adders for $+$, direct clauses for $\odot$, one selector per step, about $20Nm$ clauses) has q free variables; Section 7 reports that over $n \le 30$ two CDCL solvers scale as roughly $2^{0.71n}$ and $2^{0.74n}$ on planted instances and $2^{0.69n}$ on refutations—finite-size observations with few runs, not asymptotic exponents. Binary decision diagrams are the symbolic form of the reachable-state program with the same 2^m bound. Flipping one operator injects a difference of expected weight $\approx m/2$ (Proposition 3); this describes the single-flip neighbourhood of $x \mapsto E_A(x)$, not the behaviour of any local-search algorithm, which we do not analyse. Guessing the carries linearizes the system, but there are $\Theta(qm)$ of them.

Summary. None of the approaches analysed in this section—dynamic programming, meet-in-the-middle and its memoryless variants, dissection, the standard forms of Schroeppel–Shamir, the representation technique, generalized birthday and lattice reduction, bit-by-bit lifting, and CDCL—improves the exponent $1/2$ of Proposition 5, against a current best for Subset-Sum of 0.49 (claimed) and $1/2 - 1/450000$ in the worst case and 0.283 on random density-one instances. This records their status as analysed here, not a universal impossibility; triangularity, the one structural feature beyond invertibility, has so far bought space, not time.

6 A matching lower bound in the ideal chain model

Exhaustive search, meet-in-the-middle and its memoryless variants use the instance only by evaluating chain steps forward or backward from states they have reached. We make this the model and show that in it the square-root barrier is exactly the price of invertibility. The result concerns an idealized distributional experiment—planted search against independent random permutations—and we say explicitly what it does and does not imply about real instances.

6.1 The ideal chain model

Fix q and m. An instance of the ideal chain model is a family of $2q$ independent uniformly random permutations

$$\pi_{i,b} : \{0,1\}^m \to \{0,1\}^m, \qquad i \in \{1,\dots,q\},\ b \in \{0,1\},$$

where $\pi_{i,0}$ plays the role of "add A_{i+1}" and $\pi_{i,1}$ of "XNOR A_{i+1}" at step i. For $x \in \{0,1\}^q$ let $\Pi_x(s_0) := \pi_{q,x_q} \circ \cdots \circ \pi_{1,x_1}(s_0)$. An adversary $\mathcal{A}$ has oracle access to every $\pi_{i,b}$ and $\pi_{i,b}^{-1}$; its cost is the number of queries.

Planted search experiment. Sample the permutations, $s_0 \leftarrow \{0,1\}^m$ and $x^* \leftarrow \{0,1\}^q$, and set $\mathtt{t} := \Pi_{x^*}(s_0)$. $\mathcal{A}$ receives $(s_0, \mathtt{t})$, makes at most T queries, and outputs $x \in \{0,1\}^q$; it *succeeds* if $\Pi_x(s_0) = \mathtt{t}$.

Planted decision experiment. Sample $\beta \leftarrow \{0,1\}$; if $\beta = 1$ proceed as above, if $\beta = 0$ let $\mathtt{t} \leftarrow \{0,1\}^m$ be independent of everything else. $\mathcal{A}$ receives $(s_0, \mathtt{t})$ and outputs a guess of β.

A real instance has $\pi_{i,0}(z) = z + A_{i+1}$ and $\pi_{i,1}(z) = z \odot A_{i+1}$ (Lemma 1); the model replaces these structured, mutually dependent permutations by independent ideal ones. Algorithms that only compose and invert steps—meet-in-the-middle, exhaustive search, distinguished-point search—transfer to the model; algorithms that use arithmetic structure, such as the dynamic program and the dissection, do not, and the independence of the ideal permutations removes relations such algorithms could exploit. The model thus delimits a class of algorithms; as with the generic group model [32], a lower bound in it does not preclude a non-generic algorithm, and it concerns planted instances in a distributional experiment, not the worst-case existence problem of Definition 1.

6.2 The main theorem

Theorem 8 (Generic hardness of planted search). *Let $q \geq 3$, $h := \lceil q/2 \rceil$, and $\max(q, 4) \leq T \leq 2^{m-3}$. In the planted search experiment of the ideal chain model, every adversary making at most T oracle queries succeeds with probability at most*

$$\frac{3T}{2^h} \;+\; \frac{10\,T^2}{2^m} \;+\; \frac{1}{2^q}.$$

Consequently success probability $1/2$ requires $T \geq c \cdot \min\bigl(2^h, 2^{m/2}\bigr)$ for an absolute constant $c > 0$; when $m \geq q$, in particular in the balanced case $m = n = q+1$, this is $T = \Omega(2^{q/2})$.

The *covering* term $3T/2^h$ is the real cost, matching meet-in-the-middle at $T \approx 2^h$; the *collision* term $10T^2/2^m$ dominates it only when $2^m \lesssim 2^q$, which is why the general statement is $\Omega(\min(2^{q/2}, 2^{m/2}))$; the *guessing* term is 2^{-q}. The restriction $T \geq \max(q, 4)$ is harmless (pad with dummy queries) and the constants are not optimized. The collision term is not an artifact: a re-convergent "ladder" of $4q$ colliding edges, with $\pi_{\ell,0}(a_{\ell-1}) = \pi_{\ell,1}(b_{\ell-1}) = a_\ell$ and $\pi_{\ell,1}(a_{\ell-1}) = \pi_{\ell,0}(b_{\ell-1}) = b_\ell$ at every level, would cover all 2^q operator strings, so collision-freeness is exactly what makes the covering count valid.

6.3 Proof of Theorem 8

Two lazy procedures. We describe two randomized procedures, Real and Dec, that answer the adversary's queries and then evaluate its output. Both maintain, for every (i, b), a partial injection ("table") recording the answered entries of $\pi_{i,b}$, initially empty. A query $\pi_{i,b}(z)$ (or $\pi_{i,b}^{-1}(z)$) whose entry is already in the table is answered from the table; otherwise the answer is *fresh*: a uniformly random value among those not yet used as an image (resp. preimage) in that table, and the new entry is recorded. A value is *seen* once it appears in s_0, $\mathtt{t}$, a query or an answer. After $\mathcal{A}$ outputs x, both procedures run the *verification walk*: they

evaluate $\Pi_x(s_0)$ by at most q further forward queries, answered by the same rules. The two procedures differ in one rule only.

Dec samples $s_0, \mathsf{t} \leftarrow \{0,1\}^m$ independently and $x^* \leftarrow \{0,1\}^q$, and never uses x^* to answer a query. Its tables are therefore uniformly random permutations revealed lazily, t is independent of everything, and x^* is independent of the entire transcript (queries, answers, output) and of t. Dec is the $\beta = 0$ decision experiment with a dummy x^*.

Real samples $s_0, \mathsf{t} \leftarrow \{0,1\}^m$ and $x^* \leftarrow \{0,1\}^q$, and enforces the constraint $\Pi_{x^*}(s_0) = \mathsf{t}$ lazily. Call the edge (i, x_i^*) at input $\Pi_{x^*_{\le i-1}}(s_0)$ the i-th *planted edge*. Real keeps a forward frontier f (the planted edges $1, \ldots, f$ have been revealed, with known endpoint s_f, initially $f = 0$) and a backward frontier g (the planted edges $q, \ldots, q-g+1$ have been revealed backwards from t, with known endpoint t_{q-g}, initially $g = 0$). A query that reveals planted edge $f+1$ forwards (i.e. $\pi_{f+1,x^*_{f+1}}(s_f)$) is answered fresh and f advances, *unless* $f + g = q - 1$, in which case the answer is *forced* to t_{q-g} (the chain closes); symmetrically for a query revealing planted edge $q - g$ backwards. All other queries follow the common rules.

Lemma 10. Real *generates exactly the planted search experiment.*

Proof. In the planted experiment the chain values $\Pi_{x^*_{\le i}}(s_0)$, $1 \le i \le q-1$, are i.i.d. uniform (each is the image of a point under an independent uniform permutation), t is uniform, and conditioned on the q planted entries the permutations are uniform. By exchangeability of the entries of a uniform permutation, revealing the planted values on demand from either end, each as a uniform value among the images not yet used in its table, and revealing all other entries as fresh values, produces the same joint law; the forced answer at closure is the already determined value of the chain. □

The stopping event. For a transcript of $\mathcal{A}$'s T queries define, in either procedure,

- Coll: some fresh answer, during the T queries or the verification walk, equals a value already seen;
- Cover_F: during the T queries, $\mathcal{A}$ has queried the path of h forward edges from s_0 labelled $x_1^*, \ldots, x_h^*$ (equivalently, in Real, $f \ge h$);
- Cover_B: during the T queries, $\mathcal{A}$ has queried the path of $q - h$ backward edges from t labelled $x_q^*, \ldots, x_{h+1}^*$ (in Real, $g \ge q - h$);

and let $\mathsf{BAD} := \mathsf{Cover}_F \vee \mathsf{Cover}_B \vee \mathsf{Coll}_T$, where Coll_T is Coll restricted to the T queries. All three are events determined by the transcript and x^*, and the time of their first occurrence is a stopping time.

Lemma 11 (Identical until bad). *Run* Real *and* Dec *with the same* s_0, t, x^**, the same adversary randomness and the same fresh draws, and stop both at the first query at which* BAD *occurs. Up to the stopping time the two transcripts coincide. Consequently* $\Pr_{\mathsf{Real}}[\mathsf{BAD}] = \Pr_{\mathsf{Dec}}[\mathsf{BAD}]$*, and* $\Pr_{\mathsf{Real}}[E \wedge \neg\mathsf{BAD}] = \Pr_{\mathsf{Dec}}[E \wedge \neg\mathsf{BAD}]$ *for every event* E *determined by* $\mathcal{A}$*'s transcript, output and* x^**.*

Proof. The procedures differ only in the forced answer at closure, which requires $f + g = q - 1$ at the moment of a frontier advance. Before BAD we have $f \leq h - 1$ and $g \leq q - h - 1$, so $f + g \leq q - 2$ and no closure occurs; every other query is answered by the common rules from identical tables and identical fresh draws. By induction on the queries, the tables and transcripts coincide up to the stopping time. The two probability statements are the fundamental lemma of game playing applied to the stopped processes. □

Lemma 12 (Bad events in Dec). $\Pr_{\mathsf{Dec}}[\mathsf{Coll}] \leq 10T^2/2^m$ *and* $\Pr_{\mathsf{Dec}}[(\mathsf{Cover}_F \vee \mathsf{Cover}_B) \wedge \neg\mathsf{Coll}_T] \leq 3T/2^h$. *The same bound on* Coll *holds in* Real *for the fresh answers of the verification walk.*

Proof. At most $T + q \leq 2T$ fresh answers are drawn in total, each table holds at most $2T$ entries, and at most $2 + 3T$ values are ever seen (the inputs and answers of the T queries and the answers of the walk). A fresh answer is uniform over at least $2^m - 2T \geq \frac{3}{4}2^m$ values and hits the seen set with probability at most $(2+3T)/(\frac{3}{4}2^m)$; a union bound over $2T$ draws gives $\Pr[\mathsf{Coll}] \leq \frac{8}{3}(3T^2+2T)/2^m \leq 10T^2/2^m$ for $T \geq 4$. The same computation applies to the verification walk in Real, whose non-forced answers are fresh by the same rule.

For the covering events, consider the graph whose vertices are seen values and whose edges are the answered entries of the T queries. On $\neg\mathsf{Coll}_T$ every fresh answer creates a new vertex, so this graph is a forest whose components are trees rooted at s_0, at t, or at points the adversary queried without having seen them; the queried paths from s_0 form a tree: distinct label prefixes of length h whose paths have been queried end in distinct level-h edges, so at most T such prefixes $\mathcal{P}_h$ exist. In Dec, x^* is independent of the transcript, hence of $\mathcal{P}_h$, so $\Pr[\mathsf{Cover}_F \wedge \neg\mathsf{Coll}_T] = \mathbb{E}[|\mathcal{P}_h|\mathbf{1}_{\neg\mathsf{Coll}_T}]/2^h \leq T/2^h$. Symmetrically at most T label suffixes of length $q - h$ have queried paths from t, so $\Pr[\mathsf{Cover}_B \wedge \neg\mathsf{Coll}_T] \leq T/2^{q-h} \leq 2T/2^h$, using $q - h \geq h - 1$. □

Lemma 13 (Success without bad events). *In* Real*, if* Coll *does not occur (during the* T *queries and the verification walk) and* $x \neq x^*$*, then* $\Pi_x(s_0) \neq \mathsf{t}$.

Proof. Consider the graph of all seen values and all answered entries, including the verification walk. On $\neg\mathsf{Coll}$ every fresh answer creates a new vertex, and a forced answer joins the forward component (containing s_0) to the backward component (containing t), which are distinct at that moment; hence the graph is a forest, and s_0 and t lie in the same component only if the chain has closed, in which case the planted chain is the unique path between them. The verification walk is a path in this forest from s_0 along the labels of x, with levels strictly increasing. If the chain has not closed, the walk stays in the component of s_0 and never reaches t. If it has closed, let j be the first index with $x_j \neq x^*_j$: the walk follows the planted chain to its vertex at level $j-1$ and then takes a non-planted edge to a vertex w at level j, w distinct from the chain vertex at level j (two edges from the same vertex to the same w would require a fresh answer to hit a seen value). The unique forest path from w to t returns through level $j - 1$, so a level-increasing walk from w cannot follow it, and any other route would be a second path. Hence the walk does not reach t. □

Conclusion. Let S be the event that $\mathcal{A}$ succeeds and Coll_V the event that Coll occurs during the verification walk. In Real,

$$\Pr[\mathsf{S}] \le \Pr[\mathsf{BAD}] + \Pr[\mathsf{Coll}_V \wedge \neg\mathsf{BAD}] + \Pr[x = x^* \wedge \neg\mathsf{BAD}] + \Pr[\mathsf{S} \wedge x \neq x^* \wedge \neg\mathsf{Coll}].$$

The last term is 0 by Lemma 13. By Lemma 11, $\Pr_{\mathsf{Real}}[\mathsf{BAD}] = \Pr_{\mathsf{Dec}}[\mathsf{BAD}]$ and $\Pr_{\mathsf{Real}}[x = x^* \wedge \neg\mathsf{BAD}] = \Pr_{\mathsf{Dec}}[x = x^* \wedge \neg\mathsf{BAD}] \le \Pr_{\mathsf{Dec}}[x = x^*] = 2^{-q}$, since in Dec the output is independent of x^*. By Lemma 12, $\Pr_{\mathsf{Dec}}[\mathsf{BAD}] + \Pr_{\mathsf{Real}}[\mathsf{Coll}_V \wedge \neg\mathsf{BAD}] \le 3T/2^h + 10T^2/2^m$, where the collision bound counts all $2T$ fresh draws once. This proves the displayed bound. For the consequence, if the success probability is at least $1/2$ then one of the three terms is at least $1/6$; the guessing term is below $1/6$ for $q \ge 3$, so $T \ge 2^h/18$ or $T \ge \sqrt{2^m/60}$, and when $m \ge q$ the second alternative implies $T = \Omega(2^{q/2})$ as well. □

Corollary 4 (Planted decision in the ideal chain model). *Under the hypotheses of Theorem 8, every adversary making at most T queries distinguishes $\beta = 1$ from $\beta = 0$ in the planted decision experiment with advantage at most $3T/2^h + 10T^2/2^m$. Hence constant advantage requires $T = \Omega(\min(2^{q/2}, 2^{m/2}))$ queries, and $T = \Omega(2^{q/2})$ when $m \ge q$.*

Proof. The $\beta = 1$ experiment is Real and the $\beta = 0$ experiment is Dec, both without the verification walk. By Lemma 11 the output distributions differ by at most $\Pr_{\mathsf{Dec}}[\mathsf{BAD}]$, which Lemma 12 bounds by $3T/2^h + \Pr_{\mathsf{Dec}}[\mathsf{Coll}_T] \le 3T/2^h + 10T^2/2^m$. □

The corollary concerns distinguishing a planted target from an independent one in the ideal model; it does not bound the complexity of deciding whether a given real instance has a solution.

7 Experiments

All experiments use a reference implementation of $E_A(x)$, the carry decomposition, meet-in-the-middle with backward evaluation, a planting routine, exact dynamic programming over the transfer operators of Section 4, and a CNF encoding solved by CaDiCaL [8] and Glucose 4. Balanced instances have $N = m = n$.

Linear structure. Table 4 validates the run-product formula of Theorem 3 by Monte Carlo over random A ($m = 16$, 1.2×10^5 samples), including the exact nulls at run length 2 mod 4. The average correlation of Theorem 4 was confirmed exactly against dynamic programming for $N \le 40$ and by Monte Carlo ($N = 6$: 0.112 vs. 0.1094). For Theorem 5, the eigenvalue formula agrees with direct diagonalization to machine precision ($j \le 4$), the closed form for $c_j(z)$ agrees with exact rational dynamic programming for $z \le 16$ (e.g. $c_2(1..4) = \frac{1}{4}, -\frac{1}{4}, -\frac{7}{32}, 0$ and $c_3(1..4) = \frac{1}{8}, -\frac{5}{16}, -\frac{31}{256}, \frac{25}{256}$), Monte Carlo on 16-bit words confirms the model (for $j = 2$: measured $+0.2478, -0.2519, -0.2183, +0.0019$ at $z = 1, \ldots, 4$ against $+\frac{1}{4}, -\frac{1}{4}, -\frac{7}{32}, 0$), and the bounds of parts (iii)–(iv) hold numerically for $j \le 8$, $z \le 40$.

Table 4. Bit-1 correlation: Monte Carlo vs. the closed form $\prod_j c(z_j)$ of Theorem 3, $c(z) = 2^{-z/2}\cos(z\pi/4)$.

Runs	Monte Carlo	$\prod_j c(z_j)$	Runs	Monte Carlo	$\prod_j c(z_j)$
$[1,1]$	$+0.2504$	$+0.2500$	$[3]$	-0.2532	-0.2500
$[1,1,1]$	$+0.1265$	$+0.1250$	$[4]$	-0.2494	-0.2500
$[3,3]$	$+0.0614$	$+0.0625$	$[5]$	-0.1310	-0.1250
$[2,1]$	-0.0012	0.0000	$[7]$	$+0.0623$	$+0.0625$
$[1,2]$	$+0.0029$	0.0000			

Table 5. Solution-count statistics at $m = n$ by full enumeration (2×10^5, 10^5, 3×10^4 instances). Poisson(1/2) reference: $P_0 = 0.6065$, Fano $= 1$.

$n = m$	mean	Fano factor	P_0	P_1
10	0.4995	1.617	0.678	0.207
12	0.5009	1.430	0.659	0.230
14	0.4955	1.263	0.643	0.256

Differential structure and solution counts. Lemma 7(iii) was verified exhaustively at $m = 10$ for eight representative differences, and the transposition law (Theorem 6) exhaustively over all 2^{3m} triples at $m = 8, 10$ (ten decimal places) and by Monte Carlo at $m = 10, \ldots, 20$, where the fitted decay rate is $2^{-0.677m}$ against the predicted $\log_2(5/8) = -0.678$. At $m = 16$, $N = 12$ (2×10^5 instances) the collision probabilities of double-flip families, relative to the even-distance baseline $2^{-(m-1)}$, are 53.2 and 52.9 for adjacent transpositions (exact law 55.9), 0.8–1.0 for adjacent $(+,+) \to (\odot,\odot)$ flips, and 8.0 for a separated double flip. Table 5 reports solution-count statistics at $m = n$ by full enumeration: the mean is $1/2$ as predicted, and the Fano factor exceeds 1 and decays toward 1 over $n = 10, 12, 14$. This is consistent with an aggregate of polynomially many exponentially small pairwise correlations, of which the transposition law is one explicitly computed family; the data do not by themselves establish a Poisson limit or attribute the variance to transpositions alone.

Meet-in-the-middle scaling. On random balanced instances with $n = 21, 23, \ldots, 41$ (median of three runs), the meet-in-the-middle solver's median time grows by a factor 2.05–2.15 per two units of n above $n = 29$, with fitted slope 0.55 bits per unit n (Figure 1) against the theoretical 0.50; the operation count is $2^{(n-1)/2}$ by construction, and we have not separately measured the memory-hierarchy effects we suspect behind the excess.

SAT-solver scaling. Table 6 and Figure 1 report median solve times on planted instances (CaDiCaL, $n \le 30$; Glucose 4, $n \le 26$) and median refutation times on unsatisfiable random instances (about 60% of random instances at $m = n$; CaDiCaL, $n \le 26$). Least-squares fits over the full ranges give $\approx 2^{0.71n}$ (CaDiCaL, planted), $\approx 2^{0.74n}$ (Glucose, planted) and $\approx 2^{0.69n}$ (CaDiCaL, refutation),

Table 6. Median SAT-solver times (s) at $m = n$: planted instances (CaDiCaL: 5 runs, 10 at n=28, 3 at n=30; Glucose 4: 5 runs) and unsatisfiable random instances (CaDiCaL, 5 runs). Least-squares growth rates over the full ranges: $2^{0.71n}$, $2^{0.74n}$, $2^{0.69n}$ (finite-size fits; see text for range sensitivity); meet-in-the-middle is $2^{0.5n}$.

$n = m$	14	16	18	20	22	24	26	28	30
CaDiCaL, planted	—	0.054	0.068	0.298	0.529	1.514	3.257	11.2	57.9
Glucose 4, planted	—	0.047	0.113	0.344	0.634	2.140	9.002	—	—
CaDiCaL, unsatisfiable	0.020	0.052	0.171	0.328	0.729	2.611	7.086	—	—

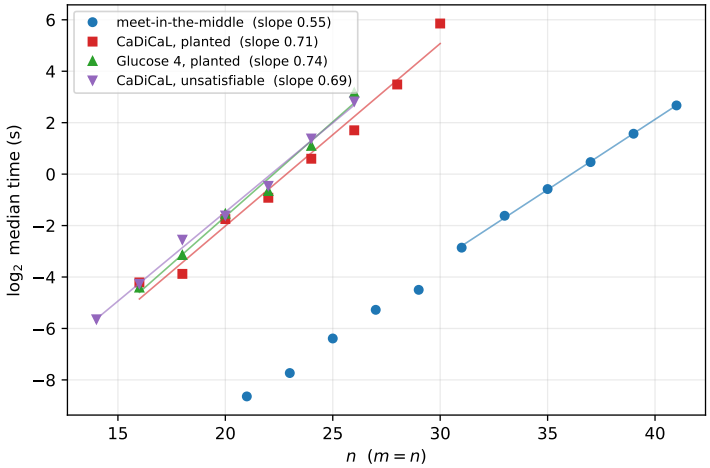


Fig. 1. Log-scale median running times at $m = n$ with least-squares slopes (bits of time per unit n): meet-in-the-middle (0.55; theoretical 0.50), CaDiCaL and Glucose on planted instances (0.71, 0.74), and CaDiCaL refutation of unsatisfiable instances (0.69).

steeper than the $2^{0.5n}$ of meet-in-the-middle. The fits are range-sensitive (restricted to $n \leq 26$ the CaDiCaL planted fit is $\approx 2^{0.58n}$) and the per-instance variance is large (planted $n = 28$: 3.7–29.5 s over ten runs); they are finite-size observations with few runs and establish neither asymptotic exponents nor exponential resolution length, which we state only as hypotheses. Within the tested range no solver approaches the meet-in-the-middle exponent.

8 Conclusion and open questions

Four-bit guards and parity normalization give a deterministic Subset-Sum reduction proving Add/XNOR NP-complete, including every fixed rational power relation between operand count and word width, with linear parameter blow-up from 3-SAT. The chain's dominant linear and differential behaviour is exactly

computable and experimentally confirmed. Meet-in-the-middle runs in $O^*(2^{q/2})$, triangularity gives the same time with space $O^*(2^{q/4})$ plus an explicit balance parameter, the $2^{q/2}$ bound is tight for planted search in the ideal chain model, CDCL solvers did worse in our experiments, and the technique that recently broke the barrier for Subset-Sum needs a commutative structure that the natural formulation of Add/XNOR does not provide. The lower bound is model-relative and distributional, and the average-case hardness that would underlie cryptographic use is a conjecture, not a theorem.

Three questions remain central: an algorithm with time $O^*(2^{(1/2-\varepsilon)n})$, which must use a property of the operations beyond evaluation and inversion (Theorem 8) and cannot use commutativity of partial solutions in the natural formulation (Lemma 9); a $2^{o(n)}$ algorithm, open in both directions now that ETH has been refuted, a positive answer to which would refute ETH independently via Theorem 2; and a parameter-preserving reduction between Add/XNOR and Subset-Sum in either direction. Also open are a concentration bound for the balance parameter of the dissection, the $\{+, \oplus\}$ variant, a resolution lower bound for the CNF encoding, and the second moment of the solution count. Add/XNOR is a small, explicit problem on which the question that survives the current landscape—whether meet-in-the-middle can be beaten without commutativity—can be studied with no hypothesis in the way.

References

1. A. Abboud, K. Bringmann, D. Hermelin, and D. Shabtay. SETH-based lower bounds for subset sum and bicriteria path. *ACM Transactions on Algorithms*, 18(1):6:1–6:22, 2022.
2. W. Aiello, M. Bellare, G. Di Crescenzo, and R. Venkatesan. Security amplification by composition: The case of doubly-iterated, ideal ciphers. In *Advances in Cryptology – CRYPTO '98*, volume 1462 of *LNCS*, pages 390–407, 1998.
3. J. Alman and V. Vassilevska Williams. Truly subquadratic 3SUM and truly subcubic APSP via triangles in sparse lopsided graphs, 2026. arXiv:2610.06783, 5 October 2026.
4. Anthropic. Lean formalization of the 3SUM and APSP algorithms, 2026.
5. Anthropic. Manuscript on the refutation of the exponential time hypothesis, 2026. Dated 26 October 2026.
6. P. Austrin, P. Kaski, M. Koivisto, and J. Nederlof. Dense subset sum may be the hardest. In *33rd Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 47 of *LIPIcs*, pages 13:1–13:14, 2016.
7. A. Becker, J.-S. Coron, and A. Joux. Improved generic algorithms for hard knapsacks. In *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 364–385, 2011.
8. A. Biere, K. Fazekas, M. Fleury, and M. Heisinger. CaDiCaL, Kissat, Paracooba, Plingeling and Treengeling entering the SAT competition 2020. In *Proceedings of SAT Competition 2020*. University of Helsinki, 2020.
9. X. Bonnetain, R. Bricout, A. Schrottenloher, and Y. Shen. Improved classical and quantum algorithms for subset-sum. In *Advances in Cryptology – ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 633–666, 2020.

10. K. Bringmann. A near-linear pseudopolynomial time algorithm for subset sum. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1073–1084, 2017.
11. X. Chen, Y. Jin, T. Randolph, and R. A. Servedio. Subset sum in time $2^{n/2}/\mathrm{poly}(n)$. In *Approximation, Randomization, and Combinatorial Optimization (APPROX/RANDOM 2023)*, volume 275 of *LIPIcs*, pages 39:1–39:18, 2023.
12. M. J. Coster, A. Joux, B. A. LaMacchia, A. M. Odlyzko, C.-P. Schnorr, and J. Stern. Improved low-density subset sum algorithms. *Computational Complexity*, 2:111–128, 1992.
13. W. Diffie and M. E. Hellman. Exhaustive cryptanalysis of the NBS data encryption standard. *Computer*, 10(6):74–84, 1977.
14. I. Dinur, O. Dunkelman, N. Keller, and A. Shamir. Efficient dissection of composite problems, with applications to cryptanalysis, knapsacks, and combinatorial search problems. In *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *LNCS*, pages 719–740, 2012.
15. P. Gaži and U. Maurer. Cascade encryption revisited. In *Advances in Cryptology – ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 37–51, 2009.
16. E. Horowitz and S. Sahni. Computing partitions with applications to the knapsack problem. *Journal of the ACM*, 21(2):277–292, 1974.
17. N. Howgrave-Graham and A. Joux. New generic algorithms for hard knapsacks. In *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 235–256, 2010.
18. R. Impagliazzo and R. Paturi. On the complexity of k-SAT. *Journal of Computer and System Sciences*, 62(2):367–375, 2001.
19. R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001.
20. R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.
21. A. Klimov and A. Shamir. A new class of invertible mappings. In *Cryptographic Hardware and Embedded Systems – CHES 2002*, volume 2523 of *LNCS*, pages 470–483, 2002.
22. J. C. Lagarias and A. M. Odlyzko. Solving low-density subset sum problems. *Journal of the ACM*, 32(1):229–246, 1985.
23. H. Lipmaa and S. Moriai. Efficient algorithms for computing differential properties of addition. In *Fast Software Encryption (FSE) 2001*, volume 2355 of *LNCS*, pages 336–350, 2002.
24. M. Mucha, J. Nederlof, J. Pawlewicz, and K. Węgrzycki. Equal-subset-sum faster than the meet-in-the-middle. In *27th Annual European Symposium on Algorithms (ESA)*, volume 144 of *LIPIcs*, pages 73:1–73:16, 2019.
25. J. Nederlof and K. Węgrzycki. Improving Schroeppel and Shamir's algorithm for subset sum via orthogonal vectors. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1670–1683, 2021.
26. OpenAI. Building scientific understanding with academic communities, 2026. Release announcement, October 2026.
27. OpenAI. The exponential time hypothesis is false, 2026. Manuscript, version dated 25 October 2026.
28. OpenAI. Lean formalization scope note 378: The exponential time hypothesis is false, 2026.
29. OpenAI. Subset sum in time $O(2^{0.49n})$, 2026.

30. T. Randolph and K. Węgrzycki. Beating meet-in-the-middle for subset balancing problems. In *Proceedings of the 58th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1314–1325, 2026. arXiv:2511.10823.
31. R. Schroeppel and A. Shamir. A $T = O(2^{n/2})$, $S = O(2^{n/4})$ algorithm for certain NP-complete problems. *SIAM Journal on Computing*, 10(3):456–464, 1981.
32. V. Shoup. Lower bounds for discrete logarithms and related problems. In *Advances in Cryptology – EUROCRYPT '97*, volume 1233 of *LNCS*, pages 256–266, 1997.
33. P. C. van Oorschot and M. J. Wiener. Parallel collision search with cryptanalytic applications. *Journal of Cryptology*, 12(1):1–28, 1999.
34. V. Vassilevska Williams. On some fine-grained questions in algorithms and complexity. In *Proceedings of the International Congress of Mathematicians (ICM 2018)*, pages 3447–3487, 2018.
35. D. Wagner. A generalized birthday problem. In *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *LNCS*, pages 288–304, 2002.
36. J. Wallén. Linear approximations of addition modulo 2^n. In *Fast Software Encryption (FSE) 2003*, volume 2887 of *LNCS*, pages 261–273, 2003.
37. L. Ye and B. Li. Subset sum via partial match, 2026. arXiv:2610.10446v1. Theorem 4.1: randomized time $O(2^{(1/2-1/450000)n})$ for n integers of poly(n) bits.