Does Subset Sum Admit Short Proofs?

Michał Włodarczyk
University of Warsaw, Poland

Abstract

We investigate the question whether Subset Sum can be solved by a polynomial-time algorithm with access to a certificate of length poly(k) where k is the maximal number of bits in an input number. In other words, can it be solved using only few nondeterministic bits?

This question has motivated us to initiate a systematic study of certification complexity of parameterized problems. Apart from Subset Sum, we examine problems related to integer linear programming, scheduling, and group theory. We reveal an equivalence class of problems sharing the same hardness with respect to having a polynomial certificate. These include Subset Sum and Boolean Linear Programming parameterized by the number of constraints. Secondly, we present new techniques for establishing lower bounds in this regime. In particular, we show that Subset Sum in permutation groups is at least as hard for nondeterministic computation as 3Coloring in bounded-pathwidth graphs.

2012 ACM Subject Classification Theory of computation → Fixed parameter tractability

Keywords and phrases subset sum, nondeterminism, fixed-parameter tractability

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2024.58

Related Version *Full Version*: `https://arxiv.org/abs/2409.03526` [92]

Funding *Michał Włodarczyk*: Supported by Polish National Science Centre SONATA-19 grant number 2023/51/D/ST6/00155.

1 Introduction

Nondeterminism constitutes a powerful lens for studying complexity theory. The most prominent instantiation of this concept is the class NP capturing all problems with solutions checkable in polynomial time. Another well-known example is the class NL of problems that can be solved nondeterministically in logarithmic space [7]. But the usefulness of nondeterminism is not limited to merely filtering candidates for deterministic classes. A question studied in proof complexity theory is how much nondeterminism is needed to solve certain problems or, equivalently, how long proofs have to be to prove certain theorems [27]. Depending on the considered logic, these theorems may correspond to instances of problems complete for NP [65], coNP [28], or W[SAT] [31]. The central goal of proof complexity is to establish lower bounds for increasingly powerful proof systems in the hope of building up techniques to prove, e.g., NP $\neq$ coNP. What is more, there are connections between nondeterministic running time lower bounds and fine-grained complexity [25]. In the context of online algorithms, nondeterminism is used to measure how much knowledge of future requests is needed to achieve a certain performance level [14, 18, 35].

Bounded nondeterminism plays an important role in organizing parameterized complexity theory. The first class studied in this context was W[P] comprising parameterized problems solvable in FPT time, i.e., $f(k) \cdot \text{poly}(n)$, when given access to $f(k) \cdot \log(n)$ nondeterministic bits [29]. In the last decade, classes defined by nondeterministic computation in limited space have attracted significant attention [5, 43, 81] with a recent burst of activity around the class XNLP [15, 16, 17] of problems solvable in nondeterministic time $f(k) \cdot \text{poly}(n)$ and space $f(k) \cdot \log(n)$. While the study of W[P] and XNLP concerns problems considered very hard from the perspective of FPT algorithms, a question that has eluded a systematic examination so far is **how much nondeterminism is necessary to solve FPT problems in polynomial time.** A related question has been asked about the amount of nondeterminism needed to solve d-CNF-SAT in sub-exponential time [32].

To concretize our question, we say that a parameterized problem P admits a *polynomial certificate* if an instance (I, k) can be solved in polynomial time when given access to $\mathrm{poly}(k)$ nondeterministic bits[1]. For example, every problem in NP admits a polynomial certificate under parameterization by the input length. This definition captures, e.g., FPT problems solvable via branching as a certificate can provide a roadmap for the correct branching choices. Furthermore, every parameterized problem that is in NP and admits a polynomial kernelization has a polynomial certificate given by the NP certificate for the compressed instance. The containment in NP plays a subtle role here: Wahlström [90] noted that a polynomial compression for the K-Cycle problem is likely to require a target language from outside NP exactly because K-Cycle does not seem to admit a polynomial certificate.

In this article we aim to organize the folklore knowledge about polynomial certification into a systematic study, provide new connections, techniques, and motivations, and lay the foundations for a hardness framework.

When is certification easy or hard? The existence of a certificate of size $p(k) = \mathrm{poly}(k)$ entails an FPT algorithm with running time $2^{p(k)}\mathrm{poly}(n)$, by enumerating all possible certificates. We should thus restrict ourselves only to problems solvable within such running time. On the other hand, when such an algorithm is available then one can solve the problem in polynomial time whenever $p(k) \le \log n$. Therefore, it suffices to handle the instances with $\log n < p(k)$. Consequently, for such problems it is equivalent to ask for a certificate of size $\mathrm{poly}(k + \log n)$ as this can be bounded polynomially in k via the mentioned trade-off (see Lemma 11). This observation yields polynomial certificates for problems parameterized by the solution size, such as Multicut [74] or Planarization [56], which do not fall into the previously discussed categories.

What are the problems solvable in time $2^{k^{\mathcal{O}(1)}} n^{\mathcal{O}(1)}$ yet unlikely to admit a polynomial certificate? The Bandwidth problem has been conjectured not be belong to W[P] because one can merge multiple instances into one, without increasing the parameter, in such a way that the large instance is solvable if and only if all the smaller ones are. It is conceivable that a certificate for the large instance should require at least one bit for each of the smaller instances, hence it cannot be short [45]. The same argument applies to every parameterized problem that admits an AND-composition, a construction employed to rule out polynomial kernelization [33, 47], which effectively encodes a conjunction of multiple 3SAT instances as a single instance of the problem. Such problems include those parameterized by graph width measures like treewidth or pathwidth, and it is hard to imagine polynomial certificates for them. However, kernelization hardness can be also established using an OR-composition, which does not stand at odds with polynomial certification.

The close connection between AND-composition and polynomial certificates has been observed by Drucker, Nederlof, and Santhanam [39] who focused on parameterized search problems solvable by *One-sided Probabilistic Polynomial* (OPP) algorithms (cf. [80]). They asked which problems admit an OPP algorithm that finds a solution with probability $2^{-\mathrm{poly}(k)}$.

[1] It is more accurate to say that a "certificate" refers to a particular instance while a problem can admit a "certification". We have decided however to choose a shorter and more established term. We also speak of "certificates" instead of "witnesses" because "witness" sometimes refers to a concrete representation of a solution for problems in NP, see e.g., [39].

This may seem much more powerful than using poly(k) nondeterministic bits but the success probability can be replaced by $\Omega(1)$ when given a single access to an oracle solving k-variable Circuit-SAT [39, Lemma 3.5]. A former result of Drucker [37] implies that an OPP algorithm with success probability $2^{-\mathrm{poly}(k)}$ (also called a *polynomial Levin witness compression*) for a search problem admitting a so-called constructive AND-composition would imply NP $\subseteq$ coNP/poly [39, Theorem 3.2].

Constructive vs. non-constructive proofs. The restriction to search problems is crucial in the work [39] because the aforementioned hardness result does not apply to algorithms that may recognize yes-instances without constructing a solution explicitly but by proving its existence in a non-constructive fashion. As noted by Drucker [38, §1.3], his negative results do not allow to rule out this kind of algorithms.

In general, search problems may be significantly harder than their decision counterparts. For example, there are classes of search problems for which the solution is always guaranteed to exist (e.g., by the pigeonhole principle, in the case of class PPP [1]) but the existence of a polynomial algorithm computing some solution is considered unlikely. For a less obvious example, consider finding a non-trivial divisor of a given integer n. A polynomial (in $\log n$) algorithm finding a solution could be used to construct the factorization of n, resolving a major open problem. But the existence of a solution is equivalent to n being composite and this can be verified in polynomial time by the AKS primality test [4].

As yet another example, consider the problem of finding a knotless embedding of a graph, i.e., an embedding in $\mathbb{R}^3$ in which every cycle forms a trivial knot in a topological sense. The class $\mathcal{G}$ of graphs admitting such an embedding is closed under taking minors so Robertson and Seymour's Theorem ensures that $\mathcal{G}$ is characterized by a finite set of forbidden minors [85], leading to a polynomial algorithm for recognizing graphs from $\mathcal{G}$. Observe that excluding all the forbidden minors yields a non-constructive proof that a knotless embedding exists. On the other hand, the existence of a polynomial algorithm constructing such an embedding remains open [71].

To address this discrepancy, we propose the following conjecture which asserts that not only *finding* assignments to many instances of 3SAT requires many bits of advice but even *certifying* that such assignments exist should require many bits of advice. We define the parameterized problem AND-3SAT[k] where an instance consists of a sequence of n many 3SAT formulas on k variables each, and an instance belongs to the language if all these formulas are satisfiable. We treat k as a parameter.

▶ **Conjecture 1.** AND-3SAT[k] *does not admit a polynomial certificate unless NP $\subseteq$ coNP/poly.*

Observe that AND-3SAT[k] is solvable in time $2^k \cdot \mathrm{poly}(k) \cdot n$, so the questions whether it admits a certificate of size poly(k), poly(k) $\cdot \log n$, or poly($k + \log n$) are equivalent. We formulate Conjecture 1 as a conditional statement because we believe that its proof in the current form is within the reach of the existing techniques employed in communication complexity and kernelization lower bounds [33, 80]. Then the known examples of AND-composition could be interpreted as reductions from AND-3SAT[k] that justify non-existence of polynomial certificates.

1.1 The problems under consideration

Our focus: Subset Sum. In Subset Sum we are given a sequence of n integers (also called items), a target integer t (all numbers encoded in binary), and we ask whether there is a subsequence summing up to t. This is a fundamental NP-hard problem that can be solved in pseudo-polynomial time $\mathcal{O}(tn)$ by the classic algorithm by Bellman from the 50s [11]. In 2017 the running time has been improved to $\tilde{\mathcal{O}}(t+n)$ by Bringmann [19]. Subset Sum reveals miscellaneous facets in complexity theory: it has been studied from the perspective of exponential algorithms [77, 78], logarithmic space [59, 61], approximation [21, 26, 63, 76], kernelization [36, 52, 57], fine-grained complexity [3, 22, 83, 82], cryptographic systems [55], and average-case analysis [75]. Our motivating goal is the following question.

▶ **Question 2.** *Does* Subset Sum *admit a polynomial certificate for parameter* $k = \log t$*?*

From this point of view, the pseudo-polynomial time $\mathcal{O}(tn)$ can be interpreted as FPT running time $\mathcal{O}(2^k n)$. It is also known that a kernelization of size poly(k) is unlikely [36]. Observe that we cannot hope for a certificate of size $o(\log t)$ because the algorithm enumerating all possible certificates would solve Subset Sum in time $2^{o(\log t)} n^{\mathcal{O}(1)} = t^{o(1)} n^{\mathcal{O}(1)}$ contradicting the known lower bound based on the Exponential Time Hypothesis (ETH) [3].

The parameterization by the number of relevant bits exhibits a behavior different from those mentioned so far, that is, width parameters and solution size, making it an uncharted territory for nondeterministic algorithms. The study of Subset Sum[$\log t$] was suggested by Drucker et al. [39] in the context of polynomial witness compression. These two directions are closely related yet ultimately incomparable: the requirement to return a solution makes the task more challenging but the probabilistic guarantee is less restrictive than constructing a certificate. However, establishing hardness in both paradigms boils down to finding a reduction of a certain kind from AND-3SAT[k].

The *density* of a Subset Sum instance is defined as $n \,/\, \log(t)$ in the cryptographic context [8, 54, 69]. As it is straightforward to construct a certificate of size n, we are mostly interested in instances of high density, which also appear hard for exponential algorithms [8]. On the other hand, instances that are very dense enjoy a special structure that can be leveraged algorithmically [22, 49]. The instances that seem the hardest in our regime are those in which n is slightly superpolynomial in $\log t$. Apart from the obvious motivation to better understand the structure of Subset Sum, we believe that the existence of short certificates for dense instances could be valuable for cryptography.

There are several other studied variants of the problem. In Unbounded Subset Sum the input is specified in the same way but one is allowed to use each number repeatedly. Interestingly, this modification enables us to certify a solution with $\mathcal{O}(\log^2 t)$ bits (see the full version). Another variant is to replace the addition with some group operation. In Group-G Subset Sum we are given a sequence of n elements from G and we ask whether one can pick a subsequence whose group product equals the target element $t \in G$. Note that we do not allow to change the order of elements when computing the product what makes a difference for non-commutative groups. This setting has been mostly studied for G being the cyclic group $\mathbb{Z}_q$ [9, 10, 24, 66, 84]. To capture the hardness of an instance, we choose the parameter to be $\log |G|$ (or equivalent). In particular, we will see that Group-$\mathbb{Z}_q$ Subset Sum[$\log q$] is equivalent in our regime to Subset Sum[$\log t$]. We will also provide examples of groups for which certification is either easy or conditionally hard.

Integer Linear Programming. We shall consider systems of equations in the form $\{Ax = b \mid x \in \{0,1\}^n\}$ where $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, with the parameter being the number of constraints m. This is a special case of Integer Linear Programming (ILP) over boolean domain, known in the literature as *pseudo-boolean optimization*. This case has been recognized as particularly interesting by the community working on practical ILP solvers because pseudo-boolean optimization can be treated with SAT solvers [6, 23, 40, 60, 89]. It is also applicable in the fields of approximation algorithms [91] and election systems [72]. Eisenbrand and Weismantel [42] found an elegant application of Steinitz Lemma to this problem and gave an FPT algorithm with running time $(||A||_\infty + m)^{\mathcal{O}(m^2)} \cdot n$ (cf. [58]).

For simplicity we will consider variants with bounded $||A||_\infty$. In the 0-1 ILP problem we restrict ourselves to matrices $A \in \{-1, 0, 1\}^{m \times n}$ and in MONOTONE 0-1 ILP we consider $A \in \{0, 1\}^{m \times n}$. A potential way to construct a short certificate would be to tighten the *proximity bounds* from [42] for such matrices: find an extremal solution x^* to the linear relaxation $\{Ax = b \mid x \in [0, 1]^n\}$ and hope that some integral solution z lies nearby, i.e., $||z - x^*||_1 \leq \text{poly}(m)$. This however would require tightening the bounds on the vector norms in the *Graver basis* of the matrix A. Unfortunately, there are known lower bounds making this approach hopeless [12, 13, 67].

It may be tempting to seek the source of hardness in the large values in the target vector b. We will see however that the problem is no easier when we assume $b = 0$ and look for any non-zero solution. We refer to such problem as 0-SUM 0-1 ILP.

A special case of MONOTONE 0-1 ILP is given by a matrix $A \in \{0, 1\}^{m \times n}$ with $n = \binom{m}{d}$ columns corresponding to all size-d subsets of $\{1, \ldots, m\}$. Then $Ax = b$ has a boolean solution if and only if b forms a degree sequence of some d-hypergraph, i.e., it is d-hypergraphic. There is a classic criterion by Erdős for a sequence to be graphic [44] (i.e., 2-hypergraphic) but already for $d = 3$ deciding if b is d-hypergraphic becomes NP-hard [34]. It is straightforward to certify a solution with m^d bits, what places the problem in NP for each fixed d, but it is open whether it is in NP when d is a part of the input (note that the matrix A is implicit so the input size is $\mathcal{O}(md \log m)$). This basically boils down to the same dilemma: **can we certify the existence of a boolean ILP solution x without listing x in its entirety?**

There is yet another motivation to study the certification complexity of 0-1 ILP. If this problem admits a certificate of size poly(m) then any other problem that can be modeled by 0-1 ILP with few constraints must admit a short certificate as well. This may help classifying problems into these that can be solved efficiently with ILP solvers and those that cannot.

Other problems parameterized by the number of relevant bits. A classic generalization of SUBSET SUM is the KNAPSACK problem where each item is described by a size p_i and a weight w_i; here we ask for a subset of items of total weight at least w but total size not exceeding t. Following Drucker et al. [39] we parameterize it by the number of bits necessary to store items' sizes and weights, i.e., $\log(t + w)$. The need to process weights makes KNAPSACK harder than SUBSET SUM from the perspective of fine-grained complexity [30] but they are essentially equivalent on the ground of exponential algorithms [77]. We will see that they are equivalent in our regime as well.

We will also consider the following scheduling problem which is in turn a generalization of KNAPSACK. In SCHEDULING WEIGHTED TARDY JOBS we are given a set of n jobs, where each job $j \in [n]$ has a processing time $p_j \in \mathbb{N}$, a weight $w_j \in \mathbb{N}$, and a due date $d_j \in \mathbb{N}$. We schedule the jobs on a single machine and we want to minimize the total weight of jobs completed after their due dates (those jobs are called *tardy*). Equivalently, we try to maximize the total weight of jobs completed in time.

In the scheduling literature, SCHEDULING WEIGHTED TARDY JOBS is referred to as $1||\sum w_j U_j$ using Graham's notation. The problem is solvable in pseudo-polynomial time $\mathcal{O}(n \cdot d_{max})$ by the classic Lawler and Moore's algorithm [70]. The interest in $1||\sum w_j U_j$ has been revived due to the recent advances in fine-grained complexity [2, 20, 46, 64]. Here, the parameter that captures the number of relevant bits is $\log(d_{max} + w_{max})$.

1.2 Our contribution

The standard *polynomial parameter transformation* (PPT) is a polynomial-time reduction between parameterized problems that maps an instance with parameter k to one with parameter $k' = \text{poly}(k)$. We introduce the notion of a *nondeterministic polynomial parameter transformation* (NPPT) which extends PPT by allowing the reduction to guess poly(k) nondeterministic bits. Such reductions preserve the existence of a polynomial certificate. We write $P \leq_{\text{NPPT}} Q$ (resp. $P \leq_{\text{PPT}} Q$) to indicate that P admits a NPPT (resp. PPT) into Q.

We demonstrate how NPPT help us organize the theory of polynomial certification, similarly as PPT come in useful for organizing the theory of Turing kernelization [53]. As our first result, we present an equivalence class of problems that share the same certification-hardness status as Subset Sum[$\log t$]. In other words, either all of them admit a polynomial certificate or none of them. Despite apparent similarities between these problems, some of the reductions require a nontrivial use of nondeterminism.

▶ **Theorem 3.** *The following parameterized problems are equivalent with respect to NPPT:*
1. Subset Sum[$\log t$]
2. Knapsack[$\log(t + w)$], Knapsack[$\log(p_{max} + w_{max})$]
3. 0-1 ILP[m], Monotone 0-1 ILP[m], 0-Sum 0-1 ILP[m]
4. Group-$\mathbb{Z}_q$ Subset Sum[$\log q$]

Even though we are unable to resolve Question 2, we believe that revealing such an equivalence class supports the claim that a polynomial certificate for Subset Sum[$\log t$] is unlikely. Otherwise, there must be some intriguing common property of all problems listed in Theorem 3 that has eluded researchers so far despite extensive studies in various regimes.

Next, we present two negative results. They constitute a proof of concept that AND-3SAT[k] can be used as a non-trivial source of hardness. First, we adapt a reduction from [2] to show that scheduling with weights and due dates is hard assuming Conjecture 1.

▶ **Theorem 4 (★).** AND-3SAT[k] $\leq_{PPT}$ Scheduling Weighted Tardy Jobs[$\log(d_{max} + w_{max})$].

It is possible to formulate this result in terms of AND-composition but we chose not to work with this framework since it is tailored for refuting kernelization and relies on concepts that do not fit into our regime (e.g., polynomial relation [47]).

Our second hardness result involves Group-S_k Subset Sum[k]: a variant of Subset Sum on permutation groups. Such groups contain exponentially-large cyclic subgroups (see Lemma 13) so this problem is at least as hard as Group-$\mathbb{Z}_q$ Subset Sum[$\log q$] (which is equivalent to Subset Sum[$\log t$]). We reduce from 3Coloring parameterized by pathwidth which is at least as hard as AND-3SAT[k] with respect to PPT. Indeed, we can transform each 3SAT formula in the input (each of size $\mathcal{O}(k^3)$) into an instance of 3Coloring via the standard NP-hardness proof, and take the disjoint union of such instances, which implies AND-3SAT[k] $\leq_{\text{PPT}}$ 3Coloring[pw]. Notably, the reduction in the other direction is unlikely (see the full version) so 3Coloring[pw] is probably harder than AND-3SAT[k].

▶ **Theorem 5.** 3Coloring[pw] $\leq_{NPPT}$ Group-S_k Subset Sum[k].

Consequently, Group-S_k Subset Sum[k] does not admit a polynomial certificate assuming Conjecture 1 and NP $\not\subseteq$ coNP/poly. Unlike Theorem 4, this time establishing hardness requires a nondeterministic reduction. An interesting feature of 3Coloring[pw] is that it is NL-complete under logspace reductions when the pathwidth pw is restricted to $\mathcal{O}(\log n)$ [5, 81]. On the other hand, Subset Sum can be solved in time $\tilde{\mathcal{O}}(tn^2)$ and space polylog(tn) using algebraic techniques [59]. Therefore, obtaining a logspace PPT from 3Coloring[pw] to Subset Sum[$\log t$] (where pw $= \log n$ implies $t = 2^{\text{polylog}(n)}$) would lead to a surprising consequence: a proof that NL $\subseteq$ DSPACE(polylog(n)) that is significantly different from Savitch's Theorem (see also discussion in [81, §1] on low-space determinization). This suggests that a hypothetical reduction to Subset Sum[$\log t$] should either exploit the "full power" of NPPT (so it cannot be improved to a logspace PPT) or start directly from AND-3SAT[k].

Finally, we examine the case of the group family $\mathbb{Z}_k^k$ on which Subset Sum is still NP-hard (as this generalizes Subset Sum on cyclic groups) but enjoys a polynomial certificate. Specifically, we exploit the bound on the maximal order of an element in $\mathbb{Z}_k^k$ to prove that there always exists a solution of bounded size.

▶ **Lemma 6 (★).** Group-$\mathbb{Z}_k^k$ Subset Sum[k] *admits a polynomial certificate.*

In summary, Group-G Subset Sum appears easy for $G = \mathbb{Z}_k^k$ (due to bounded maximal order), hard for $G = S_k$ (due to non-commutativity), and the case $G = \mathbb{Z}_{2^k}$ lies somewhere in between. In the light of Theorem 3, tightening this gap seems a promising avenue to settle Question 2.

Organization of the paper. We begin with the preliminaries where we formally introduce the novel concepts, such as NPPT. We prove Theorems 3 and 5 in Sections 3 and 4, respectively. The proofs marked with (★) can be found in the full version of the article [92].

2 Preliminaries

We denote the set $\{1, \ldots, n\}$ by $[n]$. For a sequence $x_1, x_2, \ldots, x_n$, its subsequence is any sequence of the form $x_{i_1}, \ldots, x_{i_m}$ for some choice of increasing indices $1 \le i_1 < \cdots < i_m \le n$. All considered logarithms are binary.

A parameterized problem P is formally defined as a subset of $\Sigma^* \times \mathbb{N}$. For the sake of disambiguation, whenever we refer to a parameterized problem, we denote the choice of the parameter in the $[\cdot]$ bracket, e.g., 3Coloring[pw]. We call P *fixed-parameter tractable* (FPT) is the containment $(I, k) \in P$ can be decided in time $f(k) \cdot \text{poly}(|I|)$ for some computable function f. We say that P admits a *polynomial compression* into a problem Q if there is a polynomial-time algorithm that transform (I, k) into an equivalent instance of Q of size poly(k). If Q coincides with the non-parameterized version of P then such an algorithm is called a *polynomial kernelization.* A *polynomial Turing kernelization* for P is a polynomial-time algorithm that determines if $(I, k) \in P$ using an oracle that can answer if $(I', k') \in P$ whenever $|I'| + k' \le \text{poly}(k)$.

▶ **Definition 7.** *Let $P \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized problem. We say that P has a polynomial certificate if there is an algorithm $\mathcal{A}$ that, given an instance (I, k) of P and a string y of poly(k) bits, runs in polynomial time and accepts or rejects (I, k) with the following guarantees.*

1. *If $(I, k) \in P$, then there exists y for which $\mathcal{A}$ accepts.*
2. *If $(I, k) \notin P$, then $\mathcal{A}$ rejects (I, k) for every y.*

▶ **Lemma 8.** *Let $P \subseteq \Sigma^* \times \mathbb{N}$ and $Q \subseteq \Sigma^*$. Suppose that $Q \in NP$ and P admits a polynomial compression into Q. Then P admits a polynomial certificate.*

Proof. For a given instance (I, k) of P we execute the compression algorithm to obtain an equivalent instance I' of Q of size poly(k). Since $Q \in$ NP the instance I' can be solved in polynomial-time with an access to a string y of poly$(|I'|) =$ poly(k) nondeterministic bits. Then y forms a certificate for (I, k). ◀

▶ **Definition 9.** *Let $P, Q \subseteq \Sigma^* \times \mathbb{N}$ be parameterized problems. An algorithm $\mathcal{A}$ is called a polynomial parameter transformation (PPT) from P to Q if, given an instance (I, k) of P, runs in polynomial time, and outputs an equivalent instance (I', k') of Q with $k' \le$ poly(k).*

An algorithm $\mathcal{B}$ is called a nondeterministic polynomial parameter transformation (NPPT) from P to Q if, given an instance (I, k) of P and a string y of poly(k) bits, runs in polynomial time, and outputs an instance (I', k') of Q with the following guarantees.
1. *$k' \le$ poly(k)*
2. *If $(I, k) \in P$, then there exists y for which $\mathcal{B}$ outputs $(I', k') \in Q$.*
3. *If $(I, k) \notin P$, then $\mathcal{B}$ outputs $(I', k') \notin Q$ for every y.*

Clearly, PPT is a special case of NPPT. We write $P \le_{\text{PPT}} Q$ ($P \le_{\text{NPPT}} Q$) if there is a (nondeterministic) PPT from P to Q. We write $P \equiv_{\text{PPT}} Q$ ($P \equiv_{\text{NPPT}} Q$) when we have reductions in both directions. It is easy to see that the relation $\le_{\text{NPPT}}$ is transitive. Similarly as the relation $\le_{\text{PPT}}$ is monotone with respect to having a polynomial kernelization, the relation $\le_{\text{NPPT}}$ is monotone with respect to having a polynomial certificate.

▶ **Lemma 10.** *Let $P, Q \in \Sigma^* \times \mathbb{N}$ be parameterized problems. If $P \le_{\text{NPPT}} Q$ and Q admits a polynomial certificate then P does as well.*

Proof. Given an instance (I, k) of P the algorithm guesses a string y_1 of length poly(k) guiding the reduction to Q and constructs an instance (I', k') with $k' =$ poly(k). Then it tries to prove that $(I', k') \in Q$ by guessing a certificate y_2 of length poly$(k') =$ poly(k). ◀

A different property transferred by PPT is polynomial Turing kernelization. Hermelin et al. [53] proposed a hardness framework for this property by considering complexity classes closed under PPT (the WK-hierarchy).

Next, we prove the equivalence mentioned in the Introduction.

▶ **Lemma 11.** *Suppose $P \subseteq \Sigma^* \times \mathbb{N}$ admits an algorithm $\mathcal{A}$ deciding if $(I, k) \in P$ in time $2^{p(k)}$poly$(|I|)$ where $p(\cdot)$ is a polynomial function. Then $P[k] \equiv_{\text{PPT}} P[k + \log |I|]$.*

Proof. The direction $P[k + \log |I|] \le_{\text{PPT}} P[k]$ is trivial. To give a reduction in the second direction, we first check if $p(k) \le \log |I|$. If yes, we execute $\mathcal{A}$ in time poly$(|I|)$ and according to the outcome we return a trivial yes/no-instance. Otherwise we have $\log |I| < p(k)$ so we can output (I, k') for the new parameter $k' = k + \log |I|$ being polynomial in k. ◀

Pathwidth. A path decomposition of a graph G is a sequence $\mathcal{P} = (X_1, X_2, \dots, X_r)$ of *bags*, where $X_i \subseteq V(G)$, and:
1. For each $v \in V(G)$ the set $\{i \mid v \in X_i\}$ forms a non-empty subinterval of $[r]$.
2. For each edge $uv \in E(G)$ there is $i \in [r]$ with $\{u, v\} \subseteq X_i$.

The *width* of a path decomposition is defined as $\max_{i=1}^{r} |X_i| - 1$. The *pathwidth* of a graph G is the minimum width of a path decomposition of G.

▶ **Lemma 12** ([29, Lemma 7.2])**.** *If a graph G has pathwidth at most p, then it admits a* nice *path decomposition $\mathcal{P} = (X_1, X_2, \dots, X_r)$ of width at most p, for which:*
- $X_1 = X_r = \emptyset$.

- *For each $i \in [r-1]$ there is either a vertex $v \notin X_i$ for which $X_{i+1} = X_i \cup \{v\}$ or a vertex $v \in X_i$ for which $X_{i+1} = X_i \setminus \{v\}$.*

Furthermore, given any path decomposition of G, we can turn it into a nice path decomposition of no greater width, in polynomial time.

The bags of the form $X_{i+1} = X_i \cup \{v\}$ are called *introduce bags* while the ones of the form $X_{i+1} = X_i \setminus \{v\}$ are called *forget bags*.

Similarly as in the previous works [5, 16, 81] we assume that a path decomposition of certain width is provided with the input. This is not a restrictive assumption for our model since pathwidth can be approximated within a polynomial factor in polynomial time [50].

Group theory. The basic definitions about groups can be found in the book [86]. A homomorphism between groups G, H is a mapping $\phi\colon G \to H$ that preserves the group operation, i.e., $\phi(x) \circ_H \phi(y) = \phi(x \circ_G y)$ for all $x, y \in G$. An isomorphism is a bijective homomorphism and an automorphism of G is an isomorphism from G to G. We denote by $Aut(G)$ the automorphism group of G with the group operation given as functional composition. A subgroup N of G is *normal* if for every $g \in G, n \in N$ we have $g \circ_G n \circ_G g^{-1} \in N$.

The symmetric group S_k comprises permutations over the set $[k]$ with the group operation given by composition. For a permutation $\pi \in S_k$ we consider a directed graph over the vertex set $[k]$ and arcs given as $\{(v, \pi(v)) \mid v \in [k]\}$. The cycles of this graph are called the cycles of π.

We denote by $\mathbb{Z}_k$ the cyclic group with addition modulo k. We write the corresponding group operation as $\oplus_k$. An order of an element $x \in G$ is the size of the cyclic subgroup of G generated by x. The Landau's function $g(k)$ is defined as the maximum order of an element x in S_k. It is known that $g(k)$ equals $\max \mathsf{lcm}(k_1, \dots, k_\ell)$ over all partitions $k = k_1 + \dots + k_\ell$ (these numbers correspond to the lengths of cycles in x) and that $g(k) = 2^{\Theta(\sqrt{k \log k})}$ [79]. An element of large order can be found easily if we settle for a slightly weaker bound.

▶ **Lemma 13.** *For each k there exists $\pi \in S_k$ of order $2^{\Omega(\sqrt{k}/\log k)}$ and it can be found in time poly(k).*

Proof. Consider all the primes $p_1, \dots, p_\ell$ that are smaller than $\sqrt{k}$. By the prime number theorem there are $\ell = \Theta(\sqrt{k}/\log k)$ such primes [51]. We have $p_1 + \dots + p_\ell \le \sqrt{k} \cdot \sqrt{k} = k$ so we can find a permutation in S_k with cycles of lengths $p_1, \dots, p_\ell$ (and possibly trivial cycles of length 1). We have $\mathsf{lcm}(p_1, \dots, p_\ell) = \prod_{i=1}^{\ell} p_i \ge 2^\ell = 2^{\Omega(\sqrt{k}/\log k)}$. ◀

For two groups N, H and a homomorphism $\phi\colon H \to Aut(N)$ we define the *outer semidirect product* [86] $N \rtimes_\phi H$ as follows. The elements of $N \rtimes_\phi H$ are $\{(n, h) \mid n \in N, h \in H\}$ and the group operation $\circ$ is given as $(n_1, h_1) \circ (n_2, h_2) = (n_1 \circ \phi_{h_1}(n_2), h_1 \circ h_2)$. A special case of the semidirect product occurs when we combine subgroups of a common group.

▶ **Lemma 14** ([86, §4.3]). *Let G be a group with a normal subgroup N and a subgroup H, such that every element $g \in G$ can be written uniquely as $g = n \circ h$ for $n \in N, h \in H$. Let $\phi\colon H \to Aut(N)$ be given as $\phi_h(n) = h \circ n \circ h^{-1}$ (this is well-defined because N is normal in G). Then G is isomorphic to the semidirect product $N \rtimes_\phi H$.*

Group-G Subset Sum	**Parameter:** $\log \lvert G \rvert$
Input: A sequence of elements $g_1, g_2, \dots, g_n \in G$, an element $g \in G$	
Question: Is there a subsequence $(i_1 < i_2 < \dots < i_r)$ of $[n]$ such that $g_{i_1} \circ g_{i_2} \circ \dots \circ g_{i_r} = g$?	

We assume that the encoding of the group elements as well as the group operation $\circ$ are implicit for a specific choice of a group family. For a group family parameterized by k, like $(S_k)_{k=1}^{\infty}$, we treat k as the parameter. In all considered cases it holds that $k \le \log |G| \le \text{poly}(k)$ so these two parameterizations are equivalent under PPT.

3 Equivalences

We formally introduce the variants of ILP that will be studied in this section.

> 0-1 ILP **Parameter:** m
> **Input:** A matrix $A \in \{-1, 0, 1\}^{m \times n}$, a vector $b \in \mathbb{Z}^m$
> **Question:** Is there a vector $x \in \{0, 1\}^n$ for which $Ax = b$?

In Monotone 0-1 ILP we restrict ourselves to matrices $A \in \{0, 1\}^{m \times n}$. In 0-Sum 0-1 ILP we have $A \in \{-1, 0, 1\}^{m \times n}$ and we seek a binary vector $x \neq 0$ for which $Ax = 0$.

We first check that all the parameterized problems considered in this section are solvable in time $2^{k^{\mathcal{O}(1)}} n^{\mathcal{O}(1)}$. For Subset Sum$[\log t]$ we can use the classic $\mathcal{O}(tn)$-time algorithm [11] which can be easily modified to solve Group-$\mathbb{Z}_q$ Subset Sum$[\log q]$ in time $\mathcal{O}(qn)$. For Knapsack$[\log(p_{max} + w_{max})]$ there is an $\mathcal{O}(p_{max} \cdot w_{max} \cdot n)$-time algorithm [82] which also works for the larger parameterization by $\log(t + w)$. Next, 0-1 ILP$[m]$ can be solved in time $2^{\mathcal{O}(m^2 \log m)} \cdot n$ using the algorithm for general matrix A [42]. This algorithm can be used to solve 0-sum 0-1 ILP$[m]$ due to Observation 21. Hence by Lemma 11 we can assume in our reductions that $(\log n)$ is bounded by a polynomial function of the parameter.

▶ **Lemma 15.** Knapsack$[\log(t + w)] \equiv_{PPT}$ Knapsack$[\log(p_{max} + w_{max})]$.

Proof. We only need to show the reduction from Knapsack$[\log(p_{max} + w_{max})]$. When $p_{max} \cdot n < t$ we can afford taking all the items. On the other hand, if $w_{max} \cdot n < w$ then no solution can exist. Therefore, we can assume that $\log t \le \log p_{max} + \log n$ and $\log w \le \log w_{max} + \log n$. By Lemma 11 we can assume $\log n$ to be polynomial in $\log(p_{max} + w_{max})$ so the new parameter $\log(t + w)$ is polynomial in the original one. ◀

▶ **Lemma 16.** Subset Sum$[\log t] \equiv_{NPPT}$ Knapsack$[\log(t + w)]$.

Proof. The $(\le)$ reduction is standard: we translate each input integer p_i into an item (p_i, p_i) and set $w = t$. Then we can pack items of total weight t into a knapsack of capacity t if and only if the Subset Sum instance is solvable.

Now consider the $(\ge)$ reduction. Let $k = \log(t+w)$. By the discussion at the beginning of this section we can assume that $\log n \le \log(t \cdot w) \le 2k$. We can also assume that $w_{max} < w$ as any item with weight exceeding w and size fitting into the knapsack would form a trivial solution. Let $W = w \cdot n + 1$.

Suppose there is a set of items with total size equal $t' \le t$ and total weight equal $w' \ge w$. Note that w' must be less than W. We nondeterministically choose t' and w': this requires guessing $\log t + \log W \le 4k$ bits. Now we create an instance of Subset Sum by mapping each item (p_i, w_i) into integer $p_i \cdot W + w_i$ and setting the target integer to $t'' = t' \cdot W + w'$. If we guessed (t', w') correctly then such an instance clearly has a solution. On the other hand, if this instance of Subset Sum admits a solution then we have $\sum_{i \in I} (p_i \cdot W + w_i) = t' \cdot W + w'$ for some $I \subseteq [n]$. Since both w' and $\sum_{i \in I'} w_i$ belong to $[1, W)$ we must have $\sum_{i \in I} w_i = w'$ and $\sum_{i \in I} p_i = t'$ so the original instance of Knapsack has a solution as well. Finally, it holds that $\log t'' \le 5k$ so the parameter is being transformed linearly. ◀

We will need the following extension of the last argument.

▶ **Lemma 17.** *Let $W \in \mathbb{N}$ and $a_1, \dots, a_n$, $b_1, \dots, b_n$ be sequences satisfying $a_i, b_i \in [0, W)$ for each $i \in [n]$. Suppose that $S := \sum_{i=1}^n a_i W^{i-1} = \sum_{i=1}^n b_i W^{i-1}$. Then $a_i = b_i$ for each $i \in [n]$.*

Proof. Consider the remainder of S when divided by W. Since W divides all the terms in S for $i \in [2, n]$ and $a_1, b_1 \in [0, W)$ we must have $a_1 = (S \mod W) = b_1$. Next, consider $S' = (S - a_1)/W = \sum_{i=2}^n a_i W^{i-2} = \sum_{i=2}^n b_i W^{i-2}$. Then $a_2 = (S' \mod W) = b_2$. This argument generalizes readily to every $i \in [n]$. ◀

▶ **Lemma 18.** Subset Sum$[\log t] \equiv_{NPPT}$ Monotone 0-1 ILP$[m]$.

Proof. ($\le$): Consider an instance $(\{p_1, \dots, p_n\}, t)$ of Subset Sum. Let $k = \lceil \log t \rceil$. We can assume that all numbers p_i belong to the interval $[t]$. For an integer $x \in [t]$ let $\mathsf{bin}(x) \in \{0,1\}^k$ denote the binary encoding of x so that $x = \sum_{j=1}^k \mathsf{bin}(x)_j \cdot 2^{j-1}$. Observe that the condition $x_1 + \dots + x_m = t$ can be expressed as $\sum_{j=1}^k \left(\sum_{i=1}^m \mathsf{bin}(x_i)_j\right) \cdot 2^{j-1} = t$.

We nondeterministically guess a sequence $b = (b_1, \dots, b_k)$ so that b_j equals $\sum_{i \in I} \mathsf{bin}(p_i)_j$ where $I \subseteq [n]$ is a solution. This sequence must satisfy $\max_{j=1}^k b_j \le t$, and so we need k^2 nondeterministic bits to guess b. We check if the sequence b satisfies $\sum_{j=1}^k b_j \cdot 2^{j-1} = t$; if no then the guess was incorrect and we return a trivial no-instance. Otherwise we construct an instance of Monotone 0-1 ILP$[k]$ with a system $Ax = b$. The vector b is given as above and its length is k. The matrix A comprises n columns where the i-th column is $\mathsf{bin}(p_i)$. This system has a solution $x \in \{0,1\}^n$ if and only if there exists $I \subseteq [n]$ so that $\sum_{i \in I} \mathsf{bin}(p_i)_j = b_j$ for all $j \in [k]$. This implies that $\sum_{i \in I} p_i = t$. Conversely, if such a set $I \subseteq [n]$ exists, then there is $b \in [t]^k$ for which $Ax = b$ admits a boolean solution.

($\ge$): Consider an instance $Ax = b$ of Monotone 0-1 ILP$[m]$. As usual, we assume $\log n \le \mathrm{poly}(m)$. We can also assume that $||b||_\infty \le n$ as otherwise $Ax = b$ is clearly infeasible. We construct an instance of Subset Sum with n items and target integer $t = \sum_{j=1}^m b_j \cdot (n+1)^{j-1}$. Note that $t \le m \cdot ||b||_\infty \cdot (n+1)^m$ so $\log t \le \mathrm{poly}(m)$. For $i \in [n]$ let $a^i \in \{0,1\}^m$ denote the i-th column of the matrix A. We define $p_i = \sum_{j=1}^m a_j^i \cdot (n+1)^{j-1}$ and we claim that that instance $J = (\{p_1, \dots, p_n\}, t)$ of Subset Sum is solavble exactly when the system $Ax = b$ has a boolean solution.

First, if $x \in \{0,1\}^m$ forms a solution to $Ax = b$ then for each $j \in [m]$ we have $\sum_{i=1}^n x_i a_j^i (n+1)^{j-1} = b_j (n+1)^{j-1}$ and so $\sum_{i=1}^n x_i p_i = t$. Hence the set $I = \{i \in [n] \mid x_i = 1\}$ encodes a solution to J. In the other direction, suppose that there is $I \subseteq [n]$ for which $\sum_{i \in I} p_i = t$. Then $t = \sum_{j=1}^m (\sum_{i \in I} a_j^i) \cdot (n+1)^{j-1}$. Due to Lemma 17 we must have $b_i = \sum_{i \in I} a_j^i$ for each $i \in [m]$ and there is subset of columns of A that sums up to the vector b. This concludes the proof. ◀

For the next reduction, we will utilize the lower bound on the norm of vectors in a so-called Graver basis of a matrix. For two vectors $y, x \in \mathbb{Z}^n$ we write $y \lhd x$ if for every $i \in [n]$ it holds that $y_i x_i \ge 0$ and $|y_i| \le |x_i|$. A non-zero vector $x \in \mathbb{Z}^n$ belongs to the Graver basis of $A \in \mathbb{Z}^{m \times n}$ if $Ax = 0$ and no other non-zero solution $Ay = 0$ satisfies $y \lhd x$. In other words, x encodes a sequence of columns of A, some possibly repeated or negated, that sums to 0 and none of its nontrivial subsequences sums to 0. The following lemma concerns the existence of vectors with a large ℓ_1-norm in a Graver basis of a certain matrix. We state it in the matrix-column interpretation.

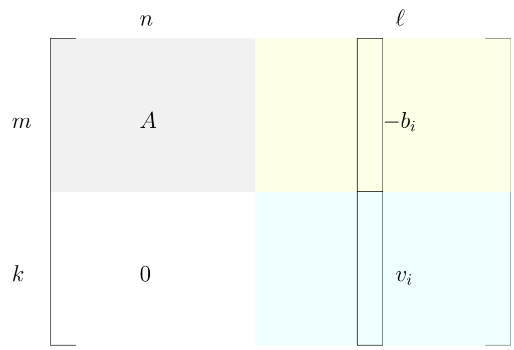

Figure 1 The matrix A' in Lemma 20.

▶ **Lemma 19** ([12, Thm. 9, Cor. 5]). *For every $k \in \mathbb{N}$ there is a sequence $(v_1, \dots, v_n)$ of vectors from $\{-1, 0, 1\}^k$ such that*
1. $n = \Theta(2^k)$*,*
2. *the vectors $v_1, \dots, v_n$ sum up to 0, and*
3. *no proper non-empty subsequence of $(v_1, \dots, v_n)$ sums up to 0.*

▶ **Lemma 20.** Monotone 0-1 ILP$[m] \leq_{PPT}$ 0-Sum 0-1 ILP$[m]$.

Proof. Consider an instance $Ax = b$ of Monotone 0-1 ILP$[m]$ with $A \in \{0, 1\}^{m \times n}$. We can assume that A contains 1 in every column as otherwise such a column can be discarded. Let $v_1, \dots, v_\ell \in \{-1, 0, 1\}^k$ be the sequence of vectors from Lemma 19 with $\ell \geq n$ and $k = \mathcal{O}(\log n)$. Next, we can assume that $||b||_\infty \leq n \leq \ell$ as otherwise there can be no solution. We decompose b into a sum $b_1 + \dots + b_\ell$ of vectors from $\{-1, 0, 1\}^m$, possibly using zero-vectors for padding. Now we construct a matrix $A' \in \{-1, 0, 1\}^{(m+k) \times (n+\ell)}$. The first n columns are given by the columns of A with 0 on the remaining k coordinates. The last ℓ columns are of the form $(-b_i, v_i)$ for $i \in [\ell]$. See Figure 1 for an illustration. The new parameter is $m + k$ which is $m + \mathcal{O}(\log n)$.

We claim that $Ax = b$ is feasible over boolean domain if and only if $A'y = 0$ admits a non-zero boolean solution y. Consider a solution x to $Ax = b$. We define y as x concatenated with vector 1^ℓ. In each of the first m rows we have $(A'y)_j = (Ax)_j - b_j = 0$. In the remaining k rows we have first n zero vectors followed by the sequence $v_1, \dots, v_\ell$ which sums up to 0 by construction. Hence $A'y = 0$ while $y \neq 0$.

Now consider the other direction and let $y \neq 0$ be a solution to $A'y = 0$. Let us decompose y as a concatenation of $y_1 \in \{0, 1\}^n$ and $y_2 \in \{0, 1\}^\ell$. First suppose that $y_2 = 0$. Then $y_1 \neq 0$ and $Ay_1 = 0$ but this is impossible since $A \in \{0, 1\}^{m \times n}$ and, by assumption, every column of A contains 1. It remains to consider the case $y_2 \neq 0$. By inspecting the last k rows of A' we infer that the non-zero indices of y_2 correspond to a non-empty subsequence of $v_1, \dots, v_\ell$ summing up to 0. By construction, this is not possible for any proper subsequence of $(v_1, \dots, v_\ell)$ so we must have $y_2 = 1^\ell$. Hence $0 = A'y = A'y_1 + A'y_2 = A'y_1 + (-b, 0)$ and so $Ay_1 = b$. This concludes the proof of the reduction. ◀

We will now reduce from 0-Sum 0-1 ILP$[m]$ to 0-1 ILP$[m]$. The subtlety comes from the fact that in the latter problem we accept the solution $x = 0$ while in the first we do not. Observe that the reduction is easy when we can afford guessing a single column from a solution. For a matrix $A \in \mathbb{Z}^{m \times n}$ and $i \in [n]$ we denote by $A^i \in \mathbb{Z}^{m \times 1}$ the i-th column of A and by $A^{-i} \in \mathbb{Z}^{m \times (n-1)}$ the matrix obtained from A by removal of the i-th column.

▶ **Observation 21.** *An instance $Ax = 0$ of* 0-SUM 0-1 ILP$[m]$ *is solvable if and only if there is $i \in [n]$ such that the instance $A^{-i}y = -A^i$ of* 0-1 ILP$[m]$ *is solvable.*

▶ **Lemma 22.** 0-SUM 0-1 ILP$[m]$ $\leq_{NPPT}$ 0-1 ILP$[m]$.

Proof. Observation 21 enables us to solve 0-SUM 0-1 ILP$[m]$ in polynomial time when $\log n$ is large compared to m, by considering all $i \in [n]$ and solving the obtained 0-1 ILP$[m]$ instance. Hence we can again assume that $\log n \leq \text{poly}(m)$. In this case, Observation 21 can be interpreted as an NPPT that guesses $\text{poly}(m)$ bits to identify the index $i \in [n]$. ◀

▶ **Lemma 23.** 0-1 ILP$[m]$ $\leq_{NPPT}$ MONOTONE 0-1 ILP$[m]$.

Proof. We decompose the matrix A as $A^+ - A^-$ where A^+, A^- have entries from $\{0,1\}$. Suppose that there exists a vector x satisfying $Ax = b$. We nondeterministically guess vectors b^+, b^- that satisfy $A^+x = b^+$, $A^-x = b^-$ and we check whether $b^+ - b^- = b$; if no then the guess is rejected. This requires $m \log n$ nondeterministic bits. We create an instance of MONOTONE 0-1 ILP$[m]$ with $2m$ constraints given as $A^+y = b^+$, $A^-y = b^-$. If we made a correct guess, then $y = x$ is a solution to the system above. On the other hand, if this system admits a solution y then $Ay = A^+y - A^-y = b^+ - b^- = b$ so y is also a solution to the original instance. ◀

▶ **Lemma 24.** SUBSET SUM$[\log t]$ $\equiv_{NPPT}$ GROUP-$\mathbb{Z}_q$ SUBSET SUM$[\log q]$.

Proof. For the reduction ($\leq$) consider $q = nt$ and leave t intact. We can assume that each input number belongs to $[1, t)$ hence the sum of every subset belongs to $[1, q)$ and so there is no difference in performing addition in $\mathbb{Z}$ or $\mathbb{Z}_q$.

Now we handle the reduction ($\geq$). Let S be the subset of numbers that sums up to t modulo q. Since each item belongs to $[0, q)$ their sum in $\mathbb{Z}$ is bounded by nq; let us denote this value as t'. We nondeterministically guess $t' \in [0, nq]$ and check whether $t' = t \mod m$. We consider an instance J of SUBSET SUM over $\mathbb{Z}$ with the unchanged items and the target t'. We have $\log t' \leq \log n + \log q$ what bounds the new parameter as well as the number of necessary nondeterministic bits. If the guess was correct then J will have a solution. Finally, a solution to J yields a solution to the original instance because $t' = t \mod q$. ◀

Using the presented lemmas, any two problems listed in Theorem 3 can be reduced to each other via NPPT.

4 Permutation Subset Sum

This section is devoted to the proof of Theorem 5. We will use an intermediate problem involving a computational model with ℓ binary counters, being a special case of bounded *Vector Addition System with States* (VASS) [68]. This can be also regarded as a counterpart of the intermediate problem used for establishing XNLP-hardness, which concerns cellular automata [16, 43].

For a sequence $\mathcal{F} = (f_1, \ldots, f_n)$, $f_i \in \{O, R\}$ (optional/required), we say that a subsequence of $[n]$ is *$\mathcal{F}$-restricted* if it contains all the indices i with $f_i = R$. We say that a sequence of vectors $v_1, \ldots, v_n \in \{-1, 0, 1\}^\ell$ forms a *0/1-run* if $v_1 + \cdots + v_n = 0$ and for each $j \in [n]$ the partial sum $v_1 + \cdots + v_j$ belongs to $\{0,1\}^\ell$.

0-1 COUNTER MACHINE	**Parameter:** ℓ
Input: Sequences $\mathcal{V} = (v_1, \ldots, v_n)$, $v_i \in \{-1, 0, 1\}^\ell$, and $\mathcal{F} = (f_1, \ldots, f_n)$, $f_i \in \{O, R\}$.	
Question: Is there a subsequence $(i_1 < i_2 < \cdots < i_r)$ of $[n]$ that is $\mathcal{F}$-restricted and such that $(v_{i_1}, v_{i_2}, \ldots, v_{i_r})$ forms a 0/1-run?	

Intuitively, a vector $v_i \in \{-1,0,1\}^\ell$ tells which of the ℓ counters should be increased or decreased. We must "execute" all the vector v_i for which $f_i = R$ plus some others so that the value of each counter is always kept within $\{0,1\}$.

We give a reduction from 3Coloring[pw] to 0-1 Counter Machine[ℓ].

3Coloring	**Parameter:** pw
Input: An undirected graph G and a path decomposition of G of width at most pw	
Question: Can we color $V(G)$ with 3 colors so that the endpoints of each edge are assigned different colors?	

▶ **Lemma 25 (★).** 3Coloring[pw] $\le_{PPT}$ 0-1 Counter Machine[ℓ].

In the proof, we assign each vertex a label from [pw + 1] so that the labels in each bag are distinct. We introduce a counter for each pair (label, color) and whenever a vertex is introduced in a bag, we make the machine increase one of the counters corresponding to its label. For each edge uv there is a bag containing both u, v; we then insert a suitable sequence of vectors so that running it is possible if and only if the labels of u, v have active counters in different colors. Finally, when a vertex is forgotten we deactivate the corresponding counter.

In order to encode the operations on counters as composition of permutations, we will employ the following algebraic construction. For $q \in \mathbb{N}$ consider an automorphism $\phi_1 \colon \mathbb{Z}_q^2 \to \mathbb{Z}_q^2$ given as $\phi_1((x,y)) = (y,x)$. Clearly $\phi_1 \circ \phi_1$ is identify, so there is a homomorphism $\phi \colon \mathbb{Z}_2 \to Aut(\mathbb{Z}_q^2)$ that assigns identity to $0 \in \mathbb{Z}_2$ and ϕ_1 to $1 \in \mathbb{Z}_2$. We define the group U_q as the outer semidirect product $\mathbb{Z}_q^2 \rtimes_\phi \mathbb{Z}_2$ (see Section 2). That is, the elements of U_q are $\{((x,y),z) \mid x,y \in \mathbb{Z}_q, z \in \mathbb{Z}_2\}$ and the group operation $\circ$ is given as

$$((x_1,y_1),z_1) \circ ((x_2,y_2),z_2) = \begin{cases} ((x_1 \oplus_q x_2),(y_1 \oplus_q y_2), z_1 \oplus_2 z_2) & \text{if } z_1 = 0 \\ ((x_1 \oplus_q y_2),(y_1 \oplus_q x_2), z_1 \oplus_2 z_2) & \text{if } z_1 = 1. \end{cases} \tag{1}$$

The z-coordinate works as addition modulo 2 whereas the element z_1 governs whether we add (x_2,y_2) or (y_2,x_2) modulo q on the (x,y)-coordinates. The neutral element is $((0,0),0)$. Note that U_q is non-commutative.

For $((x,y),z) \in U_q$ we define its *norm* as $x+y$. Consider a mapping $\Gamma \colon \{-1,0,1\} \to U_q$ given as $\Gamma(-1) = ((1,0),1)$, $\Gamma(0) = ((0,0),0)$, $\Gamma(1) = ((0,1),1)$.

▶ **Lemma 26.** *Let $b_1,\dots,b_n \in \{-1,0,1\}$ and $q > n$. Then $b_1,\dots,b_n$ forms a 0/1-run (in dimension $\ell = 1$) if and only if the group product $g = \Gamma(b_1) \circ \Gamma(b_2) \circ \cdots \circ \Gamma(b_n)$ in U_q is of the form $g = ((0,n'),0)$ for some $n' \in [n]$.*

Proof. Recall that $\Gamma(0)$ is the neutral element in U_q. Moreover, removing 0 from the sequence does not affect the property of being a 0/1-run, so we can assume that $b_i \in \{-1,1\}$ for each $i \in [n]$. Note that the inequality $q > n$ is preserved by this modification. This inequality is only needed to ensure that the addition never overflows modulo q.

Suppose now that $b_1,\dots,b_n$ is a 0/1-run. Then it comprises alternating 1s and -1s: $(1,-1,1,-1,\dots,1,-1)$. Hence the product $g = \Gamma(b_1) \circ \cdots \circ \Gamma(b_n)$ equals $(\Gamma(1) \circ \Gamma(-1))^{n/2}$. We have $((0,1),1) \circ ((1,0),1) = ((0,2),0)$ and so $g = ((0,n),0)$.

Now suppose that $b_1,\dots,b_n$ is not a 0/1-run. Then either $\sum_{i=1}^n b_i = 1$ or $\sum_{i=1}^j b_i \notin \{0,1\}$ for some $j \in [n]$. In the first scenario n is odd so g has 1 on the z-coordinate and so it is not in the form of $((0,n'),0)$. In the second scenario there are 3 cases: (a) $b_1 = -1$, (b) $(b_i,b_{i+1}) = (1,1)$ for some $i \in [n-1]$, or (c) $(b_i,b_{i+1}) = (-1,-1)$ for some $i \in [n-1]$.

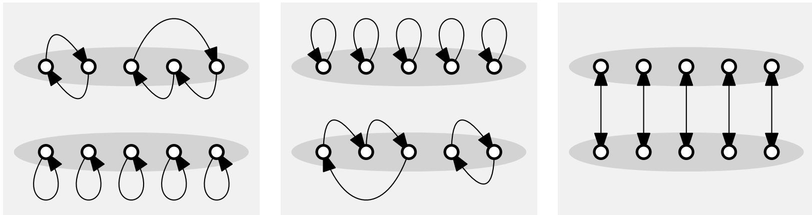

Figure 2 An illustration to Lemma 27. The three permutations are $\pi_1, \pi_0, \pi_z \in S_{10}$. The first one acts as g on the upper set and as identity on the lower set. In the second one these roles are swapped whereas π_z acts as symmetry between the two sets. The permutations $\widehat{\Gamma}(1), \widehat{\Gamma}(-1)$ in Lemma 28 are obtained as $\pi_0 \circ \pi_z$ and $\pi_1 \circ \pi_z$. Multiplying a sequence of permutations from $\{\widehat{\Gamma}(1), \widehat{\Gamma}(-1)\}$ yields a permutation acting as identity on the upper set if and only if the arguments are alternating 1s and -1s.

Case (a): $g = \Gamma(-1) \circ h = ((1,0),1) \circ h$ for some $h \in U_q$ of norm $\le n-1$. Then g cannot have 0 at the x-coordinate because $n < q$ and the addition does not overflow.

Case (b): $\Gamma(1)^2 = ((0,1),1)^2 = ((0,1) \oplus_q (1,0), 1 \oplus_2 1) = ((1,1),0)$. For any $h_1, h_2 \in U_q$ of total norm $\le n-1$ the product $h_1 \circ ((1,1),0) \circ h_2$ cannot have 0 at the x-coordinate.

Case (c): Analogous to (b) because again $\Gamma(-1)^2 = ((1,0),1)^2 = ((1,0) \oplus_q (0,1), 1 \oplus_2 1) = ((1,1),0)$. ◀

Next, we show how to embed the group U_q into a permutation group over a universe of small size. On an intuitive level, we need to implement two features: counting modulo q on both coordinates and a mechanism to swap the coordinates. To this end, we will partition the universe into two sets corresponding to the two coordinates. On each of them, we will use a permutation of order q to implement counting without interacting with the other set. Then we will employ a permutation being a bijection between the two sets, which will work as a switch. See Figure 2 for a visualization.

▶ **Lemma 27.** *For every $n \in \mathbb{N}$ there exist $q > n$ and $\hat{r} = \mathcal{O}(\log^3 n)$ for which there is a homomorphism $\chi \colon U_q \to S_{\hat{r}}$.*

Proof. By Lemma 13 we can find a permutation g of order $q > n$ in S_r for some $r = \mathcal{O}(\log^3 n)$. The subgroup of S_r generated by g is isomorphic to $\mathbb{Z}_q$. We will now consider the permutation group over the set $[r] \times \mathbb{Z}_2$, which is isomorphic to S_{2r}. Instead of writing χ explicitly, we will identify a subgroup of S_{2r} isomorphic to U_q.

Let π_z be the permutation given as $\pi_z(i,j) = (i, 1-j)$ for $(i,j) \in [r] \times \mathbb{Z}_2$, i.e., it switches the second coordinate. Let π_0 act as g on $[r] \times 0$ and as identify on $[r] \times 1$. Analogously, let π_1 act as g on $[r] \times 1$ and as identify on $[r] \times 0$. Let N be the subgroup of S_{2r} generated by π_0 and π_1; it is isomorphic to $\mathbb{Z}_q^2$ and each element of N is of the form (π_0^x, π_1^y) for some $x, y \in \mathbb{Z}_q$. Next, let H be the subgroup generated by π_z; it is isomorphic to $\mathbb{Z}_2$. Now consider a homomorphism $\phi \colon H \to Aut(N)$ given as conjugation $\phi_\pi(g) = \pi \circ g \circ \pi^{-1}$. In this special case, the semidirect product $N \rtimes_\phi H$ is isomorphic to the subgroup of S_{2r} generated by the elements of N and H (Lemma 14). On the other hand, ϕ_{π_z} maps $(\pi_0^x, \pi_1^y) \in N$ into (π_0^y, π_1^x) so this is exactly the same construction as used when defining U_q. We infer that U_q is isomorphic to a subgroup of S_{2r} and the corresponding homomorphism is given by the mapping of the generators: $\chi((0,1),0) = \pi_0$, $\chi((1,0),0) = \pi_1$, $\chi((0,0),1) = \pi_z$. ◀

Armed with such a homomorphism, we translate Lemma 26 to the language of permutations.

▶ **Lemma 28.** *For every $n \in \mathbb{N}$ there exists $r = \mathcal{O}(\log^3 n)$, a permutation $\pi \in S_r$ of order greater than n, and a mapping $\widehat{\Gamma}\colon \{-1,0,1\} \to S_r$ so that the following holds. A sequence $b_1,\ldots,b_n \in \{-1,0,1\}$ is a 0/1-run if and only if the product $\widehat{\Gamma}(b_1) \circ \widehat{\Gamma}(b_2) \circ \cdots \circ \widehat{\Gamma}(b_n)$ is of the form $\pi^{n'}$ for some $n' \in [n]$.*

Proof. Let $\chi\colon U_q \to S_r$ be the homomorphism from Lemma 27 for $q > n$ and $r = \mathcal{O}(\log^3 n)$. We define $\widehat{\Gamma}\colon \{-1,0,1\} \to S_r$ as $\widehat{\Gamma}(i) = \chi(\Gamma(i))$ using the mapping Γ from Lemma 26. Since χ is a homomorphism, the condition $\Gamma(b_1) \circ \cdots \circ \Gamma(b_n) = ((0,n'),0)$ is equivalent to $\widehat{\Gamma}(b_1) \circ \cdots \circ \widehat{\Gamma}(b_n) = \chi(((0,n'),0))$. We have $g = \chi(((0,n'),0))$ for some $n' \in [n]$ if and only if $g = \pi^{n'}$ for $\pi = \chi((0,1),0)$. The order of π is $q > n$, as requested. ◀

For a sequence of vectors from $\{0,1\}^\ell$ we can use a Cartesian product of ℓ permutation groups S_r to check the property of being a 0/1-run by inspecting the product of permutations from $S_{\ell r}$. This enables us to encode the problem with binary counters as Group-S_k Subset Sum$[k]$. We remark that we need nondeterminism to guess the target permutation. This boils down to guessing the number n' from Lemma 28 for each of ℓ coordinates.

Finally, Theorem 5 follows by combining Lemma 25 with Lemma 29.

▶ **Lemma 29 (★).** *0-1 Counter Machine$[\ell]$ $\leq_{NPPT}$ Group-S_k Subset Sum$[k]$.*

5 Conclusion

We have introduced the nondeterministic polynomial parameter transformation (NPPT) and used this concept to shed some light on the unresolved questions about short certificates for FPT problems. We believe that our work will give an impetus for further systematic study of certification complexity in various contexts.

The main question remains to decipher certification complexity of Subset Sum$[\log t]$. Even though Subset Sum enjoys a seemingly simple structure, some former breakthroughs required advanced techniques such as additive combinatorics [3, 26] or number theory [61]. Theorem 3 makes it now possible to analyze Subset Sum$[\log t]$ through the geometric lens using concepts such as lattice cones [41] or Graver bases [12, 13, 67].

Drucker et al. [39] suggested also to study k-Disjoint Paths and K-Cycle in their regime of polynomial witness compression. Recall that the difference between that model and ours is that they ask for a *randomized* algorithm that *outputs* a solution. Observe that a polynomial certificate (or witness compression) for k-Disjoint Paths would entail an algorithm with running time $2^{k^{\mathcal{O}(1)}} n^{\mathcal{O}(1)}$ which seems currently out of reach [62]. What about a certificate of size $(k + \log n)^{\mathcal{O}(1)}$?

Interestingly, Planar k-Disjoint Paths does admit a polynomial certificate: if $k^2 \leq \log n$ one can execute the known $2^{\mathcal{O}(k^2)} n^{\mathcal{O}(1)}$-time algorithm [73] and otherwise one can guess the homology class of a solution (out of $n^{\mathcal{O}(k)} \leq 2^{\mathcal{O}(k^3)}$) and then solve the problem in polynomial time [87]. Another interesting question is whether k-Disjoint Paths admits a certificate of size $(k + \log n)^{\mathcal{O}(1)}$ on acyclic digraphs. Note that we need to incorporate $(\log n)$ in the certificate size because the problem is W[1]-hard when parameterized by k [88]. The problem admits an $n^{\mathcal{O}(k)}$-time algorithm based on dynamic programming [48].

For K-Cycle we cannot expect to rule out a polynomial certificate via a PPT from AND-3SAT$[k]$ because the problem admits a polynomial compression [90], a property unlikely to hold for AND-3SAT$[k]$ [47]. Is it possible to establish the certification hardness by NPPT (which does not preserve polynomial compression) or would such a reduction also lead to unexpected consequences?

A different question related to bounded nondeterminism is whether one can rule out a logspace algorithm for directed reachability (which is NL-complete) using only polylog(n) nondeterministic bits. Observe that relying on the analog of the assumption NP $\not\subseteq$ coNP/poly for NL would be pointless because NL = coNL by Immerman-Szelepcsényi Theorem. This direction bears some resemblance to the question whether directed reachability can be solved in polynomial time and polylogarithmic space, i.e., whether NL $\subseteq$ SC [1].

References

1 Scott Aaronson. The complexity zoo, 2005. URL: https://cse.unl.edu/~cbourke/latex/ComplexityZoo.pdf.

2 Amir Abboud, Karl Bringmann, Danny Hermelin, and Dvir Shabtay. Scheduling lower bounds via AND subset sum. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 168 of *LIPIcs*, pages 4:1–4:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICS.ICALP.2020.4.

3 Amir Abboud, Karl Bringmann, Danny Hermelin, and Dvir Shabtay. SETH-based lower bounds for subset sum and bicriteria path. *ACM Trans. Algorithms*, 18(1):6:1–6:22, 2022. doi:10.1145/3450524.

4 Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Annals of mathematics*, pages 781–793, 2004.

5 Eric Allender, Shiteng Chen, Tiancheng Lou, Periklis A. Papakonstantinou, and Bangsheng Tang. Width-parametrized SAT: time–space tradeoffs. *Theory Comput.*, 10:297–339, 2014. doi:10.4086/TOC.2014.V010A012.

6 Fadi A. Aloul, Arathi Ramani, Igor L. Markov, and Karem A. Sakallah. Generic ILP versus specialized 0-1 ILP: an update. In Lawrence T. Pileggi and Andreas Kuehlmann, editors, *Proceedings of the 2002 IEEE/ACM International Conference on Computer-aided Design, ICCAD 2002, San Jose, California, USA, November 10-14, 2002*, pages 450–457. ACM / IEEE Computer Society, 2002. doi:10.1145/774572.774638.

7 Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach.* Cambridge University Press, 2009.

8 Per Austrin, Petteri Kaski, Mikko Koivisto, and Jesper Nederlof. Dense subset sum may be the hardest. In Nicolas Ollinger and Heribert Vollmer, editors, *33rd Symposium on Theoretical Aspects of Computer Science, STACS 2016, February 17-20, 2016, Orléans, France*, volume 47 of *LIPIcs*, pages 13:1–13:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPICS.STACS.2016.13.

9 Kyriakos Axiotis, Arturs Backurs, Karl Bringmann, Ce Jin, Vasileios Nakos, Christos Tzamos, and Hongxun Wu. Fast and simple modular subset sum. In Hung Viet Le and Valerie King, editors, *4th Symposium on Simplicity in Algorithms, SOSA 2021, Virtual Conference, January 11-12, 2021*, pages 57–67. SIAM, 2021. doi:10.1137/1.9781611976496.6.

10 Kyriakos Axiotis, Arturs Backurs, Ce Jin, Christos Tzamos, and Hongxun Wu. Fast modular subset sum using linear sketching. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 58–69. SIAM, 2019. doi:10.1137/1.9781611975482.4.

11 Richard Bellman. Dynamic programming. *Princeton University Press, Princeton, NJ, USA*, 1:3–25, 1958.

12 Sebastian Berndt, Matthias Mnich, and Tobias Stamm. New support size bounds and proximity bounds for integer linear programming. In *International Conference on Current Trends in Theory and Practice of Computer Science*, pages 82–95. Springer, 2024. doi:10.1007/978-3-031-52113-3_6.

13 Yael Berstein and Shmuel Onn. The Graver complexity of integer programming. *Annals of Combinatorics*, 13:289–296, 2009.

14 Hans-Joachim Böckenhauer, Dennis Komm, Rastislav Královič, Richard Královič, and Tobias Mömke. Online algorithms with advice: The tape model. *Information and Computation*, 254:59–83, 2017. doi:10.1016/J.IC.2017.03.001.

15 Hans L. Bodlaender, Carla Groenland, Hugo Jacob, Lars Jaffke, and Paloma T. Lima. XNLP-completeness for parameterized problems on graphs with a linear structure. In Holger Dell and Jesper Nederlof, editors, *17th International Symposium on Parameterized and Exact Computation, IPEC 2022, September 7-9, 2022, Potsdam, Germany*, volume 249 of *LIPIcs*, pages 8:1–8:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICS.IPEC.2022.8.

16 Hans L. Bodlaender, Carla Groenland, Jesper Nederlof, and Céline M. F. Swennenhuis. Parameterized problems complete for nondeterministic FPT time and logarithmic space. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 193–204. IEEE, 2021. doi:10.1109/FOCS52979.2021.00027.

17 Hans L. Bodlaender, Isja Mannens, Jelle J. Oostveen, Sukanya Pandey, and Erik Jan van Leeuwen. The parameterised complexity of integer multicommodity flow. In Neeldhara Misra and Magnus Wahlström, editors, *18th International Symposium on Parameterized and Exact Computation, IPEC 2023, September 6-8, 2023, Amsterdam, The Netherlands*, volume 285 of *LIPIcs*, pages 6:1–6:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.IPEC.2023.6.

18 Joan Boyar, Lene M Favrholdt, Christian Kudahl, Kim S Larsen, and Jesper W Mikkelsen. Online algorithms with advice: A survey. *ACM Computing Surveys (CSUR)*, 50(2):1–34, 2017. doi:10.1145/3056461.

19 Karl Bringmann. A near-linear pseudopolynomial time algorithm for subset sum. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1073–1084. SIAM, 2017. doi:10.1137/1.9781611974782.69.

20 Karl Bringmann, Nick Fischer, Danny Hermelin, Dvir Shabtay, and Philip Wellnitz. Faster minimization of tardy processing time on a single machine. *Algorithmica*, 84(5):1341–1356, 2022. doi:10.1007/S00453-022-00928-W.

21 Karl Bringmann and Vasileios Nakos. A fine-grained perspective on approximating subset sum and partition. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1797–1815. SIAM, 2021. doi:10.1137/1.9781611976465.108.

22 Karl Bringmann and Philip Wellnitz. On near-linear-time algorithms for dense subset sum. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1777–1796. SIAM, 2021. doi:10.1137/1.9781611976465.107.

23 Sam Buss and Jakob Nordström. Proof complexity and SAT solving. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability - Second Edition*, volume 336 of *Frontiers in Artificial Intelligence and Applications*, pages 233–350. IOS Press, 2021. doi:10.3233/FAIA200990.

24 Jean Cardinal and John Iacono. Modular subset sum, dynamic strings, and zero-sum sets. In Hung Viet Le and Valerie King, editors, *4th Symposium on Simplicity in Algorithms, SOSA 2021, Virtual Conference, January 11-12, 2021*, pages 45–56. SIAM, 2021. doi:10.1137/1.9781611976496.5.

25 Marco L. Carmosino, Jiawei Gao, Russell Impagliazzo, Ivan Mihajlin, Ramamohan Paturi, and Stefan Schneider. Nondeterministic extensions of the strong exponential time hypothesis and consequences for non-reducibility. In Madhu Sudan, editor, *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, Cambridge, MA, USA, January 14-16, 2016*, pages 261–270. ACM, 2016. doi:10.1145/2840728.2840746.

26 Lin Chen, Jiayi Lian, Yuchen Mao, and Guochuan Zhang. Approximating partition in near-linear time. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 307–318. ACM, 2024. doi:10.1145/3618260.3649727.

27 Stephen Cook and Phuong Nguyen. *Logical foundations of proof complexity*, volume 11. Cambridge University Press Cambridge, 2010.

28 Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1):36–50, 1979. doi:10.2307/2273702.

29 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

30 Marek Cygan, Marcin Mucha, Karol Węgrzycki, and Michał Włodarczyk. On problems equivalent to (min, +)-convolution. *ACM Trans. Algorithms*, 15(1):14:1–14:25, 2019. doi:10.1145/3293465.

31 Stefan S. Dantchev, Barnaby Martin, and Stefan Szeider. Parameterized proof complexity. *Comput. Complex.*, 20(1):51–85, 2011. doi:10.1007/S00037-010-0001-1.

32 Evgeny Dantsin and Edward A. Hirsch. Satisfiability certificates verifiable in subexponential time. In Karem A. Sakallah and Laurent Simon, editors, *Theory and Applications of Satisfiability Testing - SAT 2011*, pages 19–32, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. doi:10.1007/978-3-642-21581-0_4.

33 Holger Dell. AND-compression of NP-complete problems: Streamlined proof and minor observations. *Algorithmica*, 75(2):403–423, 2016. doi:10.1007/S00453-015-0110-Y.

34 Antoine Deza, Asaf Levin, Syed Mohammad Meesum, and Shmuel Onn. Hypergraphic degree sequences are hard. *Bull. EATCS*, 127, 2019. URL: http://bulletin.eatcs.org/index.php/beatcs/article/view/573/572.

35 Stefan Dobrev, Rastislav Královič, and Dana Pardubská. Measuring the problem-relevant information in input. *RAIRO-Theoretical Informatics and Applications*, 43(3):585–613, 2009. doi:10.1051/ITA/2009012.

36 Michael Dom, Daniel Lokshtanov, and Saket Saurabh. Kernelization lower bounds through colors and IDs. *ACM Trans. Algorithms*, 11(2):13:1–13:20, 2014. doi:10.1145/2650261.

37 Andrew Drucker. Nondeterministic direct product reductions and the success probability of SAT solvers. In *Proceedings of the 2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, FOCS '13, pages 736–745, USA, 2013. IEEE Computer Society. doi:10.1109/FOCS.2013.84.

38 Andrew Drucker. Nondeterministic direct product reductions and the success probability of SAT solvers, 2013. URL: https://people.csail.mit.edu/andyd/success_prob_long.pdf.

39 Andrew Drucker, Jesper Nederlof, and Rahul Santhanam. Exponential Time Paradigms Through the Polynomial Time Lens. In Piotr Sankowski and Christos Zaroliagis, editors, *24th Annual European Symposium on Algorithms (ESA 2016)*, volume 57 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 36:1–36:14, Dagstuhl, Germany, 2016. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ESA.2016.36.

40 Niklas Eén and Niklas Sörensson. Translating pseudo-boolean constraints into SAT. *Journal on Satisfiability, Boolean Modeling and Computation*, 2(1-4):1–26, 2006. doi:10.3233/SAT190014.

41 Friedrich Eisenbrand and Gennady Shmonin. Carathéodory bounds for integer cones. *Oper. Res. Lett.*, 34(5):564–568, 2006. doi:10.1016/J.ORL.2005.09.008.

42 Friedrich Eisenbrand and Robert Weismantel. Proximity results and faster algorithms for integer programming using the steinitz lemma. *ACM Trans. Algorithms*, 16(1):5:1–5:14, 2020. doi:10.1145/3340322.

43 Michael Elberfeld, Christoph Stockhusen, and Till Tantau. On the space and circuit complexity of parameterized problems: Classes and completeness. *Algorithmica*, 71(3):661–701, 2015. doi:10.1007/S00453-014-9944-Y.

44 Paul Erdős. Graphs with prescribed degree of vertices. *Mat. Lapok.*, 11:264–274, 1960.

45 Michael R. Fellows and Frances A. Rosamond. Collaborating with Hans: Some remaining wonderments. In Fedor V. Fomin, Stefan Kratsch, and Erik Jan van Leeuwen, editors, *Treewidth, Kernels, and Algorithms - Essays Dedicated to Hans L. Bodlaender on the Occasion of His 60th Birthday*, volume 12160 of *Lecture Notes in Computer Science*, pages 7–17. Springer, 2020. doi:10.1007/978-3-030-42071-0_2.

46 Nick Fischer and Leo Wennmann. Minimizing tardy processing time on a single machine in near-linear time. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, July 8-12, 2024, Tallinn, Estonia*, volume 297 of *LIPIcs*, pages 64:1–64:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.ICALP.2024.64.

47 Fedor Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization: theory of parameterized preprocessing*. Cambridge University Press, 2019. doi:10.1017/9781107415157.

48 Steven Fortune, John E. Hopcroft, and James Wyllie. The directed subgraph homeomorphism problem. *Theor. Comput. Sci.*, 10:111–121, 1980. doi:10.1016/0304-3975(80)90009-2.

49 Zvi Galil and Oded Margalit. An almost linear-time algorithm for the dense subset-sum problem. *SIAM J. Comput.*, 20(6):1157–1189, 1991. doi:10.1137/0220072.

50 Carla Groenland, Gwenaël Joret, Wojciech Nadara, and Bartosz Walczak. Approximating pathwidth for graphs of small treewidth. *ACM Trans. Algorithms*, 19(2):16:1–16:19, 2023. doi:10.1145/3576044.

51 Godfrey Harold Hardy and Edward Maitland Wright. *An introduction to the theory of numbers*. Oxford university press, 1979.

52 Danny Harnik and Moni Naor. On the compressibility of NP instances and cryptographic applications. *SIAM Journal on Computing*, 39(5):1667–1713, 2010. doi:10.1137/060668092.

53 Danny Hermelin, Stefan Kratsch, Karolina Sołtys, Magnus Wahlström, and Xi Wu. A completeness theory for polynomial (Turing) kernelization. *Algorithmica*, 71(3):702–730, 2015. doi:10.1007/S00453-014-9910-8.

54 Nick Howgrave-Graham and Antoine Joux. New generic algorithms for hard knapsacks. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 235–256. Springer, 2010. doi:10.1007/978-3-642-13190-5_12.

55 Russell Impagliazzo and Moni Naor. Efficient cryptographic schemes provably as secure as subset sum. *J. Cryptol.*, 9(4):199–216, 1996. doi:10.1007/BF00189260.

56 Bart M. P. Jansen, Daniel Lokshtanov, and Saket Saurabh. A near-optimal planarization algorithm. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1802–1811. SIAM, 2014. doi:10.1137/1.9781611973402.130.

57 Bart M. P. Jansen, Shivesh Kumar Roy, and Michał Włodarczyk. On the hardness of compressing weights. In Filippo Bonchi and Simon J. Puglisi, editors, *46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, August 23-27, 2021, Tallinn, Estonia*, volume 202 of *LIPIcs*, pages 64:1–64:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.MFCS.2021.64.

58 Klaus Jansen and Lars Rohwedder. On integer programming and convolution. In Avrim Blum, editor, *10th Innovations in Theoretical Computer Science Conference, ITCS 2019, January 10-12, 2019, San Diego, California, USA*, volume 124 of *LIPIcs*, pages 43:1–43:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPICS.ITCS.2019.43.

59 Ce Jin, Nikhil Vyas, and Ryan Williams. Fast low-space algorithms for subset sum. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1757–1776. SIAM, 2021. doi:10.1137/1.9781611976465.106.

60 Saurabh Joshi, Ruben Martins, and Vasco M. Manquinho. Generalized totalizer encoding for pseudo-boolean constraints. In Gilles Pesant, editor, *Principles and Practice of Constraint Programming - 21st International Conference, CP 2015, Cork, Ireland, August 31 - September 4, 2015, Proceedings*, volume 9255 of *Lecture Notes in Computer Science*, pages 200–209. Springer, 2015. doi:10.1007/978-3-319-23219-5_15.

61 Daniel M. Kane. Unary subset-sum is in logspace. *arXiv*, abs/1012.1336, 2010. `arXiv:1012.1336`.

62 Ken-ichi Kawarabayashi, Yusuke Kobayashi, and Bruce Reed. The disjoint paths problem in quadratic time. *Journal of Combinatorial Theory, Series B*, 102(2):424–435, 2012. `doi:10.1016/J.JCTB.2011.07.004`.

63 Hans Kellerer, Renata Mansini, Ulrich Pferschy, and Maria Grazia Speranza. An efficient fully polynomial approximation scheme for the subset-sum problem. *J. Comput. Syst. Sci.*, 66(2):349–370, 2003. `doi:10.1016/S0022-0000(03)00006-0`.

64 Kim-Manuel Klein, Adam Polak, and Lars Rohwedder. On minimizing tardy processing time, max-min skewed convolution, and triangular structured ilps. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 2947–2960. SIAM, 2023. `doi:10.1137/1.9781611977554.CH112`.

65 Johannes Köbler and Jochen Messner. Is the standard proof system for SAT p-optimal? In Sanjiv Kapoor and Sanjiva Prasad, editors, *Foundations of Software Technology and Theoretical Computer Science, 20th Conference, FSTTCS 2000 New Delhi, India, December 13-15, 2000, Proceedings*, volume 1974 of *Lecture Notes in Computer Science*, pages 361–372. Springer, 2000. `doi:10.1007/3-540-44450-5_29`.

66 Konstantinos Koiliaris and Chao Xu. Faster pseudopolynomial time algorithms for subset sum. *ACM Trans. Algorithms*, 15(3):40:1–40:20, 2019. `doi:10.1145/3329863`.

67 Taisei Kudo and Akimichi Takemura. A lower bound for the Graver complexity of the incidence matrix of a complete bipartite graph. *Journal of Combinatorics*, 3(4):695–708, 2013.

68 Marvin Künnemann, Filip Mazowiecki, Lia Schütze, Henry Sinclair-Banks, and Karol Wegrzycki. Coverability in VASS Revisited: Improving Rackoff's Bound to Obtain Conditional Optimality. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming (ICALP 2023)*, volume 261 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 131:1–131:20, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. `doi:10.4230/LIPIcs.ICALP.2023.131`.

69 Jeffrey C Lagarias and Andrew M Odlyzko. Solving low-density subset sum problems. *Journal of the ACM (JACM)*, 32(1):229–246, 1985. `doi:10.1145/2455.2461`.

70 Eugene L Lawler and J Michael Moore. A functional equation and its application to resource allocation and sequencing problems. *Management science*, 16(1):77–84, 1969.

71 Francis Lazarus and Arnaud de Mesmay. Knots and 3-dimensional computational topology, 2017. URL: `https://pagesperso.g-scop.grenoble-inp.fr/~lazarusf/Enseignement/compuTopo6.pdf`.

72 Andrew Peter Lin. *Solving hard problems in election systems*. Rochester Institute of Technology, 2012. URL: `https://repository.rit.edu/cgi/viewcontent.cgi?article=1332&context=theses`.

73 Daniel Lokshtanov, Pranabendu Misra, Michał Pilipczuk, Saket Saurabh, and Meirav Zehavi. An exponential time parameterized algorithm for planar disjoint paths. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 1307–1316. ACM, 2020. `doi:10.1145/3357713.3384250`.

74 Dániel Marx and Igor Razgon. Fixed-parameter tractability of multicut parameterized by the size of the cutset. *SIAM J. Comput.*, 43(2):355–388, 2014. `doi:10.1137/110855247`.

75 Ralph Merkle and Martin Hellman. Hiding information and signatures in trapdoor knapsacks. *IEEE transactions on Information Theory*, 24(5):525–530, 1978. `doi:10.1109/TIT.1978.1055927`.

76 Marcin Mucha, Karol Węgrzycki, and Michał Włodarczyk. A subquadratic approximation scheme for partition. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 70–88. SIAM, 2019. `doi:10.1137/1.9781611975482.5`.

77 Jesper Nederlof, Erik Jan van Leeuwen, and Ruben van der Zwaan. Reducing a target interval to a few exact queries. In Branislav Rovan, Vladimiro Sassone, and Peter Widmayer, editors, *Mathematical Foundations of Computer Science 2012 - 37th International Symposium, MFCS 2012, Bratislava, Slovakia, August 27-31, 2012. Proceedings*, volume 7464 of *Lecture Notes in Computer Science*, pages 718–727. Springer, 2012. doi:10.1007/978-3-642-32589-2_62.

78 Jesper Nederlof and Karol Wegrzycki. Improving Schroeppel and Shamir's algorithm for subset sum via orthogonal vectors. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 1670–1683. ACM, 2021. doi:10.1145/3406325.3451024.

79 Jean-Louis Nicolas. *On Landau's Function $g(n)$*, pages 228–240. Springer Berlin Heidelberg, Berlin, Heidelberg, 1997. doi:10.1007/978-3-642-60408-9_18.

80 Ramamohan Paturi and Pavel Pudlak. On the complexity of circuit satisfiability. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, pages 241–250, New York, NY, USA, 2010. Association for Computing Machinery. doi:10.1145/1806689.1806724.

81 Michał Pilipczuk and Marcin Wrochna. On space efficiency of algorithms working on structural decompositions of graphs. *ACM Trans. Comput. Theory*, 9(4):18:1–18:36, 2018. doi:10.1145/3154856.

82 David Pisinger. Linear time algorithms for knapsack problems with bounded weights. *Journal of Algorithms*, 33(1):1–14, 1999. doi:10.1006/JAGM.1999.1034.

83 Adam Polak, Lars Rohwedder, and Karol Wegrzycki. Knapsack and subset sum with small items. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 106:1–106:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.ICALP.2021.106.

84 Krzysztof Potepa. Faster deterministic modular subset sum. In Petra Mutzel, Rasmus Pagh, and Grzegorz Herman, editors, *29th Annual European Symposium on Algorithms, ESA 2021, September 6-8, 2021, Lisbon, Portugal (Virtual Conference)*, volume 204 of *LIPIcs*, pages 76:1–76:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.ESA.2021.76.

85 Neil Robertson and Paul D Seymour. Graph minors. XX. Wagner's conjecture. *Journal of Combinatorial Theory, Series B*, 92(2):325–357, 2004. doi:10.1016/J.JCTB.2004.08.001.

86 Derek JS Robinson. *An introduction to abstract algebra*. Walter de Gruyter, 2003.

87 Alexander Schrijver. Finding k disjoint paths in a directed planar graph. *SIAM Journal on Computing*, 23(4):780–788, 1994. doi:10.1137/S0097539792224061.

88 Aleksandrs Slivkins. Parameterized tractability of edge-disjoint paths on directed acyclic graphs. *SIAM J. Discret. Math.*, 24(1):146–157, 2010. doi:10.1137/070697781.

89 Pavel Smirnov, Jeremias Berg, and Matti Järvisalo. Pseudo-boolean optimization by implicit hitting sets. In Laurent D. Michel, editor, *27th International Conference on Principles and Practice of Constraint Programming, CP 2021, Montpellier, France (Virtual Conference), October 25-29, 2021*, volume 210 of *LIPIcs*, pages 51:1–51:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.CP.2021.51.

90 Magnus Wahlström. Abusing the Tutte matrix: An algebraic instance compression for the K-set-cycle problem. In Natacha Portier and Thomas Wilke, editors, *30th International Symposium on Theoretical Aspects of Computer Science, STACS 2013, February 27 - March 2, 2013, Kiel, Germany*, volume 20 of *LIPIcs*, pages 341–352. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2013. doi:10.4230/LIPICS.STACS.2013.341.

91 David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011. URL: http://www.cambridge.org/de/knowledge/isbn/item5759340/?site_locale=de_DE.

92 Michał Włodarczyk. Does subset sum admit short proofs? *arXiv*, abs/2409.03526, 2024. arXiv:2409.03526.