On the parameterized complexity of \( k \)-SUM

Amir Abboud\(^*\)  Kevin Lewi\(^*\)  Ryan Williams\(^*\)\(^†\)

\texttt{abboud@cs.stanford.edu}  \texttt{klewi@cs.stanford.edu}  \texttt{rrw@cs.stanford.edu}

Abstract

In the \( k \)-SUM problem, we are given a set of numbers and asked if there are \( k \) of them which sum to 0. The case of \( k = 3 \) has been extensively studied in computational geometry, and known to be intimately related to many low-dimensional problems. The case of arbitrary \( k \) is the natural parameterization of \textsc{Subset Sum} and is well-known in parameterized algorithms and complexity.

We present new FPT reductions between the \( k \)-SUM problem and the \( k \)-Clique problem, yielding several complexity-theoretic and algorithmic consequences. Our reductions show that \( k \)-SUM on “small” numbers (in the range \([-n^{f(k)}, n^{f(k)}]\) for any computable function \( f \)) is \( W[1] \)-complete, and that \( k \)-SUM (in general) is \( W[1] \)-complete under a common derandomization assumption. These results effectively resolve the parameterized complexity of \( k \)-SUM, initially posed in 1992 by Downey and Fellows in their seminal paper on parameterized intractability [11, 12]. Our method is quite general and applies to other weighted problems as well.

1 Introduction

The area of parameterized algorithms and complexity has seen tremendous progress over the last few years, with the resolution of many longstanding open problems. But the parameterized classification of certain core problems, such as \( k \)-SUM, has remained open.

Definition 1.1 (The \( k \)-SUM Problem). The \( k \)-SUM problem is to determine, given a set of integers \( x_1, \ldots, x_n, t \) and an integer parameter \( k \geq 1 \), if there exists \( S \subseteq [n] \) such that \( |S| = k \) and \( \sum_{i \in S} x_i = t \).

We use the notation \((k, n, M)\)-\textsc{SUM} to denote the special case of the problem where we are given \( n \) integers from the interval \([0, M]\) and wish to find \( k \) that sum to a target \( t \).

The 3-SUM problem is notorious in computational geometry; it is widely conjectured that 3-SUM cannot be solved in time much faster than the simple \( O(n^2) \) time algorithm. The conjecture has many known consequences [15, 13, 16, 4, 3, 23, 22, 17, 1]. The \( O(n^2) \) algorithm for 3-SUM generalizes to a \( \tilde{O}(n^{[k/2]}) \) time algorithm for \( k \)-SUM; based on connections with other problems, this algorithm is also conjectured to be essentially optimal [1]. (It is known that an \( n^{\varepsilon k} \) time algorithm, for all \( \varepsilon > 0 \), would imply that 3-SAT is in \( 2^{o(n)} \) time [23].) In their seminal work on parameterized intractability, Downey and Fellows [11] proved that \( k \)-SUM is contained in the parameterized class \( W[P] \), and is \( W[1] \)-hard. Classifying the problem within a finite level of the...
W-hierarchy was open until 2007, when Buss and Islam [7] managed to prove that \( k\text{-SUM} \in \mathsf{W}[3] \). In this paper, we classify \( k\text{-SUM} \) further.

For problems with weights like \( k\text{-SUM} \), there are inherent tradeoffs between \( M \), the maximum absolute value of the integers, and \( n \), the number of integers. (In fact, \( k\text{-SUM} \) can be solved in \( O(M + n \log n) \) time [13].) But there is a simple randomized reduction from the general \( k\text{-SUM} \) problem to \( k\text{-SUM} \) where \( M \leq n^{2k} \); in brief, one chooses a uniform random prime \( p \in [2, n^{2k}] \), and reduces all numbers in the instance modulo \( p \). This randomized reduction preserves the \( k\text{-SUM} \) answer with high probability [3]. We use the name Small-\( k\text{-SUM} \) to denote the special case of \( k\text{-SUM} \) on such “small numbers.”

Our main theorem is that Small-\( k\text{-SUM} \) is \( \mathsf{W}[1] \)-complete:

**Theorem 1.2.** Let \( f : \mathbb{N} \to \mathbb{N} \) be any computable function such that \( f(k) \geq k \) for all \( k \). The \( k\text{-SUM} \) problem on \( n \) integers from the interval \([0, n^{f(k)}]\) is \( \mathsf{W}[1] \)-complete. (That is, in our notation, \( (k, n, n^{f(k)})\text{-SUM} \) is \( \mathsf{W}[1] \)-complete.)

More precisely, we provide a deterministic parameterized reduction from \( k\text{-SUM} \) on \( n \) numbers in \([0, n^{f(k)}]\) to \( g(f(k)) \cdot n^{o(1)} \) instances of the \( k\text{-Clique} \) problem on \( n \)-node graphs, for some universal computable \( g \). This yields a genuinely new characterization of the class \( \mathsf{W}[1] \), as the class of problems that are parameterized reducible to Small-\( k\text{-SUM} \). Given the power of 3SUM-hardness in algorithms and computational geometry, we believe Theorem 1.2 will also be applicable beyond parameterized complexity. In particular, the proof provides a generic way of translating a weighted problem into an unweighted one, at the cost of introducing pairwise constraints. This already yields several quick applications, given below.

Using the randomized reduction from \( k\text{-SUM} \) to Small-\( k\text{-SUM} \), we can conclude under a common derandomization hypothesis that general \( k\text{-SUM} \) is also \( \mathsf{W}[1] \)-complete:

**Theorem 1.3.** Assuming there is a function \( f : \{0,1\}^* \to \{0,1\} \) computable in time \( 2^n \) that does not have \( 2^{\varepsilon n} \) size SAT-oracle circuits for some \( \varepsilon > 0 \), \( k\text{-SUM} \) is \( \mathsf{W}[1] \)-complete.

The proof of this theorem and relevant definitions are given in Appendix B.

As \( k\text{-SUM} \) was already known to be \( \mathsf{W}[1] \)-hard and in \( \mathsf{W}[3] \), let us give a few reasons why it is interesting to put the problem in \( \mathsf{W}[1] \) (although we admit our techniques may ultimately be more interesting). First, although \( \mathsf{W}[1] \)-complete problems are probably not FPT, prominent problems in \( \mathsf{W}[1] \) (such as \( k\text{-Clique} \)) can still be solved substantially faster than exhaustive search over all \( \binom{n}{k} \) subsets [20]. In contrast, prominent problems in \( \mathsf{W}[2] \) (such as \( k\text{-Dominating Set} \)) do not have such algorithms unless CNF Satisfiability is in \( 2^{\delta n} \) time for some \( \delta < 1 \) [23], which is a major open problem in exact algorithms. Therefore, understanding which parameterized problems lie in \( \mathsf{W}[1] \) is closely related to understanding which problems can be solved faster than exhaustive search. (This intuition, coupled with the fact that \( k\text{-SUM} \) is in \( n^{k/2} \) time, led us to conjecture that \( k\text{-SUM} \) is in \( \mathsf{W}[1] \).) Second, \( k\text{-SUM} \in \mathsf{W}[1] \) means that, after preprocessing, we can verify whether \( k \) given numbers sum to 0 in \( f(k) \) time for some \( f \), as opposed to \( k \cdot \text{poly}(n) \) time. (More precisely, \( \mathsf{W}[1] \) problems are those solvable by nondeterministic FPT algorithms which only perform their \( f(k) \) nondeterministic steps at the end of the computation, for some \( f \) [9].) Finally, showing that a problem is in \( \mathsf{W}[1] \) rather than \( \mathsf{W}[3] \) means that it can be expressed in an apparently weaker logic than before, with fewer quantifiers [14]. That is, putting a problem in \( \mathsf{W}[1] \) decreases the descriptive complexity of the problem.

These results have several interesting consequences.
Node-Weighted $k$-Clique. The most immediate consequence is a new reduction from the node-weighted $k$-clique problem to unweighted $k$-clique (Theorem 3.3). Let $\omega < 2.373$ be the matrix multiplication exponent. We obtain an $O(m^{2\omega/(1+\omega)} \cdot n^{o(1)})$ time algorithm for the problem of detecting whether a node-weighted graph on $m$ edges contains a triangle whose node weights sum to 0. The running time matches the running time of the randomized algorithm in [25]. For node weights restricted to the interval $[-n^{f(k)}, n^{f(k)}]$ for a non-decreasing computable function $f : \mathbb{N} \rightarrow \mathbb{N}$, our algorithm improves on the previously best known deterministic algorithm for the problem [25], which runs in time $O(n^\omega)$, even if the graph is sparse. We refer the reader to Definition 2.3 of the Exact Node-Weight 3-Clique problem in the following theorem.

**Theorem 1.4.** Let $f : \mathbb{N} \rightarrow \mathbb{N}$ be a non-decreasing computable function. The Exact Node-Weight 3-Clique problem on a graph with node weights in $[-\text{poly}(n), \text{poly}(n)]$ and $m$ edges can be solved deterministically in time $O(m^{2\omega/(1+\omega)} \cdot n^{o(1)})$.

In Section 3.4 we prove a more general version of this theorem, which follows from a generalization of the main reduction from $k$-SUM to $k$-Clique.

Other Parameterized Problems. By providing a tight classification of the complexity of $k$-SUM, we are able to conclude new results for the parameterized complexity of various problems in coding theory and graph algorithms.

Bhattacharyya et al. [6] study various extensions of $k$-SUM to vector spaces over finite fields. For appropriately chosen parameters, we can show that these variants are also $W[1]$-complete. For example, the $(k, n)$-LINDEPENDEDENCE$_q$ problem [6] asks if among $n$ length-$m$ vectors over $\mathbb{F}_q$ there are $k$ whose linear span contains a target vector. When the vectors are of length $m = O(\log \log n)$, our reductions allow us to prove that the $(k, n)$-LINDEPENDEDENCE$_q$ problem is in $W[1]$. The $W[1]$-hardness of the problem follows by a reduction from $k$-SUM. Also, Cattaneo and Perdrix [8] define the Weighted Distribution over $\mathbb{F}_q$ problem, which is equivalent to the $k$-SUM problem over a finite field of characteristic $q$. They show the problem is in $W[2]$ and is $W[1]$-hard. By using similar tricks as we do with [6], we can conclude that for certain parameters this problem is also $W[1]$-complete. These results are detailed in Appendix C.

The reduction given from $k$-Clique to Small-$k$-SUM also holds when the graph is either node or edge weighted with weights in the range $[-M, M]$, where $M$ is bounded by $n^{f(k)}$ for a non-decreasing computable function $f : \mathbb{N} \rightarrow \mathbb{N}$, and we wish to find a $k$-clique of weight exactly 0. Hence we can also show that these weighted $k$-clique problems are $W[1]$-complete.

1.1 Some Intuition

Let us give some intuition for Theorem 1.2. Both the containment in $W[1]$ and the hardness for $W[1]$ require new technical ideas. Downey and Fellows [11, 12] proved that $k$-SUM is $W[1]$-hard by a reduction requiring fairly large numbers: they are exponential in $n$, but can still be generated in an FPT way. To prove that $k$-SUM is $W[1]$-hard even when the numbers are only exponential in $k \log n$, we need a much more efficient number-theoretic encoding of $k$-Clique instances. We apply some machinery from additive combinatorics, namely a construction of large sets of integers avoiding trivial solutions to the linear equation $\sum_{i=1}^{k-1} x_i = (k-1)x_k$ [21]. These sets allow us to efficiently “pack” a $k$-Clique instance into a $(\binom{\log n}{k} + k)$-$SUM$ instance on small numbers.

Proving that Small-$k$-SUM is in $W[1]$ takes several technical steps. We provide a parameterized reduction from $k$-SUM on $n$ numbers to an FPT number of $k$-Clique instances on $n$ nodes, such
that some graph has a $k$-clique if and only if the original $n$ numbers have a $k$-SUM. To efficiently reduce from numbers to graphs, we first reduce the numbers to an analogous problem on vectors. We define an intermediate problem $(k, n, M, d)$-VECTOR-SUM, in which one is given a list of $n$ vectors from $\{-kM, \ldots, 0, \ldots, kM\}^d$, and is asked to determine if there are $k$ vectors which sum to the all-zero vector. We give an FPT reduction from $k$-SUM to $(k, n, M, d)$-VECTOR-SUM where $M$ and $d$ are “small” (such that $M^d$ is approximately equal to the original weights of the $k$-SUM instance). Next, we “push” the weights in these vectors onto the edges of a graph connecting the vectors, where the edge weights are much smaller than the original numbers: we reduce from $(k, n, M, d)$-VECTOR-SUM to a certain edge-weighted $k$-clique problem using a polynomial “squaring trick” which creates a graph with edge weights that are closely related in size to $M$. Finally, we reduce from the weighted problem to the unweighted version of the problem by brute-forcing all feasible weight combinations on the edges; as the edge weights are small, this creates $f(k) \cdot n^{o(1)}$ unweighted $k$-Clique instances for some function $f$.

Prior Work Relating $k$-SUM and $k$-Clique. There has been recent work in relating the complexity of $k$-SUM and variations of $k$-Clique for the specific case of $k = 3$. Patrascu [22] shows a tight reduction from 3-SUM to listing 3-cliques; a reduction from listing 3-cliques to 3-SUM is given by Jafargholi and Viola [17]. Vassilevska and Williams [25] consider the exact edge-weight 3-clique problem and give a tight reduction from 3-SUM. For the case of $k > 3$, less is known, as the techniques used for the case of $k = 3$ do not seem to generalize easily. Abboud and Lewi [1] give reductions between $k$-SUM and various exact edge-weighted graph problems where the goal is to find an instance of a specific subgraph whose edge weights sum to 0.

Organization. In Section 2 we give formal definitions of the problems that we consider in our reductions along with notational conventions and the relationship between $k$-SUM and Small-$k$-SUM. In Section 3, we show how to reduce $k$-SUM to $k$-Clique, which allows us to conclude that $k$-SUM is contained in $W[1]$, along with an improved deterministic algorithm for the Exact Node-Weight $k$-Clique problem for small weights. In Section 4 we show our improved reduction from $k$-Clique to $k$-SUM on small weights, which establishes that Small-$k$-SUM is $W[1]$-hard.

2 Preliminaries

In this paper, let $[i, j]$ denote the interval of integers $\{i, \ldots, j\}$. As shorthand, we will define $[n] := [1, n]$. For a vector $v \in \mathbb{Z}^d$, we denote by $v[j]$ the value in the $j^{th}$ coordinate of $v$. We let $0$ denote the all zeros vector. The default domain and range of a function is $\mathbb{N}$.

2.1 Problem Definitions

We formally define the problems involved in our reductions. We always let $k$ be the parameter when considering the parameterized complexity of the problem.

Definition 2.1 (The $k$-Vector-Sum Problem). For integers $k, n, M, d > 0$, the $k$-vector-sum problem $(k, n, M, d)$-VECTOR-SUM is to determine, given an instance $((v_i)_{i \in [n]}, t)$ where each $v_i \in [0, M]^d$ and $t \in [0, kM]^d$, if there exists a size-$k$ subset $S \subseteq [n]$ such that $\sum_{i \in S} v_i = t$. 

Prior Work Relating $k$-SUM and $k$-Clique. There has been recent work in relating the complexity of $k$-SUM and variations of $k$-Clique for the specific case of $k = 3$. Patrascu [22] shows a tight reduction from 3-SUM to listing 3-cliques; a reduction from listing 3-cliques to 3-SUM is given by Jafargholi and Viola [17]. Vassilevska and Williams [25] consider the exact edge-weight 3-clique problem and give a tight reduction from 3-SUM. For the case of $k > 3$, less is known, as the techniques used for the case of $k = 3$ do not seem to generalize easily. Abboud and Lewi [1] give reductions between $k$-SUM and various exact edge-weighted graph problems where the goal is to find an instance of a specific subgraph whose edge weights sum to 0.

Prior Work Relating $k$-SUM and $k$-Clique. There has been recent work in relating the complexity of $k$-SUM and variations of $k$-Clique for the specific case of $k = 3$. Patrascu [22] shows a tight reduction from 3-SUM to listing 3-cliques; a reduction from listing 3-cliques to 3-SUM is given by Jafargholi and Viola [17]. Vassilevska and Williams [25] consider the exact edge-weight 3-clique problem and give a tight reduction from 3-SUM. For the case of $k > 3$, less is known, as the techniques used for the case of $k = 3$ do not seem to generalize easily. Abboud and Lewi [1] give reductions between $k$-SUM and various exact edge-weighted graph problems where the goal is to find an instance of a specific subgraph whose edge weights sum to 0.
Note that the problem "Weight Distribution over $\mathbb{F}_q$" defined in [8] is the $k$-Vector-Sum problem over the field $\mathbb{F}_q$. Furthermore, observe that the $k$-Vector-Sum problem on $\ell$-length vectors with elements from $\mathbb{F}_q$ is isomorphic to $k$-SUM over the finite field $\mathbb{F}_q^\ell$.

**Definition 2.2** (The $k$-Clique Problem). For integers $k, n, m > 0$, the $k$-clique problem is to determine, given an instance of a graph $G$ on $n$ vertices and $m$ edges, if there exists a set of $k$ nodes which form a clique.

**Definition 2.3** (The Exact Node-Weight $k$-Clique Problem). For integers $k, n, m, M > 0$, the exact node-weight $k$-clique problem $(k, n, m, M)$-NW-CLIQUE is to determine, given an instance of a graph $G$ on $n$ vertices and $m$ edges, a weight function $w : V(G) \rightarrow [0, M]$, and a target weight $t \in [0, kM]$, if there exists a set of $k$ nodes which form a clique with total weight $t$.

**Definition 2.4** (The Exact Edge-Weight $k$-Clique Problem). For integers $k, n, m, M > 0$, the exact edge-weight $k$-clique problem $(k, n, m, M)$-EW-CLIQUE is to determine, given an instance of a graph $G$ on $n$ vertices and $m$ edges, a weight function $w : E(G) \rightarrow [-M, M]$, if there exists a set of $k$ nodes which form a clique with total weight 0.

**Definition 2.5** (Small-$k$-SUM). Let $f : \mathbb{N} \rightarrow \mathbb{N}$ be a computable function. Small-$k$-SUM is the $k$-SUM problem restricted to instances of $k$-SUM on $n$ integers in the range $[0, n^{f(k)}]$ and a target integer $t \in [0, kn^{f(k)}]$.

We note that by shifting the target number (or target vector) in the above problems, we can assume without loss of generality that the numbers (or vectors) are in the range $[-M, M]$ and the target value is 0 (or the all zeros vector). In our proofs it will be sometimes easier to work with this setting.

We will mainly focus our attention on the Small-$k$-SUM problem. However, we can show that $k$-SUM on arbitrary numbers can be randomly reduced to Small-$k$-SUM. Furthermore, under the appropriate hardness assumptions, we can derandomize this reduction. This yields a deterministic reduction from $k$-SUM to $k$-Clique. The details of this conditional derandomization result are given in Appendix B.

Additional problem definitions used in Section 4 can be found in Appendix A.

### 2.2 Background on Parameterized Complexity

For an exposition of fixed parameter tractability, reducibility, and the $\mathcal{W}$-hierarchy, we refer the reader to [11, 12, 14]. Here we give a brief overview.

A parameterized problem is defined to be a subset of $\{0, 1\}^* \times \mathbb{N}$. Let $L, L'$ be parameterized problems. $L$ is in the class $\text{FPT}$ if there is an algorithm $A$, constant $c$, and computable function $f : \mathbb{N} \rightarrow \mathbb{N}$, such that on all inputs $y = (x, k)$, $A(y)$ decides whether $y$ is in $L$ and runs in time at most $f(k) \cdot |x|^c$. A $\text{FPT}$ reduction from $L$ to $L'$ is an algorithm $R$ mapping $\{0, 1\}^* \times \mathbb{N}$ to $\{0, 1\}^* \times \mathbb{N}$, such that for all $y = (x, k)$, $R(y) \in L'$ if and only if $y \in L$, $R(y)$ runs in $f(k) \cdot |x|^c$ time (for some $c$ and $f$), and $R(y) = (x', k')$ where $k' \leq g(k)$ for some computable function $g$.

Although it is not the original definition, it is equivalent to define $\mathcal{W}[1]$ to be the class of parameterized problems which have an FPT reduction to $k$-Clique [12, 14].
3 Reducing $k$-SUM to $k$-Clique

In this section we show how to reduce the Small-$k$-SUM problem to the $k$-Clique problem. To do this, we first show a reduction from $k$-SUM to $k$-Vector-SUM, and then from $k$-Vector-SUM to an edge-weighted version of $k$-Clique. Provided that the numbers of the original $k$-SUM instance were small enough, we can brute-force the possibilities of the edge weights in a reduction to the (unweighted) $k$-Clique problem. By putting these lemmas together, we conclude that $k$-SUM on small weights is in $W[1]$, and under the appropriate derandomization assumptions, the general $k$-SUM problem is also in $W[1]$.

3.1 Reducing $k$-SUM to $k$-Vector-SUM

We present the reduction in a general form so that the main reduction from $k$-SUM to $k$-Vector-SUM follows as a corollary. Also, this lemma will be used in the proof of Lemma 3.2.

**Lemma 3.1.** Let $k, p, d, s, M > 0$ be integers such that $k < p$, $p^d \geq kM + 1$, and $s = (k + 1)^{d-1}$. There is a mapping $f : [0, M] \times [s] \times [0, kM] \rightarrow [-kp, kp]^d$ computable in time $O(\text{poly } M + k^d)$ such that for any numbers $x_1, \ldots, x_k \in [0, M]$ and $t \in [0, kM]$, $\sum_{j=1}^{k} x_j = t$ if and only if for some $i \in [s]$, the sum of the vectors $\sum_{j=1}^{k} f(x_j, i, t)$ equals 0 (the all zeros vector).

**Proof.** Note that $p^d > kM$. For an integer $x \in [0, kM]$, define $v_x \in [0, p-1]^d$ to be the representation of $x$ in base $p$, that is, let $v_x[1], \ldots, v_x[d]$ be the unique integers in $[0, p-1]$ such that $x = v_x[1] \cdot p^0 + \cdots + v_x[d] \cdot p^{d-1}$.

For every $(d - 1)$-tuple of numbers $\gamma = (c_1, \ldots, c_{d-1}) \in [0, k]^{d-1}$ and a number $t \in [0, kM]$, define the vector $t_\gamma$ to be such that

$$t_\gamma[j] = \begin{cases} v_t[1] + c_1 \cdot p, & \text{if } j = 1 \\ v_t[j] - c_{j-1} + c_j \cdot p, & \text{if } 2 \leq j \leq d - 1 \\ v_t[d] - c_{d-1}, & \text{if } j = d \end{cases}$$

Intuitively, each tuple $\gamma$ represents a possible choice of the “carries” obtained in each component when computing the sum of their corresponding base-$p$ numbers, and $t_\gamma$ corresponds to the target vector in this base-$p$ representation. Note that there are at most $s = (k + 1)^{d-1}$ such vectors. We arbitrarily number these vectors as $t_1, \ldots, t_s$, and define the mapping $f : [0, M] \times [s] \times [0, kM] \rightarrow [-kp, kp]^d$ as $f(x, i, t) = k \cdot v_x - t_i$.

Consider a set of $k$ numbers $S = \{x_1, \ldots, x_k\} \subset [0, M]$. We claim that $\sum_{j=1}^{k} x_j = t$ if and only if there exists some $i \in [s]$ such that $\sum_{j=1}^{k} f(x_j, i, t) = 0$. First observe that for any vectors $u_1, \ldots, u_k, t \in \mathbb{Z}^d$, $\sum_{j=1}^{k} (k \cdot u_j - t) = 0$ if and only if $\sum_{j=1}^{k} u_j = t$, and therefore, it suffices to show that $\sum_{j=1}^{k} x_j = t$, if and only if there exists some $\gamma \in [0, k]^{d-1}$ such that $\sum_{j=1}^{k} v_{x_j} = t_\gamma$.

For the first direction, assume that $\sum_{i=1}^{k} x_i = t$. Then, we can write the latter equality in the base-$p$ representation as

$$\sum_{j=0}^{d-1} v_{t}[j + 1] \cdot p^j = \sum_{i=1}^{k} \left( \sum_{j=0}^{d-1} v_{x_i}[j + 1] \cdot p^j \right) = \sum_{j=0}^{d-1} \left( \sum_{i=1}^{k} v_{x_i}[j + 1] \right) \cdot p^j.$$
Therefore, \( \sum_{i=1}^{k} v_{xi}[1] \in [0, k(p - 1)] \) must equal \( v_t[1] + c_1 \cdot p \) for some \( c_1 \in [0, k] \), and the sum \( c_1 + \sum_{i=1}^{k} v_{xi}[2] \in [0, kp] \) must equal \( c_2 \cdot p + v_t[2] \) for some \( c_2 \in [0, k] \), and so on. Finally, the sum \( cd-1 + \sum_{i=1}^{k} v_{xi}[d] \) must equal \( v_t[d] \). By letting \( \gamma = (c_1, \ldots, c_{d-1}) \in [0, k]^{d-1} \), these equations imply that \( \sum_{i=1}^{k} v_{xi} = t_\gamma \).

For the other direction, assume that \( \sum_{i=1}^{k} v_{xi} = t_\gamma \) where \( \gamma = (c_1, \ldots, c_{d-1}) \in [0, k]^{d-1} \). Again, we use the base-\( p \) representation to get that \( \sum_{i=1}^{k} x_i = \sum_{j=0}^{d-1} \left( \sum_{i=1}^{k} v_{xi}[j+1] \right) \cdot p^j \), and using the definition of \( t_\gamma \), we can write the sum of the variables in \( S \) as

\[
\sum_{i=1}^{k} x_i = (v_t[1] + c_1 \cdot p^1) + \left( \sum_{i=1}^{k} v_{xi}[1] \right) \cdot p^1 + \cdots + \left( \sum_{i=1}^{k} v_{xi}[d] \right) \cdot p^{d-1}
\]

\[
= v_t[1] + (v_t[2] + c_2 \cdot p^1) \cdot p^1 + \cdots + \left( \sum_{i=1}^{k} v_{xi}[d] \right) \cdot p^{d-1}
\]

\[
= \cdots = v_t[1] + v_t[2] \cdot p^1 + \cdots + v_t[d] \cdot p^{d-1} = t,
\]

which completes the proof.

\[ \square \]

**Corollary 3.1.** Let \( k, p, d, M, n > 0 \) be integers with \( k < p \) and \( p^d \geq kM + 1 \). \( k \)-SUM on \( n \) integers in the range \([0, M]\) can be reduced to \( O(k^d) \) instances of \((k, n, p-1, d)\)-VECTOR-SUM.

### 3.2 Reducing to \( k \)-Clique

Here, we consider a generalization of the \( k \)-SUM problem—namely, the Exact Node-Weight \( k \)-Clique problem. We show a reduction from Exact Node-Weight \( k \)-Clique to Exact Edge-Weight \( k \)-Clique, where the new edge weights are not too much larger than the original node weights. We then show how to reduce to many instances of the unweighted version of the problem, where each instance corresponds to a possible setting of edge weights. Then, we give an application of this general reduction to the Exact Node-Weight \( k \)-Clique problem.

**Lemma 3.2.** Let \( k, p, d, n, m, M > 0 \) be integers such that \( k < p \) and \( p^d \geq kM + 1 \), and let \( M' = O(kdp^2) \). \((k, n, m, M)\)-NW-CLIQUE can be deterministically reduced to \( O(k^d) \) instances of \((k, n, m, M')\)-EW-CLIQUE in time \( O(k^d \cdot n^2 \cdot \text{poly log } M) \).

**Proof.** Let \( G = (V, E) \) be a graph with a node weight function \( w : V \to [0, M] \) and a target number \( t \in [0, kM] \). Recall the mapping \( f : [0, M] \times [s] \times [0, kM] \to [-kp, kp]^d \) from Lemma 3.1. For each \( i \in [s] \), we define an edge weight function \( w_i : E \to [-M', M'] \) as follows. For each edge \( (u, v) \in E \), let \( u = f(w(u), i, t) \) and \( v = f(w(v), i, t) \), and set

\[
w_i(u, v) = \sum_{j=1}^{d} (u[j]^2 + v[j]^2 + 2(k - 1)u[j] \cdot v[j]).
\]

Note that for \( M' = O(kdp^2) \), \( w_i(u, v) \) is bounded in \([-M', M'] \). For \( s = O(k^d) \), we show that there is a \( k \)-clique in \((G, w)\) of node-weight \( t \) if and only if for some \( i \in [s] \), the edge-weighted graph \((G, w_i)\) contains a \( k \)-clique of edge-weight 0. First, observe that for any \( k \) vectors \( v_1, \ldots, v_k \in \mathbb{Z}^d \),

\[
\sum_{i=1}^{k} v_i = 0 \iff \sum_{j=1}^{d} \left( \sum_{i=1}^{k} v_{ij} \right)^2 = 0.
\]
Consider a set of $k$ nodes $S = \{u_1, \ldots, u_k\} \subseteq V$ that forms a $k$-clique in $G$. For any $i \in [s]$ and $u_a, u_b \in S$, let $u_a = f(w_i(u_a), i, t)$ and $u_b = f(w_i(u_b), i, t)$. Then, the edge-weight of $S$ in $(G, w_i)$ can be written as

$$\sum_{1 \leq a < b \leq k} w_i(u_a, w_b) = (k - 1) \sum_{a=1}^{k} \sum_{j=1}^{d} u_a[j]^2 + 2(k - 1) \sum_{1 \leq a < b \leq j} \sum_{j=1}^{d} u_a[j] \cdot u_b[j].$$

Since the sum is evaluated over all pairs $a, b \in [k]$ where $a < b$, the above quantity is equal to

$$(k - 1) \cdot \sum_{j=1}^{d} \left( \sum_{u \in S} f(w(u), i, t)[j] \right)^2.$$  

Therefore, for any $i \in [s]$, the edge-weight of $S$ in $(G, w_i)$, $\sum_{u,v \in S} w_i(u,v)$, is equal to 0 if and only if the sum of the vectors is $\sum_{u \in S} f(w(u), i, t)$, is equal to 0. And, by the properties of the mapping $f$, the latter occurs for some $i \in [s]$ if and only if the node-weight of $S$ in $(G, w)$, $\sum_{u \in S} w(u)$, is equal to $t$, as desired. \hfill \square

We observe that in the graphs produced by the above reduction, all $k$-cliques have non-negative weight. Therefore, we observe that Theorem 3.2 can also be viewed as a reduction to the “minimum-weight” $k$-Clique problem with edge weights, as opposed to the “exact-weight” version that we consider.

**Lemma 3.3.** For all integers $k, M > 0$, there is an $O(M^{(k)} \cdot n^2)$ time reduction from the problem $(k, n, m, M)$-EW-CLIQUE to $O(M^{(k)})$ instances of $k$-Clique on $n$ nodes and $m \cdot (k)$ edges.

**Proof.** Let $G = (V, E)$ be an edge-weighted graph with weight function $w : E \to [-M, M]$. For all possible $\binom{k}{2}$-tuples $\alpha = (\alpha_{1,2}, \alpha_{1,3}, \ldots, \alpha_{k-1,k})$ of integers from $[-M, M]$ such that $\sum_{i<j} \alpha_{i,j} = 0$, we make a $k$-partite graph $G_\alpha$ with $|V|$ nodes in each part—that is, $V(G_\alpha) = \{(v, i) \mid i \in [k], v \in V\}$. We put an edge in $G_\alpha$ between node $(u, i)$ and node $(v, j)$ if and only if $i < j$, $(u, v) \in E$, and $w(u, v) = \alpha_{i,j}$. Observe that $O(M^{(k)})$ graphs $G_\alpha$ are generated. We claim that $G$ has a $k$-clique of weight 0 if and only if at least one of the unweighted graphs $G_\alpha$ contains a $k$-clique.

For the first direction, assume that for some $\alpha = (\alpha_{1,2}, \alpha_{1,3}, \ldots, \alpha_{k-1,k})$, the graph $G_\alpha$ contains a $k$-clique. Then, by the $k$-partite construction, it must have the form $\{(v_1, 1), \ldots, (v_k, k)\} \subseteq V(G_\alpha)$. Now let $S = \{v_1, \ldots, v_k\} \subseteq V$ and note that by the construction of $G_\alpha$, $S$ must form a $k$-clique in $G$ with weight equal to $\sum_{i<j} \alpha_{i,j} = 0$. For the other direction, assume $\{u_1, \ldots, u_k\} \subseteq V$ is a $k$-clique in $G$ of weight 0, then consider the $\binom{k}{2}$-tuple $\alpha = (w(u_1, u_2), w(u_1, u_3), \ldots, w(u_{k-1}, u_k))$. Then, by definition, $G_\alpha$ contains the $k$-clique $\{(u_1, 1), \ldots, (u_k, k)\}$.

3.3 $k$-SUM is in W[1]

Using the above lemmas, we show how to efficiently reduce Small-$k$-SUM to $k$-Clique. Let $f : \mathbb{N} \to \mathbb{N}$ be a non-decreasing computable function, and consider a Small-$k$-SUM instance $(S, t)$ where $S = \{x_1, \ldots, x_n\} \subseteq [0, M]$ and $t \in [0, kM]$ with $M = n^{f(k)}$. Let $G = (V, E)$ be a node-weighted clique on $n$ nodes $V = \{v_1, \ldots, v_n\}$ with weight function $w : V \to [0, M]$ such that $w(v_i) = x_i$ for all $i \in [n]$. Then, the instance $(S, t)$ of Small-$k$-SUM has a solution if and only if the instance $(G, w, t)$ of $(k, n, n^2, M)$-NW-CLIQUE has a solution.
Consider the integers \( d = \lfloor \log n / \log \log n \rfloor \) and \( p = \lfloor k \cdot 2^{f(k)} \cdot \log n \rfloor \), chosen so that \( p^d = (nk)^{f(k)} > kM \). Using Lemma 3.2 we can reduce the instance \((G, w, t)\) of \((k, n, n^2, M)\)-NW-CLIQUE to \(O(k^d) = O(n^{\log k / \log \log n})\) instances of \((k, n, n^2, M')\)-EW-CLIQUE where \(M' = O(2^{f(k)} k^3 \cdot \log^3 n / \log \log n)\). Using Lemma 3.3 we can generate \(g(n, k) = O(n^{\log k / \log \log n} \cdot 2^{k^2 f(k)} k^{3k^2} \log^{3k^2} n)\) instances of \(k\)-Clique on \(n\) nodes and \(n^2\) edges such that at least one has a solution if and only if our Small-\(k\)-SUM instance has a solution. For constant \(k\), note that \(g(n, k) = O(n^\varepsilon)\) for all \(\varepsilon > 0\), and so we have established the following.

**Theorem 3.1.** For any \(c > 2\), if \(k\)-Clique can be solved in time \(O(n^c)\), then for any \(\varepsilon > 0\), Small-\(k\)-SUM can be solved in time \(O(n^{c+\varepsilon})\).

Furthermore, we remark that by applying the above reduction from \(k\)-SUM to \(k\)-Clique to the respective unparameterized versions of these problems, we obtain a reduction from Subset-SUM on arbitrary weights to Exact Edge-Weight Clique on small edge weights. We refer the reader to Appendix A for the definitions of these problems.

**Corollary 3.2.** For any \(\varepsilon > 0\), Subset-SUM on \(n\) numbers in \([-2^{O(n)}, 2^{O(n)}]\) can be reduced to \(2^{2n}\) instances of Exact Edge-Weight Clique on \(n\) nodes with edge weights are in \([-n^{O(1/\varepsilon)}, n^{O(1/\varepsilon)}]\).

Note that Subset-SUM on \(n\) numbers in \([-2^{O(n)}, 2^{O(n)}]\) is as hard as the general case of Subset-SUM (by Lemma B.1), and the fastest known algorithm for Subset-SUM on \(n\) numbers runs in time \(O(2^{n/2})\). The unweighted Max-Clique problem, which asks for the largest clique in a graph on \(n\) nodes, can be solved in time \(O(2^{n/4})\) [24]. Corollary 3.2 shows that even when the edge weights are small, the edge-weighted version of Max-Clique requires time \(\Omega(2^{n/2})\) unless Subset-SUM can be solved faster.

**An FPT Reduction.** We show how to make the above argument an FPT reduction. We can modify the oracle reduction for \(k\)-Clique above to get a many-one reduction to \(k\)-Clique if we simply merge all of the \(g(n, k)\) instances to a single \(k\)-Clique instance—simply consider the graphs from the separate instances as disjoint components of a single graph. The resulting graph has \(n \cdot g(n, k)\) nodes and \(n^2 \cdot g(n, k)\) edges, and contains a \(k\)-clique if and only if one of the original instances of \(k\)-Clique has a solution. Then, we make the following standard argument to appropriately bound \(g(n, k)\) via case analysis. If \(k < \lfloor \log \log n \rfloor\), then \(g(n, k) \leq n^{o(1)} \cdot 2^{f(k) \cdot \text{poly}(k)}\). If \(k \geq \lfloor \log \log n \rfloor\), then since \(n \leq 2^{2^k}\), we have that \(g(n, k) \leq 2^{2^k + f(k) \cdot \text{poly}(k)}\). We have shown the following.

**Theorem 3.2.** Small-\(k\)-SUM is in \(W[1]\).

In Appendix B, we show how to obtain a randomized FPT reduction from \(k\)-SUM to \(k\)-Clique, and how under plausible circuit lower bound assumptions, we can derandomize this reduction to show that \(k\)-SUM \(\in W[1]\). Along with the previously known proof of the \(W[1]\)-hardness of the general \(k\)-SUM problem, this yields Theorem 1.3.

### 3.4 Exact Node-Weight \(k\)-Clique

The reduction described in Section 3.3 implies that the Exact Node-Weight \(k\)-Clique problem can be reduced to \(n^{o(1)}\) instances of \(k\)-Clique, when \(k\) is a fixed constant. We observe that if the input graph has \(m\) edges, then the graphs generated by the reduction have no more than \(k^2 m\) edges. Therefore, we have the following result.
Theorem 3.3. Let $f : \mathbb{N} \rightarrow \mathbb{N}$ be a non-decreasing computable function. If $k$-Clique on $n$ nodes and $m$ edges can be solved in time $T(k, n, m)$, then $(k, n, m, M)$-NW-CLIQUE can be solved in time $T(k, n, m) \cdot n^{o(1)}$.

Also, if the node weights of the input graph are bounded in $[-M, M]$, where $M = n^{f(k)}$ for a non-decreasing computable function $f : \mathbb{N} \rightarrow \mathbb{N}$, then the reduction is deterministic, whereas if $M$ is larger we can decrease the range using a randomized reduction similar to that of Lemma B.1 (which can also be derandomized under the appropriate circuit lower bound assumptions).

It is known that the exact node-weight triangle detection problem on sparse graphs can be solved deterministically in time $O(m^{2/3} + 1)$ [2], and so Theorem 1.4 follows. Also, we note that Czumaj and Lingas [10] achieve the same running time for the “maximum” node-weight version of the problem.

4 From $k$-Clique to Small-$k$-SUM

In this section, we give a new reduction from $k$-clique to $k$-SUM in which the numbers we generate are smaller than in the previously known reductions. This proves that Small-$k$-SUM is W[1]-hard. We can view the result as an alternate proof for the Perfect Code problem, as done by Downey and Fellows [11]. The reduction is given first from $k$-Clique to $k$-Vector-SUM, and then from $k$-Vector-SUM to $k$-SUM.

Lemma 4.1. For an integer $k > 1$, $k$-Clique on $n$ nodes and $m$ edges reduces to an instance of $(k + \binom{k}{2}, kn + \binom{k}{2}m, k \cdot n^{1+\omega(1)}, k^2 + k + 1)$-VECTOR-SUM deterministically in time $O(n^3)$.

Proof. Given a graph $G = (V, E)$ we reduce an instance of $k$-Clique to an instance of $k$-Vector-SUM. First, we construct a $k$-sum-free set $D \subseteq [Q]$ of size $n$, where $Q$ can be bounded by $n^{1+\omega(1)}$, and uniquely associate each vertex $v \in V$ with an element $q_v \in D$. By Lemma A.1, the set $D$ can be constructed in time $\text{poly}(n)$.

Let $T = Q(k - 1) + 1, d = k^2 + k + 1$, and define $\delta_{i\leftarrow j}$ to be the length-$d$ vector (1-indexed) with entry $i$ consisting of the value $j$ and all other entries being 0. We define a mapping $\eta : (V \times [k]) \cup (E \times [k] \times [k]) \rightarrow [0, T]^d$ from the vertices and edges of the graph to length-$d$ vectors over $[0, T]$. For each vertex $v \in V$ and $i \in [k]$, define

$\eta(v, i) := \delta_{i\leftarrow T-(k-1)q_v} + \delta_{d-1}$

and for each edge $e = (u, v) \in E$ and $i, j \in [k]$ with $i < j$, define

$\eta(e, i, j) := \delta_{i\leftarrow q_v} + \delta_{j\leftarrow q_v} + \delta_{k-i+j-1}$

Define the target vector $t = \delta_{d-k} + (\sum_{i=1}^{k} \delta_{i\leftarrow T}) + (\sum_{i, j \in [k], i < j} \delta_{k-i+j-1})$. Then, there is a $k$-clique in the instance $G$ of $k$-Clique on $n$ nodes and $m$ edges if and only if there is a solution to the instance $I = \{\eta(v, i) \mid v \in V, i \in [k]\} \cup \{\eta(e, i, j) \mid e \in E, i, j \in [k], i < j\}, t$ of $(k + \binom{k}{2}, kn + \binom{k}{2}m, T, d)$-VECTOR-SUM.

We now show correctness for this reduction. First, assume that the set $S = \{u_1, \ldots, u_k\} \subseteq V$ forms a $k$-clique in $G$. Denote by $E(S)$ the set of edges between vertices in $S$. We claim that the set of vectors $A = \{\eta(u_j, j) \mid u_j \in S\} \cup \{\eta(e, i, j) \mid e = (u_i, u_j) \in E(S), i < j\}$, which has size $k + \binom{k}{2}$, is a solution to the instance $I$. To see this, consider the sum $\sum_{v \in A} v =$
\[ \sum_{u_j \in S} \eta(u_j, j) + \sum_{e=(u_i,u_j) \in E(S)} \eta(e, i, j) \] and note that it equals \( \sum_{j=1}^{k} (\delta_{j\rightarrow T}-(k-1)q_{v_j} + \delta_{d-1}) + \sum_{j=1}^{k}(k-1)q_{v_j} + \sum_{i,j \in [k], i<j} \delta_{k+i+j-1} = t. \)

For the other direction, consider some subset of vectors \( A \subseteq \{ \eta(v, i) \mid v \in V, i \in [k] \} \cup \{ \eta(e, i, j) \mid e \in E, i, j \in [k], i \neq j \} \) of size \( k + \binom{k}{2} \) for which the sum \( z = \sum_{v \in A} v \) equals the target vector \( t \). Let \( A_V \subseteq V \times [k] \) represent the set of pairs \((u, i)\) for \( u \in V \) and \( i \in [k] \) such that \( \eta(u, i) \in A \). Similarly, let \( A_E \subseteq E \times [k] \times [k] \) represent the set of triples \((e, i, j)\) for \( e \in E \) and \( i, j \in [k] \) such that \( \eta(e, i, j) \in A \).

Recall that \( t[d] = k \) and therefore by the definition of the mapping from vertices to vectors, \( |A_V| \) must equal \( k \). Since \( |A| = |A_V| + |A_E| \), it follows that \( |A_E| = \binom{k}{2} \). Moreover, for all \( i, j \in [k] \) with \( i < j \), since we defined \( t[k \cdot i + j] = 1 \), there is exactly one triple in \( A_E \) of the form \((e, i, j)\). This implies that for every \( j \in [k] \), there must exist at least one pair \((u, j) \in A_V \), for otherwise \( z[j] \) would be the sum of only \((k-1)\) numbers from \( D \subseteq [Q] \), and since \( t[j] = T > Q(k-1) \), this sum cannot equal \( t[j] \). Note that \( |A_V| = k \), and so for each \( j \in [k] \) there is exactly one pair \((u, j) \in A_V \).

Denote this vertex \( u \) by \( u_j \).

For each \( j \in [k] \), let \( e_1, \ldots, e_{k-1} \) be the \( k-1 \) edges in \( E \) such that for each \( i \in [k-1] \), either \((e_i, y, j) \in A_E \) or \((e_i, j, y) \in A_E \) for some \( y \in [k] \). If the former holds, then let \( e_1 = (x, v_i) \) and if the latter holds, let \( e_1 = (v_i, x) \) for some \( x, v_i \in V \). We claim that for \( i \in [k-1] \), \( v_i \) must be identical to \( u_j \). To see this, note that by the definition of the vectors, for all \( j \in [k] \),

\[ z[j] = \eta(u_j, j)[j] + \sum_{i=1}^{k-1} \eta(e,i,j)[j] = T - (k-1)q_{v_j} + \sum_{i=1}^{k-1} q_{v_i}, \]

and since \( z[j] = t[j] = T \) it must be the case that \( \sum_{i=1}^{k-1} q_{v_i} = (k-1)q_{u_j} \). Then, since \( D \) is a \( k \)-sum-free set, this implies that \( u_j = v_i \) for all \( i \in [k-1] \), which concludes the claim.

Therefore, the vectors in \( A_V \) correspond to \( k \) vertices \( V(A) \subseteq V \) and the vectors in \( A_E \) correspond to \( \binom{k}{2} \) edges \( E(A) \subseteq E \) such that precisely \( k-1 \) edges are incident to each vertex in \( V(A) \). Thus, each edge must be incident to two vertices in \( V(A) \), and so the vertices in \( V(A) \) along with the edges in \( E(A) \) form a \( k \)-clique in \( G \). \( \square \)

The following lemma shows a simple reduction from \( k \)-Vector-SUM to \( k \)-SUM.

**Lemma 4.2.** \((k, n, M, d)\)-VECTOR-SUM can be reduced to \( k \)-SUM on \( n \) integers in the range \([0, (kM+1)^d]\) in \( O(n \log M) \) time.

**Proof.** Let \( p = kM + 1 \). Let \( v_1, \ldots, v_n \) be the \( n \) vectors of a \( k \)-Vector-SUM instance with target \( t \). For each vector \( v_i = (v_i[j])_{j=0}^{d-1} \), define the integer \( x_i := \sum_{j=0}^{d-1} v_i[j] \cdot p^j \), and define \( t := \sum_{j=0}^{d-1} t_j \cdot p^j \).

Then the \( k \)-SUM instance \( \{\{x_i\}_{i \in [n]}, t\} \) has a solution if and only if the \( k \)-Vector-SUM instance \( \{(v_i)_{i \in [n]}, t\} \) has a solution. For correctness, let \( S = \{i_1, \ldots, i_k\} \) be a set of \( k \) indices. Then, \( \sum_{i \in S} x_i = \sum_{i \in S} \sum_{j=0}^{d-1} v_i[j] \cdot p^j \). By switching the order of the summations, we have that this quantity is \( \sum_{j=0}^{d-1} \sum_{i \in S} v_i[j] \cdot p^j \). Since \( \sum_{i \in S} v_i[j] \in [0, p-1] \), we conclude by the uniqueness of the representation of \( t \) that \( t_j = \sum_{i \in S} v_i[j] \) for all \( j \in [0, d-1] \). Hence, the \( k \)-Vector-SUM instance of the set \( S \) has a solution described by the index set \( S \) as well. For the other direction, if \( t_j = \sum_{i \in S} v_i[j] \) for all \( j \in [0, d-1] \), then \( \sum_{j=0}^{d-1} t_j \cdot p^j = \sum_{j=0}^{d-1} \sum_{i \in S} v_i[j] \cdot p^j \). The left side is equal to \( t \) and, by a switch of the order of summations, the right side is equal to \( \sum_{i \in S} x_i \). Therefore, \( t = \sum_{i \in S} x_i \) and so the \( k \)-SUM instance has a solution described by the index set \( S \), as desired. \( \square \)

We remark that in some cases, the proof can be changed slightly to yield smaller numbers in the \( k \)-SUM instance produced by the reduction. In particular, when reducing \( k \)-Clique to \( k \)-Vector-SUM, only the numbers in the first \( k \) coordinates can be as large as \( k \cdot n^{1+o(1)} \) while
the numbers in the last $k^2 + 1$ coordinates are bounded by $k$, and therefore, when reducing to $(k + \binom{k}{2}, kn + \binom{k}{2}m, M)$-SUM, the numbers generated can be bounded by $M = k^d \cdot (kn^{1+o(1)})^k \cdot k^{k^2+1} = O(k^{2k^2} \cdot n^{k+o(k)})$.

The above constitutes an FPT reduction, and so we have proved the main theorem of this section.

**Theorem 4.1.** Small-$k$-SUM is $W[1]$-hard.

**References**

[1] Amir Abboud and Kevin Lewi. Exact weight subgraphs and the k-sum conjecture. In *ICALP (1)*, pages 1–12, 2013.

[2] Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *J. ACM*, 42(4):844–856, 1995.

[3] Ilya Baran, Erik D. Demaine, and Mihai Pătraşcu. Subquadratic algorithms for 3SUM. *Algorithmica*, 50(4):584–596, 2008.

[4] Gill Barequet and Sariel Har-Peled. Polygon containment and translational min-hausdorff-distance between segment sets are 3sum-hard. *Int. J. Comput. Geometry Appl.*, 11(4):465–474, 2001.

[5] Felix A Behrend. On sets of integers which contain no three terms in arithmetical progression. *NAS of the United States of America*, 32(12):331, 1946.

[6] Arnab Bhattacharyya, Piotr Indyk, David P. Woodruff, and Ning Xie. The complexity of linear dependence problems in vector spaces. In *ICS*, pages 496–508, 2011.

[7] Jonathan F. Buss and Tarique Islam. Algorithms in the w-hierarchy. *Theory Comput. Syst.*, 41(3):445–457, 2007.

[8] David Cattaneo and Simon Perdrix. The parameterized complexity of domination-type problems and application to linear codes. *CoRR*, abs/1209.5267, 2012.

[9] Yijia Chen, Jorg Flum, and Martin Grohe. Machine-based methods in parameterized complexity theory. *Theoretical Computer Science*, 339(23):167–199, 2005.

[10] Artur Czumaj and Andrzej Lingas. Finding a heaviest vertex-weighted triangle is not harder than matrix multiplication. *SIAM J. Comput.*, 39(2):431–444, 2009.

[11] Rodney G. Downey and Michael R. Fellows. Fixed-parameter intractability. In *Structure in Complexity Theory Conference*, pages 36–49, 1992.

[12] Rodney G. Downey and Michael R. Fellows. Fixed-parameter tractability and completeness II: On completeness for W[1]. *Theor. Comput. Sci.*, 141(1&2):109–131, 1995.

[13] Jeff Erickson. Lower bounds for linear satisfiability problems. In *SODA*, pages 388–395, 1995.

[14] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Springer Verlag, 2006.
Additional Preliminaries

The following problems are referred to in Corollary 3.2. They are simply the unparameterized versions of $k$-SUM and Exact Edge-Weight $k$-Clique, respectively.

**Definition A.1** (The Subset-SUM Problem). The Subset-SUM problem is to determine, given a set of integers $x_1, \ldots, x_n, t$, if there exists a subset $S \subseteq [n]$ such that $\sum_{i \in S} x_i = t$.

**Definition A.2** (The Exact Edge-Weight Clique Problem). For integers $n, m, M > 0$, the Exact Edge-Weight $k$-Clique problem is to determine, given an instance of a graph $G$ on $n$ vertices and $m$ edges, a weight function $w : E(G) \to [-M, M]$, if there exists a set of nodes which form a clique with total weight 0.

We also review some useful combinatorial results that are needed in Section 4.

**Definition A.3** ($(n, k)$-perfect hash function family). For integers $n, k, s > 0$, a set of functions $F = \{f_i\}_{i \in [s]}$ where $f_i : [n] \to [k]$ is a $(n, k)$-perfect hash function family of size $s$ if and only if for every size-$k$ subset $S \subseteq [n]$, there exists some $f_i \in F$ such that $\bigcup_{j \in S} f_i(j) = [k]$. 

13
**Proposition A.1** (cf. [19, Theorem 3]). An \((n, k)\)-perfect hash function family of size \(e^k \cdot k^{O(1)} \log n\) can be constructed deterministically in time \(k^{O(k)} \cdot \text{poly}(n)\).

**Definition A.4** \((k\text{-sum-free set})\). For any \(k \geq 2\), a set \(S \subseteq \mathbb{Z}\) is a \(k\text{-sum-free set}\) if and only if for all (not necessarily distinct) \(x_1, \ldots, x_k \in S\), \(\sum_{i=1}^{k-1} x_i = (k-1) \cdot x_k\) implies that \(x_1 = \cdots = x_k\).

**Lemma A.1.** For any \(\varepsilon > 0\), there exists a \(c > 0\) such that a \(k\text{-sum-free set} S\) of size \(n\) with \(S \subseteq [0, (ckn)^{1+\varepsilon}]\) can be constructed in \(\text{poly}(n)\) time.

We outline a construction of a \(k\text{-sum-free set}\), derived from a small modification to Behrend’s original construction [5].

**Constructing \(k\text{-sum-free Sets.}\)** For a vector \(v\), let \(|v|\) represent the \(\ell_2\) norm of \(v\), and let \(|\langle v_1, v_2 \rangle|\) represent the dot product of two vectors \(v_1\) and \(v_2\). Let \(p = (k-1)d - 1\). Consider the set \(S_r(m, d)\) consisting of all integers \(x\) that can be written as \(x = \sum_{i=0}^{m-1} a_i p^i\), where for all \(i \in [0, m-1]\), \(a_i \in [0, d-1]\) and \(|\langle a_0, \ldots, a_{m-1} \rangle| = \sqrt{r}\). For an integer \(x \in S_r(m, d)\) of the form \(x = \sum_{i \in [0, m-1]} a_i \cdot p^i\), let \(|x|\) represent the \(\ell_2\) norm of the vector \(\langle a_i \rangle_{i \in [0, m-1]}\). By definition, \(|x| = \sqrt{r}||\).

**Claim A.1.** For any vectors \(v_1, \ldots, v_k\), if \(|v_1| = \cdots = |v_k|\) and \(\sum_{i \in [k]} |v_i| = |\sum_{i \in [k]} v_i|\), then \(v_1 = \cdots = v_k\).

**Proof.** Note that
\[
\left| \sum_{i \in [k]} v_i \right|^2 = \sum_{i \in [k]} |\langle v_i, v_i \rangle| + 2 \sum_{i,j \in [k], i < j} |\langle v_i, v_j \rangle| = \sum_{i \in [k]} |v_i|^2 + 2 \sum_{i,j \in [k], i < j} |\langle v_i, v_j \rangle|.
\]

By the Cauchy-Schwarz inequality, \(|\langle v_i, v_j \rangle| \leq |v_i| \cdot |v_j|\). Thus, in order for the above quantity to be equal to \(\left( \sum_{i \in [k]} |v_i| \right)^2\), it must be the case that \(|\langle v_i, v_j \rangle| = |v_i| \cdot |v_j|\) for all \(i, j \in [k]\). Hence, also by Cauchy-Schwarz, \(v_i\) and \(v_j\) are linearly dependent. Then, using the fact that \(|v_i| = |v_j|\), it follows that \(v_i = v_j\) for all pairs \(i, j \in [k]\), which completes the proof.

**Lemma A.2.** For all integers \(k \geq 2\), for all \(\ell \in [0, (m(d-1))^2]\), \(S_\ell\) is \(k\text{-sum-free.}\)

**Proof.** Let \(x_1, \ldots, x_k \in S_\ell(m, d)\) be such that \(\sum_{i \in [k-1]} x_i = (k-1) \cdot x_k\). Then, \(\| \sum_{i \in [k-1]} x_i \| = (k-1) \cdot \|x_k\|\). Since \(\|x_1\| = \|x_2\| = \cdots = \|x_k\|\), it follows that
\[
\sum_{i \in [k-1]} \|x_i\| = \left\| \sum_{i \in [k-1]} x_i \right\|.
\]

Hence, by Claim A.1, it follows that \(x_1 = \cdots = x_k\).

There are \(d^m\) possible settings for \(a_0, \ldots, a_{m-1}\) where each \(a_i \in [0, d-1]\), and the value \(|\langle a_i \rangle_{i \in [0, m-1]}|^2\) lies within the range \([0, (m(d-1))^2]\). Hence, by the pigeonhole principle, there exists some \(r \in [0, (m(d-1))^2]\) such that \(|S_r(m, d)| \geq \frac{d^m}{m(d-1)^2+1} \geq \frac{d^{m-2}}{m}\). Note that \(S_r(m, d) \subset [0, p^m]\). Hence, to obtain a \(k\text{-sum-free set of size} \ n\), we have that \(d = \frac{(m \cdot n)^{1/k}}{m-1}\). Recall that \(p = \Theta(kd)\),
and so \( p^m = O(kmn)^{m^2} \). For any \( \varepsilon > 0 \), we can set \( m = 2/\varepsilon + 2 \) so that \( p^m = (ckn)^{1+\varepsilon} \) for some constant \( c > 0 \). Hence, \( S_r(m,d) \subset [0,(ckn)^{1+\varepsilon}] \). For any \( n \), we can compute, for each \( \ell \in [0,m(d-1)^2] \), the exact size of \( S_\ell(m,d) \), returning the \( \ell \) for which \( |S_\ell(m,d)| \) is maximized. Note that \( m(d-1)^2 = O(n^\varepsilon) \), and the construction of \( S_\ell(m,d) \) requires time linear in \( |S_\ell(m,d)| \). Lemma A.1 follows.

**B A Deterministic FPT Reduction from \( k\)-SUM to \( k\)-Clique**

In this section we show how, by assuming a plausible circuit lower bound assumption, we are able to strengthen our result from Section 3.3 to a reduction from \( k\)-SUM to \( k\)-Clique (as opposed to only reducing Small-\( k\)-SUM to \( k\)-Clique).

To do this, we give a randomized process which takes a \( k\)-SUM instance and outputs a collection of Small-\( k\)-SUM instances which, with high probability, are such that a member of the collection contains a solution if and only if the original \( k\)-SUM instance contains a solution. Then, we apply a theorem which allows us to derandomize this reduction given the appropriate circuit lower bound.

**A Randomized Weight Reduction.** We first define a randomized process (cf. [18, Definition 4.1]) consisting of an algorithm \( F \) along with a predicate \( \pi \). In what follows, the algorithm \( F \) will represent a (randomized) oracle reduction from \( k\)-SUM to Small-\( k\)-SUM which succeeds on some random bit sequences, and the predicate \( \pi \) represents whether or not the input random bit sequence is such that \( F \) produces a valid oracle reduction. Let \( z = O(k \cdot \text{poly} \log n) \), and let \( \mathcal{D} \) represent the domain of \( k\)-SUM instances on \( n \) integers.

Let \( F : \mathcal{D} \times \{0,1\}^z \to \mathcal{D}^k \) be a process which takes as input \( x \) a \( k\)-SUM instance along with a random bit sequence \( r \in \{0,1\}^z \). Let the instance \( x \) describe a \( k\)-SUM instance consisting of a set \( S \) of \( n \) integers in the range \([0,M]\) with target value 0 (without loss of generality). \( F \) uses the sequence of random bits \( r \) to choose a prime \( p \in [2, d^{kn} \log n \log (kM)] \) by picking random integers within the interval until a prime is obtained. Let \( S' \) represent the set of integers derived by taking each integer in \( S \) modulo \( p \). The output of \( F \) is a collection of \( k \) instances of \( k\)-SUM, each with base set \( S' \), and the \( i \)th instance has target value \( i \cdot p \), for \( i \in [0,k-1] \).

Let \( \pi : \mathcal{D} \times \{0,1\}^z \to \{0,1\} \) be a predicate which takes as input \( x \) a \( k\)-SUM instance along with a random bit sequence \( r \in \{0,1\}^z \) and outputs a single bit. The predicate \( \pi(x,r) \) runs \( F(x,r) \) to obtain a collection \( C \) of \( k \) instances of \( k\)-SUM, and outputs 1 if and only if the \( k\)-SUM instance \( x \) has a solution and there exists some \( k\)-SUM instance in \( C \) which does not have a solution, or, the original \( k\)-SUM instance does not have a solution and none of the \( k\)-SUM instances in \( C \) have a solution. The following lemma is folklore, and also informally stated in [3], but we provide a proof here for completeness.

**Lemma B.1.** Let \( k, n, M, M' > 0 \) be integers such that \( M' = O(kn^k \log n \log (kM)) \), and let \( d > 0 \) be a sufficiently large constant. Let \( C \) be the collection of \( k\)-SUM instances output by \( F \) on input \( x \), a \( k\)-SUM instance on \( n \) integers in the range \([0,M]\), and a random bit sequence \( r \in \{0,1\}^z \). Then, each \( k\)-SUM instance in \( C \) is on \( n \) integers in the range \([0,M']\), and \( \pi(x,r) = 1 \) with probability at least \( 1 - 1/(cd) \). for some fixed constant \( c > 0 \).

**Proof.** Let \( S \) be the set of integers of the \( k\)-SUM instance. For a sufficiently large constant \( d > 0 \), for a randomly chosen prime \( p \in [2, d \cdot n^k \log n \log (kM)] \), consider the new set \( S' \) where we take all integers of \( S \) modulo \( p \). Clearly, if \( S \) has a \( k\)-SUM solution, then \( S' \) also has a solution, and
hence, there exists a member of the collection of $k$-SUM instances output by $F$ which contains a solution. For any $k$-tuple of integers in $[0, M]$ with nonzero sum $s$, we have $s \in [kM]$ so $s$ has at most $\log(kM)$ prime factors. By the prime number theorem, for sufficiently large $n$, there are at least $cd \cdot n^k \log(kM)$ primes in this interval for a fixed constant $c > 0$, and so the probability that $p$ is a prime factor of $s$ is at most
\[
\frac{\log(kM)}{cdn^k \log(kM)} \leq \left( \frac{1}{cd \cdot n^k} \right).
\]

By a union bound over all $k$-tuples of numbers from $S'$, the probability of a false positive in $S'$ is at most $1/(cd)$. We have shown that if the $k$-SUM instance $S$ with target value $0$ does not contain a solution, then with probability at least $1 - 1/(cd)$ over the choice of a random prime $p$, $S$ does not contain a $k$-tuple of integers which sum to a multiple of $p$, and hence, there does not exist a member of the collection of $k$-SUM instances output by $F$ which contains a solution.

Finally, we can remove the “modulo $p$” constraint from $S'$ as follows. Treating all elements in $S'$ as integers in $[0, p-1]$, for each $z \in \{-k \cdot p, \ldots, -p, 0, p, \ldots, k \cdot p\}$ we reduce to checking whether there are $x_1, \ldots, x_k \in S'$ such that $\sum_i x_i = z$ (where this sum is over the integers). Such a $k$-tuple exists if and only if $\sum_i x_i$ equals $0 \mod p$. \hfill \qed

**Derandomizing the Reduction.** Let $B$ be an oracle for SAT, and assume there exists a function $f \in \mathcal{E}$ with $B$-oracle circuit complexity $2^{cn}$ for some $c > 0$. We note that, for all $k$-SUM instances $x$ and random bit sequences $r$, $\pi(x, r)$ can be decided by a $B$-oracle circuit of size $n^a$ for some positive constant $a$, since we can construct a circuit which creates and then solves the $k$-SUM instances to check the predicate. We also have by Lemma B.1 that for all $x$, for uniformly sampled $r \in \{0, 1\}^s$, $\Pr[\pi(x, r) = 1]$ with probability $1 - 1/(cd)$, where $c$ is a universal constant and $d$ can be chosen to be arbitrarily large. We now apply the following derandomization theorem due to Klivans and van Melkebeek [18].

**Theorem B.1 (cf. [18, Theorem 4.4]).** Let $B$ be an oracle, $b$ a positive constant, and $\ell : \mathbb{N} \to \mathbb{N}$ a constructible function. Let $(F, \pi)$ be a randomized process using a polynomial number of random bits such that $B$ can efficiently check $(F, \pi)$. If there exists a Boolean function $f \in \mathcal{E}$ such that $C^B_f(\ell(n)) = \Omega(n)$, then there exists a function $G$ computable in $B$ and a constructible function $s(n) = O(\ell^2(n^{O(1)})/\log n)$ such that for any input $x$ of length $n$,
\[
|\Pr_r[\pi(x, r) = 1] - \Pr_s[\pi(x, G(s)) = 1]| \leq O(1/n^b)
\]
where $r$ is uniformly distributed over $\{0, 1\}^{\ell(n)}$ and $s$ over $\{0, 1\}^{s(n)}$.

By applying Theorem B.1 with $\ell(n) = \log n$, there exists a constant $s > 0$ and an efficiently computable function $G$ computable in with seed length $s \log n$ such that for any input $x$ of length $n$, $|\Pr_r[\pi(x, r) = 1] - \Pr_r[\pi(x, G(\sigma)) = 1]| = O(1/n^b)$. By choosing $d = \Omega(n)$, we have that $\Pr[\pi(x, G(\sigma)) = 1] \geq 1 - 1/n$.

**The Deterministic FPT Reduction.** Given an instance $x$ of $k$-SUM, for each $\sigma \in \{0, 1\}^{s \log n}$, let $C_\sigma$ be the collection of instances of $k$-SUM output by $F(x, G(\sigma))$. Note that $C_\sigma$ is in fact a collection of instances of Small-$k$-SUM by Lemma B.1. Hence, as described in Section 3.3, we can reduce each member of $C_\sigma$ to an instance of $k$-Clique—let $S_\sigma$ represent this collection of $k$-Clique
instances, obtained by applying the reduction to each member of $C_\sigma$. Let $S^*$ be a family of $k$-Clique instances obtained by taking the union of $S_\sigma$ over all $\sigma \in \{0,1\}^s \log n$. The original $k$-SUM instance $x$ contains a solution if and only if at least a $1/n$ fraction of the members of $S^*$ contain a $k$-Clique.

To see the correctness of this deterministic reduction, note that at most a $1/n$ fraction of $\sigma \in \{0,1\}^s \log n$ are such that $\pi(x, G(\sigma)) = 0$. For all other seeds $\sigma$, $\pi(x, G(\sigma)) = 1$ and the process $F$ correctly reduces its input $x$ to a collection of $k$-SUM instances. Therefore, if $x$ contains a solution, then all members of $S^*$ will contain a $k$-Clique, and if $x$ does not contain a solution, then at most a $1/n$ fraction of the members of $S^*$ could possibly contain a $k$-Clique. As the constant $s$ in the seed length is fixed, we note that the process $F$ also runs in time polynomial in its input and therefore our reduction is FPT. We have shown the following theorem.

**Theorem B.2.** Let $B$ be an oracle for SAT. If there is a function $f \in E$ with $B$-oracle circuit complexity $2^{\varepsilon n}$ for some $\varepsilon > 0$, then $k$-SUM $\in \mathcal{W}[1]$.

Along with the proof that $k$-SUM is $\mathcal{W}[1]$-hard, this yields Theorem 1.3. We note that the oracle $B$ can be replaced with any oracle sufficient to efficiently compute the predicate $\pi$—for example, Theorem B.2 follows where $B$ is an oracle for $k$-SUM.

## C Linear Dependence Problems in Vector Spaces

In this section we consider the $(k, r)$-TARGETSUM$_q$ problem and the $(k, r)$-LINDEPENDENCE$_q$ problem as defined in Bhattacharyya et al. [6], and show that these problems are $\mathcal{W}[1]$-complete by relating them to $k$-SUM and $k$-Vector-SUM. We note that $(k, r)$-TARGETSUM$_q$ is also called “Weight Distribution over $\mathbb{F}_q$” in [8].

**Definition C.1** (The $(k, r)$-TARGETSUM$_q$ Problem). The $(k, r)$-TARGETSUM$_q$ problem on input $r$ numbers $x_1, \ldots, x_r$ and a target $z \in \mathbb{F}_q$ asks if there exists $x_{i_1}, \ldots, x_{i_k}$ such that $z = \sum_{j=1}^k x_{i_j}$.

**Lemma C.1.** Let $f : \mathbb{Z} \to \mathbb{Z}$ be a computable function, and integers $k, r, q > 1$ such that $kq \leq r f(k)$. The $(k, r)$-TARGETSUM$_q$ problem is $\mathcal{W}[1]$-complete with respect to the parameter $k$.

**Proof.** We reduce $(k, r)$-TARGETSUM$_q$ to $k$ instances of $k$-SUM on $r$ numbers in the range $[0, kq]$. Let $x_1, \ldots, x_n$ be the input numbers and $z$ the target to an instance of $(k, r)$-TARGETSUM$_q$. For each $i \in [0, k-1]$, we create the $k$-SUM instance with numbers $x_1, \ldots, x_r$ and target $z + iq$. As the sum of $k$ numbers in $[0, q-1]$ taken over $\mathbb{F}_q$ must be equal to their sum added with $iq$ taken over the integers (for some $i \in [0, k-1]$), we have that the instance of $(k, r)$-TARGETSUM$_q$ has a solution if and only if one of the $k$ instances of $k$-SUM has a solution. Hence, so long as $kq \leq r f(k)$, $(k, r)$-TARGETSUM$_q$ is in $\mathcal{W}[1]$. For the other direction, we note that we can trivially reduce $k$-SUM on $r$ numbers in the range $[0, M]$ to $(k, r)$-TARGETSUM$_q$ where $q > kM$, as the modulus does not affect the sums of $k$ numbers. Thus, $(k, r)$-TARGETSUM$_q$ is $\mathcal{W}[1]$-hard. \qed

**Definition C.2** (The $(k, r)$-LINDEPENDENCE$_q$ Problem). The $(k, r)$-LINDEPENDENCE$_q$ problem on input $r$ length-$n$ vectors $S = \{v_1, \ldots, v_n\}$ and a target vector $z \in \mathbb{F}_q^n$ asks if there exists $v_{x_1}, \ldots, v_{x_k} \subset S$ such that $z \in \text{span}(v_{x_1}, \ldots, v_{x_k})$.

**Lemma C.2.** Let $f : \mathbb{Z} \to \mathbb{Z}$ be a computable function, and integers $k, r, n, q > 1$ such that $\max(k, q)^n \leq r f(k)$. The $(k, r)$-LINDEPENDENCE$_q$ problem on vectors of length $n$ is $\mathcal{W}[1]$-complete with respect to the parameter $k$. 

17
Proof. To show that the problem is in $W[1]$, we show how to reduce the $(k, r)$-LinDependence problem to $k^n$ instances of $k$-Vector-SUM on $qr$ vectors of length $n$ in the range $[0, q^2]$ such that the $(k, r)$-LinDependence problem has a solution if and only if there is some instance of $k$-Vector-SUM which has a solution. Let $x_1, \ldots, x_r$ be the input vectors to the $(k, r)$-LinDependence problem with target $z$. For each vector $v \in [0, k-1]^n$, we create an instance of $k$-Vector-SUM with input vectors $c \cdot x_i$ for all $c \in [0, q-1]$ and $i \in [r]$. The target of this instance is $z + q \cdot v$. For correctness, if $z$ is in the span of some $x_1, \ldots, x_k$, let $c_1, \ldots, c_k$ be the scalars such that $\sum_{j=1}^k c_j x_{i_j} = z$ in $\mathbb{F}_q$. Let $w \in \mathbb{Z}^d$ be such that $w = \sum_{j=1}^k c_j x_{i_j}$, where addition is done over the integers (as opposed to $\mathbb{F}_q$). Then, each entry $w_i$ can be expressed as $w_i = v_i q + s_i$ for $i \in [n]$, where $v_i \in [0, k-1]$ and $s_i \in [0, q-1]$. Let $v = (v_i)_{i \in [n]}$. Then, the $k$-Vector-SUM instance associated with vector $v$ will have a solution. The reverse direction follows similarly. Our reduction is an FPT reduction when $\max(k, q)^n \leq r f(k)$ using Lemma 3.2 and Lemma 3.3, which shows that $(k, r)$-LinDependence is in $W[1]$.

To show that the $(k, r)$-LinDependence problem is $W[1]$-hard, we can reduce $k$-Vector-SUM to $(k, r)$-LinDependence by simply adding auxiliary coordinates to each of the vectors which require that a solution to the $(k, r)$-LinDependence instance be such that the solution vectors sum to the target vector (as opposed to the target vector lying in the span of the solution vectors). This completes the proof of the lemma. \qed