Exact Weight Subgraphs and the k-Sum Conjecture

Amir Abboud* and Kevin Lewi

Computer Science Department, Stanford University
{abboud,klewi}@cs.stanford.edu

Abstract. We consider the EXACT-WEIGHT-H problem of finding a (not necessarily induced) subgraph H of weight 0 in an edge-weighted graph G. We show that for every H, the complexity of this problem is strongly related to that of the infamous k-SUM problem. In particular, we show that under the k-SUM Conjecture, we can achieve tight upper and lower bounds for the EXACT-WEIGHT-H problem for various subgraphs H such as matching, star, path, and cycle.

One interesting consequence is that improving on the $O(n^3)$ upper bound for EXACT-WEIGHT-4-PATH or EXACT-WEIGHT-5-PATH will imply improved algorithms for 3-SUM, 5-SUM, ALL-PAIRS SHORTEST PATHS and other fundamental problems. This is in sharp contrast to the minimum-weight and (unweighted) detection versions, which can be solved easily in time $O(n^2)$. We also show that a faster algorithm for any of the following three problems would yield faster algorithms for the others: 3-SUM, EXACT-WEIGHT-3-MATCHING, and EXACT-WEIGHT-3-STAR.

1 Introduction

Two fundamental problems that have been extensively studied separately by different research communities for many years are the k-SUM problem and the problem of finding subgraphs of a certain form in a graph. We investigate the relationships between these problems and show tight connections between k-SUM and the "exact-weight" version of the subgraph finding problem.

The k-SUM problem is the parameterized version of the well known NP-complete problem SUBSET-SUM, and it asks if in a set of n integers, there is a subset of size k whose integers sum to 0. This problem can be solved easily in time $O(n^{\lceil k/2 \rceil})$, and Baran, Demaine, and Pătraşcu [3] show how the 3-SUM problem can be solved in time $O(n^2/\log^2 n)$ using certain hashing techniques. However, it has been a longstanding open problem to solve k-SUM for *some* k in time $O(n^{\lceil k/2 \rceil - \epsilon})$ for some $\epsilon > 0$. In certain restricted models of computation, an $\Omega(n^{\lceil k/2 \rceil})$ lower bound has been established initially by Erickson [7] and later generalized by Ailon and Chazelle [1], and recently, Pătraşcu and Williams [16] show that $n^{o(k)}$ time algorithms for all k would refute the Exponential Time Hypothesis. The literature seems to suggest the following hypothesis, which we call the k-SUM Conjecture:

* This work was done while the author was supported by NSF grant CCF-1212372.

F.V. Fomin et al. (Eds.): ICALP 2013, Part I, LNCS 7965, pp. 1–12, 2013.

Conjecture 1 [The k-SUM Conjecture]. *There does not exist a $k \geq 2$, an $\varepsilon > 0$, and a randomized algorithm that succeeds (with high probability) in solving k-SUM in time $O(n^{\lceil \frac{k}{2} \rceil - \varepsilon})$.*

The presumed difficulty of solving k-SUM in time $O(n^{\lceil k/2 \rceil - \varepsilon})$ for any $\varepsilon > 0$ has been the basis of many conditional lower bounds for problems in computational geometry. The $k = 3$ case has received even more attention, and proving 3-SUM-hardness has become common practice in the computational geometry literature. In a recent line of work, Pătraşcu [15], Vassilevska and Williams [17], and Jafargholi and Viola [11] show conditional hardness based on 3-SUM for problems in data structures and triangle problems in graphs.

The problem of determining whether a weighted or unweighted n-node graph has a subgraph that is isomorphic to a fixed k node graph H with some properties has been well-studied in the past [14,12,6]. There has been much work on detection and counting copies of H in graphs, the problem of listing all such copies of H, finding the minimum-weight copy of H, etc. [17,13]. Considering these problems for restricted types of subgraphs H has received further attention, such as for subgraphs H with large indepedent sets, or with bounded treewidth, and various other structures [17,18,13,8]. In this work, we focus on the following subgraph finding problem.

Definition 1 (The EXACT-WEIGHT-H Problem). *Given an edge-weighted graph G, does there exist a (not necessarily induced) subgraph isomorphic to H such that the sum of its edge weights equals a given target value t?*[1]

No non-trivial algorithms were known for this problem. Theoretical evidence for the hardness of this problem was given in [17], where the authors prove that for any H of size k, an algorithm for the exact-weight problem can give an algorithm for the minimum-weight problem with an overhead that is only $O(2^k \cdot \log M)$, when the weights of the edges are integers in the range $[-M, M]$. They also show that improving on the trivial $O(n^3)$ upper bound for EXACT-WEIGHT-3-CLIQUE to $O(n^{3-\varepsilon})$ for any $\varepsilon > 0$ would not only imply an $\tilde{O}(n^{3-\varepsilon})$ algorithm[2] for the minimum-weight 3-CLIQUE problem, which from [19] is in turn known to imply faster algorithms for the canonical ALL-PAIRS SHORTEST PATHS problem, but also an $O(n^{2-\varepsilon'})$ upper bound for the 3-SUM problem, for some $\varepsilon' > 0$. They give additional evidence for the hardness of the exact-weight problem by proving that faster than trivial algorithms for the k-CLIQUE problem will break certain cryptographic assumptions.

Aside from the aforementioned reduction from 3-SUM to EXACT-WEIGHT-3-CLIQUE, few other connections between k-SUM and subgraph problems were known. The standard reduction from Downey and Fellows [5] gives a way to reduce the unweighted k-CLIQUE detection problem to $\binom{k}{2}$-SUM on n^2 numbers. Also, in [15] and [11], strong connections were shown between the 3-SUM problem (or, the similar 3-XOR problem) and listing triangles in unweighted graphs.

[1] We can assume, without loss of generality, that the target value is always 0 and that H has no isolated vertices.

[2] In our bounds, k is treated as a constant. The notation $\tilde{O}(f(n))$ will hide $\mathsf{polylog}(n)$ factors.

1.1 Our Results

In this work, we study the exact-weight subgraph problem and its connections to k-SUM. We show three types of reductions: k-SUM to subgraph problems, subgraphs to other subgraphs, and subgraphs to k-SUM. These results give conditional lower bounds that can be viewed as showing hardness either for k-SUM or for the subgraph problems. We focus on showing implications of the k-SUM Conjecture and therefore view the first two kinds as a source for conditional lower bounds for EXACT-WEIGHT-H, while we view the last kind as algorithms for solving the problem. Our results are summarized in Table 1 and Figure 1, and are discussed below.

Hardness. By embedding the numbers of the k-SUM problem into the edge weights of the exact-weight subgraph problem, using different encodings depending on the structure of the subgraph, we prove four reductions that are summarized in Theorem 1:

Theorem 1. *Let $k \geq 3$. If for all $\varepsilon > 0$, k-SUM cannot be solved in time $O(n^{\lceil k/2 \rceil - \varepsilon})$, then none of the following problems[3] can be solved in time $O(n^{\lceil k/2 \rceil - \delta})$, for any $\delta > 0$:*

- EXACT-WEIGHT-H *on a graph on n nodes, for any subgraph H on k nodes.*
- EXACT-WEIGHT-k-MATCHING *on a graph on $\sqrt{n}$ nodes.*
- EXACT-WEIGHT-k-STAR *on a graph on $n^{(1-1/k)}$ nodes.*
- EXACT-WEIGHT-$(k\text{-}1)$-PATH *on a graph on n nodes.*

An immediate implication of Theorem 1 is that neither 3-STAR can be solved in time $O(n^{3-\varepsilon})$, nor can 3-MATCHING be solved in time $O(n^{4-\varepsilon})$ for some $\varepsilon > 0$, unless 3-SUM can be solved in time $O(n^{2-\varepsilon'})$ for some $\varepsilon' > 0$. We later show that an $O(n^{2-\varepsilon})$ algorithm for 3-SUM for some $\varepsilon > 0$ will imply both an $\tilde{O}(n^{3-\varepsilon})$ algorithm for 3-STAR and an $\tilde{O}(n^{4-2\varepsilon})$ algorithm for 3-MATCHING. In other words, either *all* of the following three statements are true, or *none* of them are:

- 3-SUM can be solved in time $O(n^{2-\varepsilon})$ for some $\varepsilon > 0$.
- 3-STAR can be solved in time $O(n^{3-\varepsilon})$ for some $\varepsilon > 0$.
- 3-MATCHING can be solved in time $O(n^{4-\varepsilon})$ for some $\varepsilon > 0$.

From [17], we already know that solving 3-CLIQUE in time $O(n^{3-\varepsilon})$ for some $\varepsilon > 0$ implies that 3-SUM can be solved in time $O(n^{2-\varepsilon})$ for some $\varepsilon > 0$. By Theorem 1, this would imply faster algorithms for 3-STAR and 3-MATCHING as well.

Another corollary of Theorem 1 is the fact that 4-PATH cannot be solved in time $O(n^{3-\varepsilon})$ for some $\varepsilon > 0$ unless 5-SUM can be solved in time $O(n^{3-\varepsilon'})$ for some $\varepsilon' > 0$. This is in sharp contrast to the unweighted version (and the min-weight version) of 4-PATH, which can both be solved easily in time $O(n^2)$.

Theorem 1 shows that the k-SUM problem can be reduced to the EXACT-WEIGHT-H problem for various types of subgraphs H, and as we noted, this implies connections between the exact-weight problem for different subgraphs. It is natural to ask if for any other subgraphs the exact-weight problems can be related to one another. We will

[3] k-MATCHING denotes the k-edge matching on $2k$ nodes. k-STAR denotes the k-edge star on $k+1$ nodes. k-PATH denotes the k-node path on k-1 edges.

answer this question in the affirmative—in particular, we show a tight reduction from 3-CLIQUE to 4-PATH.

To get this result, we use the edge weights to encode information about the nodes in order to prove a reduction from EXACT-WEIGHT-H_1 to EXACT-WEIGHT-H_2, where H_1 is what we refer to as a "vertex-minor" of H_2. Informally, a vertex-minor of a graph is one that is obtained by edge deletions and node identifications (contractions) for arbitrary pairs of nodes of the original graph (see Section 4 for a formal definition). For example, the triangle subgraph is a vertex-minor of the path on four nodes, which is itself a vertex-minor of the cycle on four nodes.

Theorem 2. *Let H_1, H_2 be subgraphs such that H_1 is a vertex-minor of H_2. For any $\alpha \geq 2$, if* EXACT-WEIGHT-H_2 *can be solved in time $O(n^\alpha)$, then* EXACT-WEIGHT-H_1 *can be solved in time $\tilde{O}(n^\alpha)$.*

Therefore, Theorem 2 allows us to conclude that 4-CYCLE cannot be solved in time $O(n^{3-\varepsilon})$ for some $\varepsilon > 0$ unless 4-PATH can be solved in time $\tilde{O}(n^{3-\varepsilon})$, which cannot happen unless 3-CLIQUE can be solved in time $\tilde{O}(n^{3-\varepsilon})$.

To complete the picture of relations between 3-edge subgraphs, consider the subgraph composed of a 2-edge path along with another (disconnected) edge. We call this the "VI" subgraph and we define the EXACT-WEIGHT-VI problem appropriately. Since the path on four nodes is a vertex-minor of VI, we have that an $O(n^{3-\varepsilon})$ for some $\varepsilon > 0$ algorithm for EXACT-WEIGHT-VI implies an $\tilde{O}(n^{3-\varepsilon})$ algorithm for 4-PATH. In Figure 1, we show this web of connections between the exact-weight 3-edge subgraph problems and its connection to 3-SUM, 5-SUM, and ALL-PAIRS SHORTEST PATHS. In fact, we will soon see that the conditional lower bounds we have established for these 3-edge subgraph problems are all tight. Note that the detection and minimum-weight versions of some of these 3-edge subgraph problems can all be solved much faster than $O(n^3)$ (in particular, $O(n^2)$), and yet such an algorithm for the exact-weight versions for *any* of these problems will refute the 3-SUM Conjecture, the 5-SUM Conjecture, and lead to breakthrough improvements in algorithms for solving ALL-PAIRS SHORTEST PATHS and other important graph and matrix optimization problems (cf. [19])!

Another $O(n^3)$ solvable problem is the EXACT-WEIGHT-5-PATH, and by noting that both 4-CYCLE and VI are vertex-minors of 5-PATH, we get that improved algorithms for 5-PATH will yield faster algorithms for all of the above problems. Moreover, from Theorem 1, 6-SUM reduces to 5-PATH. This established EXACT-WEIGHT-5-PATH as the "hardest" of the $O(n^3)$ time problems that we consider.

We also note that Theorem 2 yields some interesting consequences under the assumption that the k-CLIQUE problem cannot be solved in time $O(n^{k-\varepsilon})$ for some $\varepsilon > 0$. Theoretical evidence for this assumption was provided in [17], where they show how an $O(n^{k-\varepsilon})$ for some $\varepsilon > 0$ time algorithm for EXACT-WEIGHT-k-CLIQUE yields a subexponential time algorithm for the multivariate quadratic equations problem, a problem whose hardness is assumed in post-quantum cryptography.

We note that the 4-clique is a vertex-minor of the 8-node path, and so by Theorem 2, an $O(n^{4-\varepsilon})$ for some $\varepsilon > 0$ algorithm for 8-PATH will yield a faster 4-CLIQUE algorithm. Note that an $O(n^{5-\varepsilon})$ algorithm for 8-PATH already refutes the 9-SUM

Conjecture. However, this by itself is not known to imply faster clique algorithms[4]. Also, there are other subgraphs for which one can only rule out $O(n^{\lceil k/2 \rceil - \varepsilon})$ for $\varepsilon > 0$ upper bounds from the k-SUM Conjecture, while assuming hardness for the k-CLIQUE problem and using Theorem 2, much stronger lower bounds can be achieved.

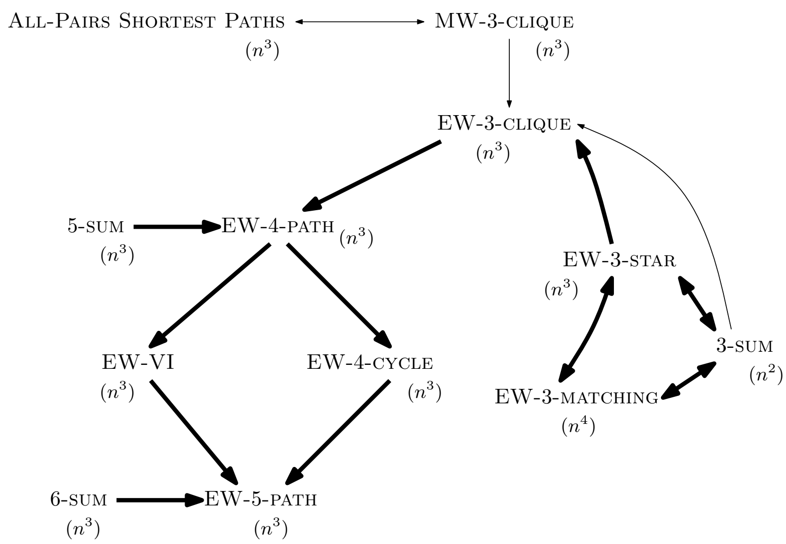

Fig. 1. A diagram of the relationships between EXACT-WEIGHT-H (denoted EW, for small subgraphs H) and other important problems. The best known running times are given for each problem, and an arrow $A \to B$ denotes that A can be tightly reduced to B, in the sense that improving the stated running time for B will imply an improvement on the stated running time for A. The reductions established in this work are displayed in bold, the others are due to [19], [17].

Algorithms. So far, our reductions only show one direction of the relationship between k-SUM and the exact-weight subgraph problems. We now show how to *use* k-SUM to solve EXACT-WEIGHT-H, which will imply that many of our previous reductions are indeed tight. The technique for finding an H-subgraph is to enumerate over a set of d smaller subgraphs that partition H in a certain way. Then, in order to determine whether the weights of these d smaller subgraphs sum up to the target weight, we use d-SUM. We say that $(S, H_1, \ldots, H_d)$ is a *d-separator* of H iff $S, H_1, \ldots, H_d$ partition $V(H)$ and there are no edges between a vertex in H_i and a vertex in H_j for any distinct $i, j \in [d]$.

Theorem 3. *Let $(S, H_1, \ldots, H_d)$ be a d-separator of H. Then,* EXACT-WEIGHT-H *can be reduced to $\tilde{O}(n^{|S|})$ instances of d-SUM each on $\max\{n^{|H_1|}, \ldots, n^{|H_d|}\}$ numbers.*

By using the known d-SUM algorithms, Theorem 3 gives a non-trivial algorithm for exact-weight subgraph problems. The running time of the algorithm depends on the choice of the separator used for the reduction. We observe that the optimal running

[4] It is not known whether the assumption that k-CLIQUE cannot be solved in time $O(n^{k-\varepsilon})$ for any $\varepsilon > 0$ is stronger or weaker than the k-SUM Conjecture.

time can be achieved even when $d = 2$, and can be identified (naively, in time $O(3^k)$) using the following expression. Let

$$\gamma(H) = \min_{(S,H_1,H_2)\text{ is a 2-separator}} \{|S| + \max\{|H_1|, |H_2|\}\}.$$

Corollary 1. EXACT-WEIGHT-H *can be solved in time* $\tilde{O}(n^{\gamma(H)})$.

Corollary 1 yields the upper bounds that we claim in Figure 1 and Table 1. For example, to achieve the $O(n^{\lceil (k+1)/2 \rceil})$ time complexity for k-PATH, observe that we can choose the set containing just the "middle" node of the path to be S, so that the graph $H \setminus S$ is split into two disconnected halves H_1 and H_2, each of size at most $\lceil (k-1)/2 \rceil$. Note that this is the optimal choice of a separator, and so $\gamma(k\text{-PATH}) = 1 + \lceil (k-1)/2 \rceil = \lceil (k+1)/2 \rceil$. It is interesting to note that this simple algorithm achieves running times that match many of our conditional lower bounds. This means that in many cases, improving on this algorithm will refute the k-SUM Conjecture, and in fact, we are not aware of any subgraph for which a better running time is known.

EXACT-WEIGHT-H is solved most efficiently by our algorithm when $\gamma(H)$ is small, that is, subgraphs with small "balanced" separators. Two such cases are when H has a large independent set and when H has bounded treewidth. We show that EXACT-WEIGHT-H can be solved in time $O(n^{k-\lfloor \frac{s}{2} \rfloor})$, if $\alpha(H) = s$, and in time $O(n^{\frac{2}{3}\cdot k + tw(H)})$. Also, we observe that our algorithm can be modified slightly to get an algorithm for the minimization problem.

Theorem 4. *Let H be a subgraph on k nodes, with independent set of size s. Given a graph G on n nodes with node and edge weights, the minimum total weight of a (not necessarily induced) subgraph of G that is isomorphic to H can be found in time* $\tilde{O}(n^{k-s+1})$.

This algorithm improves on the $O(n^{k-s+2})$ time algorithm of Vassilevska and Williams [17] for the MIN-WEIGHT-H problem.

Organization. We give formal definitions in Section 2. In Section 3 we present reductions from k-SUM to exact-weight subgraph problems that prove Theorem 1. In Section 4 we define vertex-minors and outline the proof of Theorem 2. In Section 5, we present the reduction of Theorem 3.

2 Preliminaries and Basic Constructions

For a graph G, we will use $V(G)$ to represent the set of vertices and $E(G)$ to represent the set of edges. The notation $N(v)$ will be used to represent the neighborhood of a vertex $v \in V(G)$.

2.1 Reducibility

We will use the following notion of reducibility between two problems. In weighted graph problems where the weights are integers in $[-M, M]$, n will refer to the number

Table 1. The results shown in this work for EXACT-WEIGHT-H for various H. The second column has the upper bound achieved by our algorithm from Corollary 1. Improvements on the lower bound in the third column will imply improvements on the best known algorithms for the problems in the condition column. These lower bounds are obtained by our redctions, except for the first row which was proved in [17]. For comparison, we give the running times for the (unweighted) detection and minimum-weight versions of the subgraph problems. The last row shows our conditional lower bounds for k-SUM. $\alpha(H)$ represents the independence number of H, $tw(H)$ is its treewidth. The results for the "Any" row hold for all subgraphs on k nodes. ETH stands for the Exponential Time Hypothesis.

Subgraph	Exact	Lower Bound	Condition	Detection	Min
3-CLIQUE	n^3	n^3	3-SUM, APSP	n^ω [10]	n^3
4-PATH	n^3	n^3	3-CLIQUE, 5-SUM	n^2	n^2
k-MATCHING	$n^{2\cdot\lceil\frac{k}{2}\rceil}$	$n^{2\cdot\lceil\frac{k}{2}\rceil}$	k-SUM	n^2	n^2
k-STAR	$n^{\lceil\frac{k}{2}\rceil+1}$	$n^{\lceil\frac{k+1}{2}\rceil}$ $n^{\lceil\frac{k}{2}\rceil\cdot\frac{k}{k-1}}$	(k+1)-SUM k-SUM	n	n^2
k-PATH	$n^{\lceil\frac{k+1}{2}\rceil}$	$n^{\lceil\frac{k+1}{2}\rceil}$	k-SUM	n^2	n^2
k-CYCLE	$n^{\lceil\frac{k}{2}\rceil+1}$	$n^{\lceil\frac{k+1}{2}\rceil}$	k-PATH	n^2	n^3
Any	n^k	$n^{\lceil k/2\rceil}$ $n^{\varepsilon k}$	k-SUM (ETH)	$n^{\omega k/3}$ [12]	n^k
$\alpha(H)=s$	$n^{k-\lfloor\frac{s}{2}\rfloor}$	$n^{\lceil\frac{k}{2}\rceil}$	k-SUM	n^{k-s+1} [13]	n^{k-s+1} [Thm. 4]
$tw(H)=w$	$n^{\frac{2}{3}k+w}$	$n^{\lceil\frac{k}{2}\rceil}$	k-SUM	n^{w+1} [2]	n^{2w} [8]
k-SUM	$n^{\lceil\frac{k}{2}\rceil}$	$n^{\lceil\frac{k}{2}\rceil}$	k-MATCHING, k-STAR	-	-

of nodes times $\log M$. For k-SUM problems where the input integers are in $[-M, M]$, n will refer to the number of integers times $\log M$. In the full version of the paper we formally define our notion of reducibility, which follows the definition of subcubic reductions in [19]. Informally, for any two decision problems A and B, we say that $A \ {}_a\!\leq_b\ B$ if for any $\epsilon > 0$, there exists a $\delta > 0$ such that if B can be solved (w.h.p.) in time $n^{b-\epsilon}$, then A can be solved (w.h.p.) in time $O(n^{a-\delta})$, where n is the size of the input. Note that $\mathsf{polylog}(n)$ factor improvements in solving B may not imply any improvements in solving A. Also, we say that $A \ {}_a\!\equiv_b\ B$ if and only if $A \ {}_a\!\leq_b\ B$ and $B \ {}_b\!\leq_a\ A$.

2.2 The k-SUM Problem

Throughout the paper, it will be more convenient to work with a version of the k-SUM problem that is more structured than the basic formulation. This version is usually referred to as either TABLE-k-SUM or k-SUM$'$, and is known to be equivalent to the basic formulation, up to k^k factors (by a simple extension of Theorem 3.1 in [9]). For convenience, and since $f(k)$ factors are ignored in our running times, we will refer to this problem as k-SUM.

Definition 2 **(k-SUM).** *Given k lists $L_1, \dots, L_k$ each with n numbers where $L_i = \{x_{i,j}\}_{j \in [n]} \subseteq \mathbb{Z}$, do there exist k numbers $x_{1,a_1}, \dots, x_{k,a_k}$, one from each list, such that $\sum_{i=1}^{k} x_{i,a_i} = 0$?*

In our proofs, we always denote an instance of k-SUM by $L_1, \dots, L_k$, where $L_i = \{x_{i,j}\}_{j \in [n]} \subseteq \mathbb{Z}$, so that $x_{i,j}$ is the j^{th} number of the i^{th} list L_i. We define a k-solution to be a set of k numbers $\{x_{i,a_i}\}_{i \in [k]}$, one from each list. The sum of a k-solution $\{x_{i,a_i}\}_{i \in [k]}$ will be defined naturally as $\sum_{i=1}^{k} x_{i,a_i}$.

In [15], Pătraşcu defines the CONVOLUTION-3-SUM problem. We consider a natural extension of this problem.

Definition 3 **(CONVOLUTION-k-SUM).** *Given k lists $L_1, \dots, L_k$ each with n numbers, where $L_i = \{x_{i,j}\}_{j \in [n]} \subseteq \mathbb{Z}$, does there exist a k-solution $\{x_{i,a_i}\}_{i \in [k]}$ such that $a_k = a_1 + \cdots + a_{k-1}$ and $\sum_{i=1}^{k} x_{i,a_i} = 0$?*

Theorem 10 in [15] shows that 3-SUM ${}_2{\le}_2$ CONVOLUTION-3-SUM. By generalizing the proof, we show the following useful lemma (see proof in the full version of the paper).

Lemma 1. *For all $k \geq 2$, k-SUM ${}_{\lceil k/2 \rceil}{\le}_{\lceil k/2 \rceil}$ CONVOLUTION-k-SUM.*

2.3 H-Partite Graphs

Let H be a subgraph on k nodes with $V(H) = \{h_1, \dots, h_k\}$.

Definition 4 **(H-partite graph).** *Let G be a graph such that $V(G)$ can be partitioned into k sets $P_{h_1}, \dots, P_{h_k}$, each containing n vertices. We will refer to these k sets as the* super-nodes *of G. A pair of super-nodes (P_{h_i}, P_{h_j}) will be called a* super-edge *if $(h_i, h_j) \in E(H)$. Then, we say that G is H-*partite *if every edge in $E(G)$ lies in some super-edge of G.*

We denote the set of vertices of an H-partite graph G by $V(G) = \{v_{i,j}\}_{i \in [k], j \in [n]}$, where $v_{i,j}$ is the j^{th} vertex in super-node P_{h_i}. We will say that G is the *complete H-partite graph* when $(v_{i,a}, v_{j,b}) \in E(G)$ if and only if (P_{h_i}, P_{h_j}) is a super-edge of G, for all $a, b \in [n]$.

An *H-subgraph* of an H-partite graph G, denoted by $\chi = \{v_{i,a_i}\}_{i \in [k]} \subseteq V(G)$, is a set of vertices for which there is exactly one vertex v_{i,a_i} from each super-node P_{h_i}, where a_i is an index in $[n]$. Given a weight function $w : (V(G) \cup E(G)) \to \mathbb{Z}$ for the nodes and edges of G, the total weight of the subgraph χ is defined naturally as

$$w(\chi) = \sum_{h_i \in V(H)} w(v_{i,a_i}) + \sum_{(h_i,h_j) \in E(H)} w(v_{i,a_i}, v_{j,a_j}).$$

Now, we define a more structured version of the EXACT-WEIGHT-H problem which is easier to work with.

Definition 5 (The EXACT-H Problem). *Given a complete H-partite graph graph G with a weight function $w : (V(G) \cup E(G)) \to \mathbb{Z}$ for the nodes and edges, does there exist an H-subgraph of total weight 0?*

In the full version of the paper, we prove the following lemma, showing that the two versions of the EXACT-WEIGHT-H problem are reducible to one another in a tight manner. All of our proofs will use the formulation of EXACT-H, yet the results will also apply to EXACT-WEIGHT-H. Note that our definitions of H-partite graphs uses ideas similar to color-coding [2].

Lemma 2. *Let* $\alpha > 1$. EXACT-WEIGHT-H ${}_{\alpha}\!\equiv_{\alpha}$ EXACT-H.

3 Reductions from k-SUM to Subgraph Problems

In this section we prove Theorem 1 by proving four reductions, each of these reductions uses a somewhat different way to encode k-SUM in the structure of the subgraph. First, we give a generic reduction from k-SUM to EXACT-H for an arbitrary H on k nodes. We set the node weights of the graph to be the numbers in the k-SUM instance, in a certain way. See the full version of the paper for a detailed proof.

Lemma 3 (k-SUM ${}_{\lceil k/2 \rceil}\!\leq_{\lceil k/2 \rceil}$ EXACT-H). *Let H be a subgraph with k nodes. Then, k-SUM on n numbers can be reduced to a single instance of* EXACT-H *on kn vertices.*

We utilize the edge weights of the graph, rather than the node weights, to prove a tight reduction to k-MATCHING. See the full version of the paper for a detailed proof.

Lemma 4 (k-SUM ${}_{\lceil k/2 \rceil}\!\leq_{2 \cdot \lceil k/2 \rceil}$EXACT-$k$-MATCHING). *Let H be the k-MATCHING subgraph. Then, k-SUM on n numbers can be reduced to a single instance of* EXACT-H *on $k\sqrt{n}$ vertices.*

Another special type of subgraph which can be shown to be tightly related to the k-SUM problem is the k-edge star subgraph.

Lemma 5 (k-SUM ${}_{\lceil \frac{k}{2} \rceil}\!\leq_{\lceil \frac{k}{2} \rceil \cdot \frac{k}{k-1}}$ EXACT-k-STAR). *Let H be the k-STAR subgraph, and let $\alpha > 2$. If* EXACT-H *can be solved in $O(n^{\alpha})$ time, then k-SUM can be solved in $\tilde{O}(n^{(1-1/k)\cdot\alpha})$ time.*

To prove the lemma we define the problem k-SUM^n to be the following. Given a sequence of n k-SUM instances, each on n numbers, does there exist an instance in the sequence that has a solution of sum 0? Then, we prove two claims. The first showing that k-SUM^n can be reduced to k-STAR, by associating a node in the graph with a k-SUM instance, and setting the weights of the edges incident to it to the numbers of that instance, such that each k-solution of that instance will be a k-STAR centered on that node. The second claim shows that k-SUM can be reduced to k-SUM^n by showing a self reduction for k-SUM. We use a hashing scheme due to Dietzfelbinger [4], to hash the numbers into $O(n^{1/k})$ buckets with $O(n^{1-1/k})$ numbers each, such that for every

$k-1$ numbers, the numbers that can complete a k-solution of sum 0 can only lie in certain k buckets. We utilize this to generate $O((n^{1/k})^{k-1})$ instances of k-SUM, each with $O(n^{1-1/k})$ numbers, such that one of them will have a solution iff the original k-SUM instance had a solution. This is a k-SUM^N instance, for $N = O(n^{1-1/k})$. A more detailed proof is given in the full version of the paper.

Our final reduction between k-SUM and EXACT-H for a class of subgraphs H is as follows. First, define the k-PATH subgraph H to be such that $V(H) = \{h_1, \ldots, h_k\}$ and $E(H) = \{(h_1, h_2), (h_2, h_3), \ldots, (h_{k-1}, h_k)\}$.

Lemma 6 (k-SUM $_{\lceil \frac{k}{2} \rceil}\!\leq_{\lceil \frac{k}{2} \rceil}$ EXACT-(k-1)-PATH). *Let H be the k-PATH subgraph. If* EXACT-H *can be solved in time $O(n^{\lceil k/2 \rceil - \varepsilon})$ for some $\varepsilon > 0$, then k+1-SUM can be solved in time $O(n^{\lceil k/2 \rceil - \varepsilon'})$, for some $\varepsilon' > 0$.*

Proof. We prove that an instance of CONVOLUTION-(k+1)-SUM on n numbers can be reduced to a single instance of EXACT-k-PATH, and by applying Lemma 1, this completes the proof. Given $k+1$ lists $L_1, \ldots, L_{k+1}$ each with n numbers as the input to CONVOLUTION-(k+1)-SUM, we will construct a complete H-partite graph G on kn nodes. For every r and s such that $r - s \in [n]$, for all $i \in [k]$, define the edge weights of G in the following manner.

$$w(v_{i,r}, v_{i+1,s}) = \begin{cases} x_{1,r} + x_{2,s-r}, & \text{if } i = 1 \\ x_{i+1,s-r}, & \text{if } 1 < i < k \\ x_{k,s-r} + x_{k+1,s}, & \text{if } i = k \end{cases}$$

Otherwise, if $r - s \notin [n]$, we set $w(v_{i,r}, v_{i+1,s}) = -\infty$ for all $i \in [k]$. Now to see the correctness of the reduction, take any H-subgraph $\{v_{i,a_i}\}_{i \in [k]}$ of G, and consider the $(k+1)$-solution $\{x_{i,b_i}\}_{i \in [k+1]}$, where $b_1 = a_1$, $b_{k+1} = a_k$, and for $2 \leq i \leq k$, $b_i = a_i - a_{i-1}$. First, note that the $(k+1)$-solution satisfies the property that $b_1 + \ldots + b_k = b_{k+1}$. Now, note that its total weight is $\sum_{i=1}^{k} w(v_{i,a_i}, v_{i+1,a_{i+1}}) = x_{1,a_1} + x_{2,a_2-a_1} + x_{3,a_3-a_2} + \ldots + x_{k-1,a_{k-1}-a_{k-2}} + x_{k,a_k-a_{k-1}} + x_{k+1,a_k} = \sum_{i=1}^{k+1} x_{i,b_i}$ which is exactly the sum of the (k+1)-solution. For the other direction, consider the (k+1)-solution $\{x_{i,b_i}\}_{i \in [k+1]}$ for which $b_{k+1} = b_1 + \ldots + b_{k+1}$. Then, the H-subgraph $\{v_{i,a_i}\}_{i \in [k]}$, where $a_i = b_1 + \ldots + b_i$, has total weight $\sum_{i=1}^{k+1} x_{i,b_i}$. Therefore, there is a k-solution of sum 0 iff there is an H-subgraph in G of total weight 0. □

4 Relationships between Subgraphs

In this section we outline the proof of Theorem 2 showing that EXACT-H_1 can be reduced to EXACT-H_2 if H_1 is a *vertex-minor* of H_2. The complete proof is in the full version of the paper, along with an additional observation that gives a reverse reduction. We start by defining vertex-minors.

Definition 6 (Vertex-Minor). *A graph H_1 is called a vertex-minor of graph H_2, and denoted $H_1 \leq_{\mathsf{vm}} H_2$, if there exists a sequence of subgraphs $H^{(1)}, \ldots, H^{(\ell)}$ such that $H_1 = H^{(1)}$, $H_2 = H^{(\ell)}$, and for every $i \in [\ell - 1]$, $H^{(i)}$ can be obtained from $H^{(i+1)}$ by either*

- *Deleting a single edge* $e \in E(H^{(i+1)})$*, or*
- *Contracting two nodes*[5] $h_j, h_k \in V(H^{(i+1)})$ *to one node* $h_{jk} \in V(H^{(i)})$*, such that* $N(h_{jk}) = N(h_j) \cup N(h_k)$*.*

To prove Theorem 2 it suffices to show how to reduce EXACT-H_1 to EXACT-H_2 when H_1 is obtained by either a single edge deletion or a single edge contraction. The edge deletion reduction is easy, and the major part of the proof will be showing the contraction reduction. The main observation is that we can make two copies of nodes and change their node weights in a way such that any H_2-subgraph of total weight 0 that contains one of the copies will have to contain the other. This will allow us to claim that the subgraph obtained by replacing the two copies of a node with the original will be an H_1-subgraph of total weight 0.

5 Reductions to k-SUM (and Upper Bounds)

In this section we explain the reduction used to prove Theorem 3. The full version of the paper has the proof of correctness, followed by a discussion of the algorithmic implications of Theorem 3, including Corollary 1, and Theorem 4.

Recall the definition of a d-separator $(S, H_1, \ldots, H_d)$ of graph H. To solve EXACT-H, we do the following. For each choice of an S-subgraph $\chi_S = \{v_{j,a_j}\}_{h_j \in S}$ of G, we create an instance of d-SUM where the lists $L_1, \ldots, L_d$ are of size $\max\{n^{|H_1|}, \ldots, n^{|H_d|}\}$. We construct L_i by iterating over each H_i-subgraph $\chi_{H_i} = \{v_{j,a_j}\}_{h_j \in H_i}$ and adding the total weight of the $(H_i \cup S)$-subgraph $\chi_{H_i} \cup \chi_S$ as an integer to L_i. This process is repeated for each L_i, and the target of the d-SUM instance is set to be $w(\chi_S) \cdot (d-1)$. Then, if the input graph has an H-subgraph of total weight 0, then for some choice of an S-subgraph, the corresponding d-SUM instance will have a d-solution with sum equal to the target value.

6 Conclusions

We conclude with two interesting open questions:

1. Perhaps the simplest subgraph for which we cannot give tight lower and upper bounds is the 5-CYCLE subgraph. Can we achieve $O(n^{4-\varepsilon})$ for some $\varepsilon > 0$ without breaking the k-SUM Conjecture, or can we prove that it is not possible?
2. Can we prove that EXACT-WEIGHT-4-PATH ${}_3\!\leq_3$ EXACT-WEIGHT-3-STAR? This would show that breaking the 3-SUM Conjecture will imply an $O(n^{3-\varepsilon})$ for some $\varepsilon > 0$ algorithm for ALL-PAIRS SHORTEST PATHS.

Acknowledgements. The authors would like to thank Ryan and Virginia Williams for many helpful discussions and for sharing their insights, and Hart Montgomery for initiating the conversation that led up to this work. We would also like to thank the anonymous reviewers for their comments and suggestions.

[5] The difference between our definition of vertex-minor and the usual definition of a graph minor is that we allow contracting two nodes that are not necessarily connected by an edge.

References

1. Ailon, N., Chazelle, B.: Lower bounds for linear degeneracy testing. J. ACM 52(2), 157–171 (2005)
2. Alon, N., Yuster, R., Zwick, U.: Color-coding. J. ACM 42(4), 844–856 (1995)
3. Baran, I., Demaine, E.D., Pătraşcu, M.: Subquadratic algorithms for 3SUM. Algorithmica 50(4), 584–596 (2008); In: Dehne, F., López-Ortiz, A., Sack, J.-R. (eds.) WADS 2005. LNCS, vol. 3608, pp. 409–421. Springer, Heidelberg (2005)
4. Dietzfelbinger, M.: Universal hashing and k-wise independent random variables via integer arithmetic without primes. In: Puech, C., Reischuk, R. (eds.) STACS 1996. LNCS, vol. 1046, pp. 569–580. Springer, Heidelberg (1996)
5. Downey, R.G., Fellows, M.R.: Fixed-parameter tractability and completeness ii: On completeness for w[1] (1995)
6. Eisenbrand, F., Grandoni, F.: On the complexity of fixed parameter clique and dominating set. Theor. Comput. Sci. 326(1-3), 57–67 (2004)
7. Erickson, J.: Lower bounds for linear satisfiability problems. In: Clarkson, K.L. (ed.) SODA, pp. 388–395. ACM/SIAM (1995)
8. Fomin, F.V., Lokshtanov, D., Raman, V., Saurabh, S., Raghavendra Rao, B.V.: Faster algorithms for finding and counting subgraphs. J. Comput. Syst. Sci. 78(3), 698–706 (2012)
9. Gajentaan, A., Overmars, M.H.: On a class of $o(n^2)$ problems in computational geometry. Computational Geometry 5(3), 165–185 (1995)
10. Itai, A., Rodeh, M.: Finding a minimum circuit in a graph. In: STOC 1977, pp. 1–10. ACM, New York (1977)
11. Jafargholi, Z., Viola, E.: 3sum, 3xor, triangles. Electronic Colloquium on Computational Complexity (ECCC) 20, 9 (2013)
12. Kloks, T., Kratsch, D., Müller, H.: Finding and counting small induced subgraphs efficiently. Inf. Process. Lett. 74(3-4), 115–121 (2000)
13. Kowaluk, M., Lingas, A., Lundell, E.-M.: Counting and detecting small subgraphs via equations and matrix multiplication. In: SODA 2011, pp. 1468–1476. SIAM (2011)
14. Neetil, J., Poljak, S.: On the complexity of the subgraph problem. Commentationes Mathematicae Universitatis Carolinae 026(2), 415–419 (1985)
15. Pătraşcu, M.: Towards polynomial lower bounds for dynamic problems. In: Proc. 42nd ACM Symposium on Theory of Computing (STOC), pp. 603–610 (2010)
16. Pătraşcu, M., Williams, R.: On the possibility of faster sat algorithms. In: Proc. 21st ACM/SIAM Symposium on Discrete Algorithms (SODA), pp. 1065–1075 (2010)
17. Vassilevska, V., Williams, R.: Finding, minimizing, and counting weighted subgraphs. In: Mitzenmacher, M. (ed.) STOC, pp. 455–464. ACM (2009)
18. Williams, R.: Finding paths of length k in $\mathrm{o}^*(2^\mathrm{k})$ time. Inf. Process. Lett. 109(6), 315–318 (2009)
19. Williams, V.V., Williams, R.: Subcubic equivalences between path, matrix and triangle problems. In: FOCS, pp. 645–654. IEEE Computer Society (2010)