

Exploring the Approximability Landscape of 3SUM



Karl Bringmann  

Saarland University, Saarland Informatics Campus, Saarbrücken, Germany

Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany

Ahmed Ghazy  

CISPA Helmholtz Center for Information Security, Saarbrücken, Germany

Marvin Künnemann  

Karlsruhe Institute of Technology, Germany

Abstract

Since an increasing number of problems in P have conditional lower bounds against exact algorithms, it is natural to study which of these problems can be efficiently approximated. Often, however, there are many potential ways to formulate an approximate version of a problem. We ask: How sensitive is the (in-)approximability of a problem in P to its precise formulation?

To this end, we perform a case study using the popular 3SUM problem. Its many equivalent formulations give rise to a wide range of potential approximate relaxations. Specifically, to obtain an approximate relaxation in our framework, one can choose among the options: (a) 3SUM or Convolution 3SUM, (b) monochromatic or trichromatic, (c) allowing under-approximation, over-approximation, or both, (d) approximate decision or approximate optimization, (e) single output or multiple outputs and (f) implicit or explicit target (given as input).

We show general reduction principles between some variants and find that we can classify the remaining problems (over polynomially bounded positive integers) into three regimes:

1. $(1 + \epsilon)$ -approximable in near-linear time $\tilde{O}(n + 1/\epsilon)$,
2. $(1 + \epsilon)$ -approximable in near-quadratic time $\tilde{O}(n/\epsilon)$ or $\tilde{O}(n + 1/\epsilon^2)$, or
3. non-approximable, i.e., requiring time $n^{2 \pm o(1)}$ even for *any* approximation factor.

In each of these three regimes, we provide matching upper and conditional lower bounds.

To prove our results, we establish two results that may be of independent interest: Over polynomially bounded integers, we show subquadratic equivalence of (min, +)-convolution and polyhedral 3SUM, and we prove equivalence of the Strong 3SUM conjecture and the Strong Convolution 3SUM conjecture.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Fine-grained Complexity, Conditional Lower Bounds, Approximation Schemes, Min-Plus Convolution

Digital Object Identifier 10.4230/LIPIcs.ESA.2024.34

Funding *Karl Bringmann*: This work is part of the project TIPEA that has received funding from the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (grant agreement No. 850979).

Marvin Künnemann: Research partially supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 462679611.

1 Introduction

In the last decade, work in fine-grained complexity theory has produced a growing list of conditional lower bounds for problems in P, exposing barriers for obtaining almost-linear-time algorithms for these problems. Such conditional lower bounds are usually based on our inability to solve certain core problems significantly faster than exhaustive search. Among the most popular such core problems are 3SUM, All-Pairs-Shortest-Paths, and Orthogonal Vectors, see [45] for an excellent survey.



© Karl Bringmann, Ahmed Ghazy, and Marvin Künnemann;
licensed under Creative Commons License CC-BY 4.0

32nd Annual European Symposium on Algorithms (ESA 2024).

Editors: Timothy Chan, Johannes Fischer, John Iacono, and Grzegorz Herman; Article No. 34; pp. 34:1–34:15

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

As for NP-hard problems, the difficulty of solving a polynomial-time problem *exactly* invites us to study whether we can at least efficiently *approximate* it. Here, there are usually many possible approximation variants to consider that are often equally reasonable. At times, the question which particular notion of approximation is studied is simply a consequence of which guarantees researchers have been able to show for a given algorithmic approach. However, obtaining a corresponding *hardness* of approximation in P (under a given notion of approximation) constitutes an even younger field of research (see, e.g. [44, 1, 8, 12, 6, 24, 35, 4]) and many questions are still largely unresolved.

In this work, we set out to address a very basic and general question:

How sensitive is the (in-)approximability of a problem in P to its particular formulation?

To approach this broad question in a meaningful way, we perform a comprehensive case study on one of the most important problems in fine-grained complexity theory: the 3SUM problem (given a set A of n numbers, are there $a, b, c \in A$ such that $a + b + c = 0$?).

This is one of the first problems considered for studying hardness in P [28], and has been used to give a variety of strong conditional lower bounds, see e.g. [46, 43, 7, 9, 2, 26, 16, 38, 6, 19, 4, 35] for recent examples and [45] for a more comprehensive overview. A classic algorithm solves 3SUM in quadratic time [28]. Mildly subquadratic algorithms were developed in [13, 33, 27, 32, 17], with the current state-of-the-art algorithms running in time $\mathcal{O}(\frac{n^2}{\log^2 n} \text{polyloglog}(n))$ [13, 17].

A particularly intriguing aspect of 3SUM for our purposes is that it has been studied in many formulations that turn out to be subquadratically equivalent¹ for exact algorithms. Already in its original formulation in [28], 3SUM is equivalently defined both as “Given a set A , are there $a, b, c \in A$ with $a + b + c = 0$?” and as “Given sets A, B, C , are there $a \in A, b \in B, c \in C$ with $a + b = c$?”. More generally, equivalent formulations of 3SUM can be obtained by choosing (1) whether a, b, c are taken from a single set A (monochromatic setting) or separate sets A, B, C (trichromatic setting), (2) whether a solution has to satisfy $a + b = c$, $a + b + c = 0$, or $a + b + c = t$ for some given target t , (3) whether we decide existence of a solution triple a, b, c or compute for every c whether a solution triple a, b, c exists and (4) whether the solution additionally needs to satisfy a convolution requirement, i.e., we view each input set as a sequence, and the indices i, j, k of a_i, b_j, c_k in their respective sequences have to satisfy $i + j = k$. Variations (1) and (2) are from the original formulation in [28], variation (3) is known as 3SUM⁺ and has been shown equivalent in [45], and variation (4) is known as Convolution 3SUM, which has been shown to be equivalent by Pătraşcu [43] and has led to many subsequent applications for proving 3SUM-based lower bounds for non-geometric problems. See [18, 26] for further results on equivalent formulations of 3SUM.

All of the above formulations capture the same difficulty for designing strongly subquadratic algorithms, so they can be viewed as essentially the same problem in the exact setting. However, approximate relaxations of these formulations might – in principle – differ substantially. Furthermore, even for a fixed formulation, different notions of approximation are conceivable. Let us illustrate the range of possibilities using specific examples:

P1. (Given $A \subseteq \{-W, \dots, W\}$, are there $a, b, c \in A$ such that $a + b + c = 0$?)

A canonical attempt to relax the problem would be to approximate the smallest deviation from the target value, i.e., $\min_{a \in A, b \in B, c \in C} |a + b + c|$. This formulation has been considered in Baran et al. [13]. However, as any $(1 + \varepsilon)$ -approximation can be used to distinguish

¹ Many variants of 3SUM are even equivalent under linear-time reductions.

whether the smallest deviation from the target is 0, this formulation is 3SUM-hard and we cannot expect strongly subquadratic approximation algorithms. (Indeed, Baran et al. give mildly subquadratic algorithms for this formulation.)

For our purposes, such an approximation version does not yield a reasonable formulation, as it cannot circumvent 3SUM hardness. Instead, we will always consider formulations with positive input integers.

- P2.** (Given $A, B, C \subseteq \{1, \dots, W\}$ and t , are there a, b, c with $a + b + c = t$?)

A natural approximation version would thus be to approximate how close we can get to a (positive) target t , i.e., we approximately minimize $\max\{\frac{t}{a+b+c}, \frac{a+b+c}{t}\}$ over all $a \in A, b \in B, c \in C$. Instead of allowing both overshooting and undershooting the target, we could also restrict our notion of feasible solutions to consider only one-sided deviations (i.e., we minimize over all $a \in A, b \in B, c \in C$ such that either $a + b + c \geq t$ or $a + b + c \leq t$). Two-sided and one-sided deviations are sometimes referred to as *weak* and *strong* approximations, respectively. To the best of our knowledge, these formulations have not been studied before.

- P3.** (Given $A, B, C \subseteq \{1, \dots, W\}$, are there a, b, c with $a + b = c$?)

This formulation leads to the only efficiently solvable approximation variant of 3SUM that, to the best of our knowledge, has been studied before [31, 42]: The task is to approximately decide this formulation in the sense that (1) if there exists a solution a, b, c with $a + b = c$, the algorithm should accept, (2) if there exists no approximate solution with $a + b \in [c/(1 + \varepsilon), c(1 + \varepsilon)]$, reject, and (3) in all other cases, both acceptance or rejection are valid. Mucha et al. [42] designed an $\tilde{O}(n + 1/\varepsilon)$ -time² algorithm, complemented by a tight $(1/\varepsilon)^{1-o(1)}$ lower bound under the Strong 3SUM Conjecture. This shows that some approximate relaxations are indeed solvable faster than the exact 3SUM problem.

In summary, many natural approximate 3SUM formulations arise by choosing an arbitrary combination of an exact problem formulation and the desired approximation guarantee (one-sided vs. two-sided, approximating an optimal value vs. approximate decision), see below for details. In this work, we seek to determine the fine-grained complexity of the approximate 3SUM formulations that we consider, and to determine the effect of the precise formulation. Specifically, for polynomially bounded weights $W = n^{O(1)}$, we uncover that our considered formulations can be characterized to fall into one of three regimes: (1) *approximable in almost-linear time*: we obtain a $(1 + \varepsilon)$ -approximation algorithm running in time $\tilde{O}(n + 1/\varepsilon)$ and a matching conditional lower bound, (2) *approximable in almost-quadratic time*: we obtain a $(1 + \varepsilon)$ -approximation algorithm running in time $\tilde{O}(n/\varepsilon)$ or $\tilde{O}(n + 1/\varepsilon^2)$ and a matching conditional lower bound, and (3) *non-approximable*: we obtain a conditional lower bound of $n^{2-o(1)}$ even for *any* approximation factor.

1.1 Our Framework of Approximate 3SUM problems

With the discussion above, we arrive at the following framework of approximate 3SUM problems (Apx3SUM). To obtain a problem in this framework, one must choose:

1. Chromaticity: In the *monochromatic* case, we are given a single set $A \subseteq \{1, \dots, W\}$ and consider solutions $a, b, c \in A$. In the *trichromatic* variant, we are given sets $A, B, C \subseteq \{1, \dots, W\}$ and consider solutions $a \in A, b \in B, c \in C$.

² Unless noted otherwise, $\tilde{O}(\cdot)$ hides polylogarithmic factors in n, W , and $1/\varepsilon$.

2. Convolution: In the *convolution* case, the input set(s) A and, if applicable, B, C are ordered, i.e., sequence(s). A solution a_i, b_j, c_k is feasible only if the indices i, j, k satisfy $i + j = k$. In the standard (non-convolutional) case, we have no such restriction.
3. Target: In the *implicit target* case, an exact solution is of the form $a + b = c$. In the *explicit target* case, a target t is given as part of the input and an exact solution is of the form $a + b + c = t$.
4. Allowed deviation: In the *under-approximation* case, any feasible solution must satisfy $a + b \leq c$ or $a + b + c \leq t$, respectively. Likewise, in the *over-approximation* case, any feasible solution must satisfy $a + b \geq c$ or $a + b + c \geq t$, respectively. In the *two-sided* case, no such restriction is made.
5. Approximation guarantee: We call a, b, c an approximate solution if it deviates by at most a factor $1 + \varepsilon$ from the target (respecting the allowed deviation).³ In the *approximate decision* case, the approximation algorithm (1) must output YES if there is an exact solution, (2) must output NO if there is no approximate solution, and (3) may output either result if there is no exact solution, but an approximate one. In the *approximate optimization* case, we instead view the problem as an optimization problem, in which the objective function is $\max\{\frac{t}{a+b+c}, \frac{a+b+c}{t}\}$ or $\max\{\frac{c}{a+b}, \frac{a+b}{c}\}$, respectively. The task is then to output a $(1 + \varepsilon)$ -approximation of the optimal (i.e., smallest) value of the objective function over all feasible solutions.
6. Verbosity of output: Above, we described the standard case with a *single output*. In the *multiple outputs* case, the task is to give the answer as described above for each individual $c \in C$, i.e., for each $c \in C$ we determine whether there exist a and b such that a, b, c satisfy the approximation guarantee.

In this framework, problem P3 can be expressed as Trichromatic Single-Output Implicit-Target Two-Sided Apx3SUM Decision. Problem P2 is a trichromatic, single-output, explicit-target version of Apx3SUM Optimization. Below, we shall establish that its fine-grained complexity depends on whether we consider its two-sided or one-sided version.

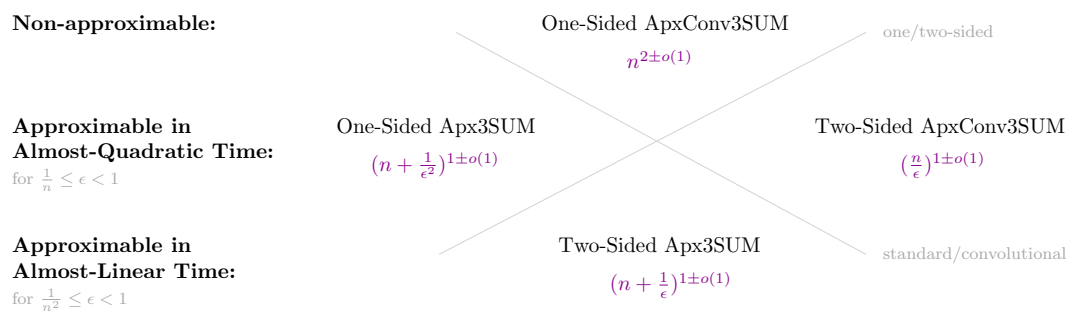
1.2 Our Results and Techniques

As first step, we substantially simplify the range of relevant variants to consider, by observing general reduction principles between these variants. Specifically, we give reductions indicating equivalence (over polynomially bounded integers) of (1) the monochromatic and trichromatic case and (2) the implicit-target and explicit-target case. Put differently, the chromaticity and target variant appear to have no influence on the fine-grained complexity, and we may *fix from now on the chromaticity to trichromatic and the target to be implicit*.

Additionally, we argue that under- and over-approximations are fine-grained equivalent, so that from now on, we distinguish only between *one-sided* (here, we use under-approximations as default) and two-sided approximations.

Curiously, the picture then simplifies further to only 4 variants to consider, see Figure 1 for an overview. The deciding axes are whether we consider one-sided or two-sided approximations and 3SUM or Convolution 3SUM – for each such choice, we obtain a different time complexity. Specifically, for each such choice, we obtain an algorithm for the most general setting along

³ E.g., for implicit-target under-approximation an approximate solution must satisfy $a + b \in [c/(1 + \varepsilon), c]$, and for explicit-target two-sided approximation it must satisfy $a + b + c \in [t/(1 + \varepsilon), t(1 + \varepsilon)]$. Also note that the latter condition is equivalent to $a + b + c \in [t(1 - \varepsilon), t(1 + \varepsilon)]$ up to changing ε by a constant factor, since $1/(1 + \varepsilon) = 1 - \Theta(\varepsilon)$. We will therefore freely choose $1/(1 + \varepsilon)$ or $1 - \varepsilon$ in the approximation lower bound, according to what gives the simpler proof.



■ **Figure 1** Overview of our results.

the remaining axes (i.e., the multi-output, approximate-optimization case), and a matching conditional lower bound for the simplest setting along the remaining axes (i.e., the single-output, approximate-decision case). The precise results are detailed below.

In a nutshell, we observe that the change from 3SUM to Conv3SUM and from two-sided to one-sided approximation increases the fine-grained complexity, while other choices turn out to be inconsequential. Furthermore, a general theme emerges for all our algorithms: we first compute small sets $S_c, c \in C$ containing candidate solutions $a + b$ for c , together with a witness (a, b) such that $s = a + b$ for each $s \in S_c$.⁴ For each $c \in C$, we then perform a binary search in S_c to find the best candidate witness for c . Here, computing the sets S_c is the crucial step, which necessarily has to be handled differently for each of the main Apx3SUM variants as evidenced by our matching conditional lower bounds.

We now detail our three regimes, as well as the specific results and techniques for each individual problem. (Note that in our time bounds throughout the paper, $\tilde{O}(\cdot)$ hides polylogarithmic factors in n , W , and $1/\epsilon$.)

Approximable in Almost-Linear Time

In this regime, we find the two-sided variants of Apx3SUM.

► **Theorem 1.1** (Two-Sided Apx3SUM). *Multi-output Two-Sided Apx3SUM Optimization can be solved in time $\tilde{O}(n + 1/\epsilon)$. Assuming the Strong 3SUM Conjecture, there is no $\tilde{O}(n + (1/\epsilon)^{1-\delta})$ -time algorithm for Single-Output Two-Sided Apx3SUM Decision with $\delta > 0$; this even holds restricted to $\epsilon = \Theta(1/n^\alpha)$ for any $\alpha \in (1, 2]$.*

The Strong 3SUM Conjecture postulates that 3SUM requires time $n^{2-o(1)}$ even for a universe size of $W = \Theta(n^2)$. It was originally formulated by Amir et al. [10], and has found uses in various contexts [34, 2, 42, 3].⁵

Our algorithm generalizes the $\tilde{O}(n + 1/\epsilon)$ -time algorithm of Mucha et al. [42] (which improves over [31]) from the single-output decision to the multi-output optimization setting. After scaling and rounding to an $\tilde{O}(1/\epsilon)$ -sized universe (similar to Mucha et al.), we apply a witness-reporting FFT to obtain an approximation set S for $A + B$; for each $c \in C$, we then perform a binary search in S for the best witness. The matching lower bound follows from a slight adaptation of Mucha et al.'s corresponding lower bound [42]. We present the details in the full version.

⁴ For the non-convolutional variants, the sets S_c are all equal to a set that approximates the sumset $A + B = \{a + b \mid a \in A, b \in B\}$.

⁵ Amir et al. stated the Strong 3SUM Conjecture in a different formulation, that we prove to be equivalent in this paper, see Section 3.

Non-Approximable

Our perhaps most surprising result is that there indeed exist non-approximable variants in our framework, specifically the one-sided variants of ApxConv3SUM.

► **Theorem 1.2 (One-Sided ApxConv3SUM).** *Assuming the Min-Plus-Convolution Conjecture, there is no $\mathcal{O}(n^{2-\delta})$ -time algorithm for Single-Output One-Sided ApxConv3SUM Decision with $\delta > 0$. This holds even for any approximation factor.*

The Min-Plus Convolution conjecture postulates that there is no $\mathcal{O}(n^{2-\delta})$ -time algorithm for computing the $(\min, +)$ -convolution of two vectors of length n . As an intriguing hardness barrier, it has become increasingly more popular over recent years, see, e.g., [14, 40, 11, 39, 23, 42, 15, 37].

To prove Theorem 1.2, we reduce via an intermediate problem known as Polyhedral 3SUM that was posed by Jeff Erickson, see [25, 14, 11]: Given sequences $A[0], \dots, A[n-1]$, $B[0], \dots, B[n-1]$ and $C[0], \dots, C[n-1]$ with the *one-sided promise* that $A[i] + B[j] \geq C[i+j]$ for all i, j , are there i, j satisfying $A[i] + B[j] = C[i+j]$?⁶ One can reduce Polyhedral 3SUM to $(\min, +)$ -convolution [14], which was interpreted as support for the hardness of $(\min, +)$ -convolution in [11]. Over polynomially bounded integers, we prove a converse reduction, which establishes Polyhedral 3SUM and $(\min, +)$ -convolution as subquadratically equivalent. This result may be of independent interest.

Given the hardness of Polyhedral 3SUM, non-approximability of One-Sided ApxConv3SUM (specifically, allowing only *under-approximations*) is essentially immediate: Think of a Polyhedral 3SUM instance A, B, C as an instance for One-Sided ApxConv3SUM Decision. If there exist i, j such that $A[i] + B[j] = C[i+j]$, an approximate decider must accept. Otherwise, for all i, j we have $A[i] + B[j] > C[i+j]$ by the one-sided promise of Polyhedral 3SUM. However, any feasible solution for One-Sided ApxConv3SUM is of the form $A[i] + B[j] \leq C[i+j]$, so no feasible solution exists at all! Thus, any algorithm for One-Sided ApxConv3SUM (for *any* approximation ratio) is able to decide existence of an exact solution of Polyhedral 3SUM.

Let us also discuss our reduction from Min-Plus Convolution to Polyhedral 3SUM. For ease of presentation, we now use the equivalent formulation of Polyhedral 3SUM with one-sided promise $A[i] + B[j] \leq C[i+j]$ (instead of $A[i] + B[j] \geq C[i+j]$). As established in [23], it suffices to reduce from the problem of deciding, given sequences A, B, C , whether the one-sided promise holds, i.e., whether $A[i] + B[j] \leq C[i+j]$ for all i, j (Cygan et al. [23] call this problem MAXCONVUB). We carefully use a trick introduced in [46] to reduce an inequality test to a logarithmic number of equality tests: for any B -bit numbers a, b, c , we have $a + b > c$ iff there exist $q \in \{0, \dots, B\}$ and $\Delta \in \{1, 2, 3\}$ with $\lfloor \frac{a}{2^q} \rfloor + \lfloor \frac{b}{2^q} \rfloor = \lfloor \frac{c}{2^q} \rfloor + \Delta$. Thus, to search for a *violation* i, j satisfying $A[i] + B[j] > C[i+j]$, we solve an appropriately defined Polyhedral 3SUM instance for each q and Δ . By performing these tests in the correct order, we can show that the one-sided promise always holds *until* we find the first violation (if there exists any). The details are deferred to the full version.

Approximable in Almost-Quadratic Time

There are two remaining problems, which turn out to be almost-quadratic time approximable. Let us describe Two-Sided ApxConv3SUM first:

⁶ The name stems from viewing the input as a point in the $3n$ -dimensional polyhedron given by $A[i] + B[j] \geq C[i+j]$; the task then becomes to decide whether the point is on the boundary.

► **Theorem 1.3** (Two-Sided ApxConv3SUM). *Multi-output Two-Sided ApxConv3SUM Optimization can be solved in time $\tilde{O}(n/\varepsilon)$. Assuming the Strong 3SUM Conjecture, there is no $\tilde{O}((n/\varepsilon)^{1-\delta})$ -time algorithm for Single-Output Two-Sided ApxConv3SUM Decision with $\delta > 0$; this even holds restricted to $\varepsilon = \Theta(1/n^\alpha)$ for any $\alpha \in (0, 1]$.*

To obtain this algorithm, we compute sets $S[1], \dots, S[n]$ of size $\tilde{O}(1/\varepsilon)$ such that for each $1 \leq i < k \leq n$ the set $S[k]$ contains a $(1 \pm \varepsilon)$ -approximation of $A[i] + B[k - i]$. For each k , the construction of $S[k]$ uses similar ideas as exploited for Two-Sided Apx3SUM. While this beats exact algorithms and yields an almost-quadratic time approximation in time $\tilde{O}(n/\varepsilon)$, one may wonder whether a faster approximation algorithm may be obtainable by somehow avoiding the computation of an individual $\tilde{O}(1/\varepsilon)$ -sized set for each $k \in [n]$.

However, time $(n/\varepsilon)^{1-o(1)}$ is indeed required, assuming the Strong 3SUM conjecture. To obtain this result, we exploit a close connection between Two-Sided ApxConv3SUM and Conv3SUM over small universe size W . A natural algorithm solves Conv3SUM over $\{1, \dots, W\}$ in time $\tilde{O}(n \cdot \min\{n, W\})$, see Section 2. We show that this bound is conditionally tight for all $W = \Theta(n^\alpha)$, $\alpha \in (0, 1]$ assuming the Strong 3SUM conjecture. Interestingly, this proves equivalence of the Strong 3SUM conjecture and the following *Strong Conv3SUM conjecture*: Conv3SUM requires time $n^{2-o(1)}$ even for a universe of size $W = \Theta(n)$. In fact, the Strong 3SUM conjecture has originally been phrased as this Strong Conv3SUM conjecture (most likely to postulate the strongest plausible hardness assumption for a 3SUM formulation), but we are not aware of any proof of equivalence prior to our work.

For the reduction from 3SUM over universe $[n^2]$ to Conv3SUM over universe $W = \Theta(n^\alpha)$ with $\alpha \in (0, 1]$, the high-level idea is as follows: We set $\beta = \frac{2\alpha}{\alpha+1}$, and choose m suitably with $m = \Theta(n^{2-\beta})$ to write any number a as $(\lfloor \frac{a}{m} \rfloor, a \bmod m)$. Intuitively, we would like to create a convolution instance $A', B', C' \in \{0, \dots, W'\}^{n'}$ where $n' = m$ and $W' = n^2/m$ by setting $A'[a \bmod m] = \lfloor \frac{a}{m} \rfloor$ for each $a \in A$ (similarly for $b \in B$ and $c \in C$) which would allow us to detect most possible witnesses. By choosing m appropriately using ideas of [18], we can ensure that there are not too many collisions (i.e., $a, a' \in A$ with $a \bmod m = a' \bmod m$), which can be handled separately. In total, we show how to obtain few such instances to decide the given 3SUM instance. By our choice of $m = \Theta(n^\beta)$, we obtain $W' = \Theta((n')^\alpha)$, and an $\mathcal{O}((n'W')^{1-\delta})$ -time Conv3SUM algorithm would give an $\mathcal{O}(n^{2-\delta'})$ -time 3SUM algorithm.

Once the $(nW)^{1-o(1)}$ hardness of Conv3SUM is proven, we can obtain our desired hardness for Two-Sided ApxConv3SUM by setting $\varepsilon = \Theta(1/W)$. For details see Sections 3 and 4.

► **Theorem 1.4** (One-Sided Apx3SUM). *Multi-output One-Sided Apx3SUM Optimization can be solved in time $\tilde{O}(n + 1/\varepsilon^2)$. Assuming the Min-Plus Convolution Conjecture, there is no $\tilde{O}(n + (1/\varepsilon)^{2-\delta})$ -time algorithm for Single-Output One-Sided Apx3SUM Decision with $\delta > 0$; this even holds restricted to $\varepsilon = \Theta(1/n^\alpha)$ for any $\alpha \in (1/2, 1]$.*

To obtain our algorithm, we adapt the notion of Δ -approximation for sumsets by Bringmann and Nakos [15]. A set S Δ -approximates a sumset $A + B = \{a + b \mid a \in A, b \in B\}$ if for all $a \in A, b \in B$, there exist $s^-, s^+ \in S$ such that $s^- \leq a + b \leq s^+ \leq s^- + \Delta$. Bringmann and Nakos showed how to compute such a set in time $\tilde{O}(n + (W/\Delta)^2)$ given any $A, B \subseteq \{1, \dots, W\}$. We adapt the techniques in [15] to provide a version with a multiplicative error of $1 + \varepsilon$ in time $\tilde{O}(n + 1/\varepsilon^2)$, together with a witness a, b with $a + b = s$ for each $s \in S$.

For the lower bound, we can again use our established equivalence of Polyhedral 3SUM and Min-Plus Convolution – this time, however, we rule out a subquadratic approximation $\mathcal{O}((1/\varepsilon)^{2-\delta})$ rather than establishing inapproximability. Specifically, given an instance A', B', C' of Polyhedral 3SUM satisfying $A'[i] + B'[j] \geq C'[i + j]$ for all i, j , we reduce to an instance A, B, C of Single-Output One-Sided Apx3SUM Decision constructed as follows:

A, B, C consist of the elements $10Wi + A'[i]$, $10Wi + B'[i]$ and $10Wi + C'[i]$, respectively, where $i \in \{0, \dots, n-1\}$. If the Polyhedral 3SUM instance is a YES instance, clearly an exact solution exists, so the approximate decider will return YES. If the Polyhedral 3SUM instance is NO, then any approximate solution $(1-\varepsilon)c \leq a + b \leq c$ with $\varepsilon = 1/(10n)$ must be of the form $a = 10Wi + A'[i]$, $b = 10Wj + B'[j]$, $c = 10Wk + C'[k]$ with $i + j = k$; this is because $i + j > k$ yields $a + b > c$, while $i + j < k$ yields $a + b < (1-\varepsilon)c$. However, by $A'[i] + B'[j] \geq C'[i+j]$, such a solution can only satisfy the hard constraint $a + b \leq c$ if $A'[i] + B'[j] = C'[i+j]$, which does not hold for any i, j . Thus, no approximate solution exists, and the approximate decider must return NO.

The above argument yields a $(1/\varepsilon)^{2-o(1)}$ lower bound for $\varepsilon = \Theta(1/n)$, which we generalize to hold for all $\varepsilon = \Theta(1/n^\alpha)$ with $\alpha \in (1/2, 1]$. In fact, our results establish a subquadratic equivalence of One-Sided Apx3SUM with Min-Plus Convolution.

1.3 Further Related Work

Our work is related to a recent effort of the fine-grained complexity community to obtain best-possible approximation schemes for classic optimization problems such as Subset Sum, Knapsack, and Partition. It is now known that Subset Sum admits approximation schemes with running time $\tilde{O}(\min\{n/\varepsilon, n + 1/\varepsilon^2\})$ [30, 29, 36] and has matching conditional lower bounds ruling out time $\mathcal{O}((n + 1/\varepsilon)^{1.999})$ [15] and time $2^{o(n)}/\varepsilon^{0.999}$ [5]. Knapsack has an approximation scheme with running time $\tilde{O}(n + 1/\varepsilon^2)$ [41, 20] and a matching conditional lower bound ruling out time $\mathcal{O}((n + 1/\varepsilon)^{1.999})$ [39, 23]. Partition has an approximation scheme with running time $\tilde{O}(n + 1/\varepsilon)$ [21] and a matching conditional lower bound ruling out time $\mathcal{O}(n + 1/\varepsilon^{0.999})$ [5]. In this paper, we explore this approach on the 3SUM problem; in particular, we use a tool for approximating sumsets that was developed for a Subset Sum approximation algorithm [15].

2 Preliminaries: 3SUM

We write $[n]$ for the set $\{1, 2, \dots, n\}$, $[s, e]$ for $\{s, \dots, e\}$, and $[s, e)$ for $\{s, \dots, e-1\}$. By \tilde{O} -notation we hide $\text{polylog}(nW/\varepsilon)$ factors, that is, we write $\tilde{O}(T) = \bigcup_{c \geq 0} \mathcal{O}(T \log^c(nW/\varepsilon))$. The exact versions of 3SUM and Conv3SUM are defined as follows.

► **Problem 2.1** (3SUM). *Given sets $A, B, C \subseteq [W]$ of size n , decide whether there exist $a \in A, b \in B, c \in C$ with $a + b = c$.*

► **Problem 2.2** (Conv3SUM). *Given sequences $A, B, C \in [W]^n$, decide whether there exist $i, j \in [n]$ with $A[i] + B[j] = C[i+j]$.*

Both problems are well known to be solvable in time $\mathcal{O}(n^2)$. This running time can be improved by logarithmic factors, see, e.g., [17]. For exact algorithms, 3SUM and Conv3SUM are subquadratically equivalent [43, 18]. Lack of further progress led to the following conjecture:

► **Conjecture 2.3** (3SUM Conjecture [28]). *3SUM has no algorithm with running time $\tilde{O}(n^{2-\delta})$ for any constant $\delta > 0$.*

This is equivalent to the analogous statement for Conv3SUM, because the problems are subquadratically equivalent [43, 18].

The Fast Fourier Transform (FFT) can be used to solve 3SUM in time $\mathcal{O}(n + W \log W)$, see, e.g., [22, Exercise 30.1-7]. For $W = n^2$, both algorithmic approaches run in time $\tilde{O}(n^2)$. The Strong 3SUM Conjecture formalizes this lack of progress in the case $W = n^2$.

► **Conjecture 2.4** (Strong 3SUM Conjecture [10]). *3SUM with $W = n^2$ has no algorithm with running time $\tilde{O}(n^{2-\delta})$ for any constant $\delta > 0$.*

Conv3SUM can be solved in time $\tilde{O}(nW)$, by reducing a given Conv3SUM instance A, B, C to the 3SUM instance in which $A[i]$ is replaced by $10Wi + A[i]$, and similarly for B and C , and then using FFT to solve the obtained 3SUM instance. For $W = n$, both algorithmic approaches solve Conv3SUM in time $\tilde{O}(n^2)$. The Strong Conv3SUM Conjecture formalizes this lack of progress.

► **Conjecture 2.5** (Strong Conv3SUM Conjecture [10]). *Conv3SUM with $W = n$ has no algorithm with running time $\tilde{O}(n^{2-\delta})$ for any constant $\delta > 0$.*

To be precise, Amir et al. [10] actually introduced the Strong Conv3SUM Conjecture (under the name “Strong 3SUM-Hardness Assumption”), but others have used the Strong 3SUM Conjecture, see, e.g., [34, 42]. Both conjectures have been used interchangeably in the literature, depending on the specific needs of reductions. However, it was not known whether the two conjectures are equivalent. In this paper, we prove equivalence of the Strong 3SUM Conjecture and the Strong Conv3SUM Conjecture, see Corollary 3.3 in Section 3. This justifies in hindsight why both conjectures have been used interchangeably.

3 Tool: Equivalence of Strong 3SUM and Strong Conv3SUM Conjectures

In this section, we present a reduction from 3SUM over a quadratic-size universe to Conv3SUM over a sublinear-size universe. We will later use this as a tool to generalize our lower bound for Two-Sided ApxConv3SUM to any relation between ε and n , see Theorem 4.4. As a consequence of this reduction we also obtain the equivalence of the Strong 3SUM Conjecture and the Strong Conv3SUM Conjecture, see Corollary 3.3.

► **Theorem 3.1.** *Conv3SUM with universe size $W = \lfloor n^\alpha \rfloor$ cannot be solved in time $\tilde{O}((nW)^{1-\delta})$, for any constants $\alpha \in (0, 1]$, $\delta > 0$, assuming the Strong 3SUM Conjecture.*

Proof. We prove this result by adapting the reduction from 3SUM to Conv3SUM by Chan and He [18]. Assume for the sake of contradiction that Conv3SUM with input size \tilde{n} and universe size $\tilde{W} = \lfloor \tilde{n}^\alpha \rfloor$ can be solved in time $\tilde{O}((\tilde{n}\tilde{W})^{1-\delta})$ for some $\delta > 0$; we show that then 3SUM with input size n and universe size n^2 can be solved in time $\tilde{O}(n^{2-\delta'})$ for some $\delta' > 0$, contradicting the Strong 3SUM Conjecture.

Suppose we are given a 3SUM instance with input sets $A, B, C \subseteq [n^2]$ of size n . Define $\beta = 2\alpha/(\alpha + 1) \in [0, 1]$ and let $m = \Theta(n^{2-\beta})$ be an integer parameter that we specify later. For notational convenience, we define $q_a := \lfloor \frac{a}{m} \rfloor$ and $r_a := a \bmod m$, denoting the quotient and remainder of an integer a divided by m . Construct the sequence of buckets $K_A[r] := \{q_a \mid r_a = r, a \in A\}$, and construct K_B and K_C analogously.⁷

Let $K_X[r][i]$ denote the i -th element of the set $K_X[r]$ for some arbitrary ordering. Define $\tilde{W} := \frac{n^2}{m} = \Theta(n^\beta)$ and let $t \in [n]$ be some threshold parameter to be specified later. For each $i \in [t]$, define the following sequences of length $2m$:

⁷ The main difference to the reduction in [18] is that inside the buckets we store the quotient q_a instead of the original number a , which results in sequences with smaller values.

$$\begin{aligned}\tilde{A}_i[r] &:= \begin{cases} K_A[r][i], & \text{if } r \in [0, m) \text{ and } i \leq |K_A[r]| \leq t \\ 20\tilde{W}, & \text{otherwise} \end{cases} \\ \tilde{B}_i[r] &:= \begin{cases} K_B[r][i], & \text{if } r \in [0, m) \text{ and } i \leq |K_B[r]| \leq t \\ 20\tilde{W}, & \text{otherwise} \end{cases} \\ \tilde{C}_i[r] &:= \begin{cases} K_C[r][i], & \text{if } r \in [0, m) \text{ and } i \leq |K_C[r]| \leq t \\ K_C[r-m][i] - 1, & \text{if } r \in [m, 2m) \text{ and } i \leq |K_C[r]| \leq t \\ 20\tilde{W}, & \text{otherwise} \end{cases}\end{aligned}$$

For all triples $(i, j, k) \in [t]^3$, solve Conv3SUM on $(\tilde{A}_i, \tilde{B}_j, \tilde{C}_k)$, and return YES if we receive YES for any triple. Otherwise, let R_A be the set containing all elements $a \in A$ in *overflow* buckets, i.e., buckets $K_A[r]$ with $|K_A[r]| > t$, and define R_B and R_C analogously. Compute the sumsets $R_A + B$ and $A + R_B$ and the difference set $R_C - A$ naively in time $O(n \cdot (|R_A| + |R_B| + |R_C|))$. If either $C \cap (R_A + B) \neq \emptyset$, or $C \cap (A + R_B) \neq \emptyset$, or $B \cap (R_C - A) \neq \emptyset$, then return YES; otherwise, return NO. This finishes the algorithm description.

To show correctness, assume that for some $a \in A, b \in B, c \in C$, we have $a + b = c$. If $a \in R_A$, then $C \cap (R_A + B) \neq \emptyset$ holds. The same happens when $b \in R_B$. If $c \in R_C$, then since $c - a = b$ we have $B \cap (R_C - A) \neq \emptyset$. In all cases, we correctly return YES.

Otherwise, we have $a \notin R_A, b \notin R_B$, and $c \notin R_C$. Thus, there exist indices $i, j, k \in [t]$ such that $q_a = K_A[r_a][i] = \tilde{A}_i[r_a]$, $q_b = K_B[r_b][j] = \tilde{B}_j[r_b]$, and $q_c = K_C[r_c][k] = \tilde{C}_k[r_c] = \tilde{C}_k[m + r_c] + 1$, which are captured in the instance $(\tilde{A}_i, \tilde{B}_j, \tilde{C}_k)$.

The equation $a + b = c$ holds if and only we have

$$(q_a + q_b = q_c \quad \text{and} \quad r_a + r_b = r_c) \quad \text{or} \quad (1)$$

$$(q_a + q_b + 1 = q_c \quad \text{and} \quad r_a + r_b = m + r_c). \quad (2)$$

In Case (1), we get

$$\tilde{A}_i[r_a] + \tilde{B}_j[r_b] = q_a + q_b = q_c = \tilde{C}_k[r_c],$$

which is a Conv3SUM solution, and we output YES. In Case (2), we instead get

$$\tilde{A}_i[r_a] + \tilde{B}_j[r_b] = q_a + q_b = q_c - 1 = \tilde{C}_k[m + r_c],$$

again, correctly a Conv3SUM solution.

In the other direction, suppose we output YES due to a solution in some Conv3SUM instance i, j, k , i.e., we have $\tilde{A}_i[x] + \tilde{B}_j[y] = \tilde{C}_k[z]$ with $x + y = z$. It is easy to see that entries set to $20\tilde{W}$ cannot contribute to a solution. Therefore, the condition $x + y = z$ translates to either $r_a + r_b = r_c$ or $r_a + r_b = m + r_c$ for some $a \in A, b \in B, c \in C$. Similarly, the condition $\tilde{A}_i[x] + \tilde{B}_j[y] = \tilde{C}_k[z]$ translates to $q_a + q_b = q_c$ or $q_a + q_b = q_c - 1$, respectively. As before, this implies $a + b = c$. Correctness is immediate in the case where we output YES due to elements in $R_A \subseteq A, R_B \subseteq B$, or $R_C \subseteq C$.

To complete our randomized reduction, we pick m as a uniformly random prime in $[n^{2-\beta}/2, n^{2-\beta}]$. Let $|R| := \max\{|R_A|, |R_B|, |R_C|\}$. By following [18, Lemmas 2.3 and 2.4], we show $\mathbb{E}[|R|] = \tilde{O}(n^{\frac{\beta}{t}})$; we prove this formally in the full version. Observe that all constructed Conv3SUM instances have length $\tilde{n} := 2m = \Theta(n^{2-\beta})$ and entries in $\{0, \dots, \tilde{W}\}$ with $\tilde{W} = \Theta(n^\beta)$. By choice of $\beta = 2\alpha/(\alpha + 1)$, it indeed holds that $\tilde{W} = \Theta(\tilde{n}^\alpha)$.

Hence, if there is a Conv3SUM algorithm with running time $\mathcal{O}((\tilde{n}\tilde{W})^{1-\delta})$ for some $\delta > 0$, we obtain a 3SUM algorithm running in expected time $\tilde{\mathcal{O}}(t^3 \cdot (\tilde{n}\tilde{W})^{1-\delta} + n \cdot |R|) = \tilde{\mathcal{O}}(t^3 \cdot (n^2)^{1-\delta} + n \cdot |R|)$. Choosing $t = 4n^{\delta/2}$ results in expected time $\tilde{\mathcal{O}}(n^{2-\delta/2} + n^{1+\beta-\delta/2}) = \tilde{\mathcal{O}}(n^{2-\delta/2})$. The derandomization of this reduction can be done by following the ideas from Chan and He [18] and is deferred to the full version. \blacktriangleleft

The reverse direction follows from a folklore reduction from Conv3SUM to 3SUM, enriched by a simple padding, yielding the following lemma.

► **Lemma 3.2** (Proof deferred to the full version). *If the Strong 3SUM Conjecture is false, then Conv3SUM with $W = n$ can be solved in time $\mathcal{O}(n^{2-\delta})$ for some $\delta > 0$.*

As such, we obtain the following equivalence.

► **Corollary 3.3.** *The Strong 3SUM Conjecture is equivalent to the Strong Conv3SUM Conjecture.*

Proof. One direction is given by Theorem 3.1 for $\alpha := 1$, the other by Lemma 3.2. \blacktriangleleft

4 Two-Sided ApxConv3SUM

Our algorithm uses the Fast Fourier Transform (FFT) to compute a multiplicative approximation of a sumset respecting convolution, a *convolved sumset*, see Lemma 4.2 below. This approximation then directly implies that Two-Sided ApxConv3SUM has time complexity $\tilde{\mathcal{O}}(n/\varepsilon)$. The lower bound is a reduction from Conv3SUM and uses our tool, Theorem 3.1, which connects the Strong 3SUM Conjecture to Conv3SUM.

Our algorithm makes use of the following tool that follows from FFT. The proof is deferred to the full version.

► **Fact 4.1** (Sumset Computation with Witnesses). *Given sets $A, B \subseteq \{0, 1, \dots, W\}$ their sumset $A + B = \{a + b \mid a \in A, b \in B\}$ can be computed in time $\mathcal{O}(W \log W)$. Moreover, in time $\tilde{\mathcal{O}}(W)$ we can compute for each $s \in A + B$ a witness $(a_s, b_s) \in A \times B$ with $a_s + b_s = s$.*

4.1 Approximation Algorithm

We begin with the computation of a convolved sumset in time $\tilde{\mathcal{O}}(n/\varepsilon)$.

► **Lemma 4.2** (Two-Sided Multiplicative Approximation of Convolved Sumset). *Given sequences $A, B \in [W]^n$ and $\varepsilon \in (0, 1]$, in time $\tilde{\mathcal{O}}(n/\varepsilon)$ we can compute sets $S[1], \dots, S[n]$, each of size $\tilde{\mathcal{O}}(1/\varepsilon)$, such that (1) for every $k \in [n]$ and $s \in S[k]$ there exists $i_{k,s} \in [k-1]$ with $s = A[i_{k,s}] + B[k - i_{k,s}]$ and (2) for every $k \in [n], i \in [k-1]$ there exists $s \in S[k]$ with $\max\{s/(A[i] + B[k-i]), (A[i] + B[k-i])/s\} \leq 1 + \varepsilon$. Moreover, in the same running time we can compute the witnesses $i_{k,s}$ promised by (1).*

Proof. The algorithm works as follows:

1. Set $\bar{\varepsilon} := \varepsilon/4$ and $M := 100/\bar{\varepsilon}$. Initialize $S[1] = \dots = S[n] = \emptyset$.
2. For each $q \in [W]$ that is a power of 2, compute the sets

$$A_q := \{i \cdot M + \lfloor A[i]/\bar{\varepsilon}q \rfloor \mid i \in [n], A[i] \in [0, 2q)\},$$

$$B_q := \{j \cdot M + \lfloor B[j]/\bar{\varepsilon}q \rfloor \mid j \in [n], B[j] \in [0, 2q)\}.$$

Moreover, associate to each $x \in A_q$ the corresponding index $i_{q,x} \in [n]$ (so that $x = i_{q,x} \cdot M + \lfloor A[i_{q,x}]/\bar{\varepsilon}q \rfloor$), and similarly associate to each $y \in B_q$ the corresponding $j_{q,y} \in [n]$.

34:12 Exploring the Approximability Landscape of 3SUM

3. For each $q \in [W]$ that is a power of 2, use Fact 4.1 to compute the sumset $C_q := A_q + B_q$, along with witnesses $(x_z, y_z) \in A_q \times B_q$ with $x_z + y_z = z$ for each $z \in C_q$. For each $z \in C_q$, add the number $s := A[i_{q,x_z}] + B[j_{q,y_z}]$ to the set $S[k]$ for $k := i_{q,x_z} + j_{q,y_z}$, and associate the witness $i_{k,s} := i_{q,x_z}$.
4. Return the sets $S[1], \dots, S[k]$ with the associated witnesses $i_{k,s}$.

The correctness proof and running time analysis are deferred to the full version. ◀

Using the above construction, we obtain our $\tilde{O}(n/\varepsilon)$ -time approximation algorithm.

► **Theorem 4.3** (Upper Bound for Two-Sided ApxConv3SUM). *Consider the following optimization version of Two-Sided ApxConv3SUM: For sequences $A, B, C \in [W]^n$ define for each $k \in [n]$ the number $\text{OPT}_k := \min_{i \in [k-1]} \max\{C[k]/(A[i] + B[k-i]), (A[i] + B[k-i])/C[k]\}$. Given sequences $A, B, C \in [W]^n$ and $\varepsilon \in (0, 1]$, compute for each $k \in [n]$ a number $i' \in [k-1]$ with $\max\{C[k]/(A[i'] + B[k-i']), (A[i'] + B[k-i'])/C[k]\} \leq (1 + \varepsilon)\text{OPT}_k$.*

This problem can be solved in time $\tilde{O}(n/\varepsilon)$.

Proof. The algorithm works as follows:

1. Use Lemma 4.2 to compute sets $S[1], \dots, S[n]$, along with a witness $i_{k,s}$ for each $k \in [n], s \in S[k]$.
2. For each $k \in [n]$, use binary search to find the largest $s_0 \in S[k]$ with $s_0 \leq C[k]$, and use binary search to find the smallest $s_1 \in S[k]$ with $s_1 \geq C[k]$. Pick $s \in \{s_0, s_1\}$ to minimize the ratio $\max\{C[k]/s, s/C[k]\}$. Report the witness $i_{k,s}$ as the result for k .

For correctness, fix any $k \in [n]$ and pick $i \in [k-1]$ that realizes the optimal ratio $\text{OPT}_k = \max\{C[k]/(A[i] + B[k-i]), (A[i] + B[k-i])/C[k]\}$. By property (2) of Lemma 4.2, there exists $s' \in S$ with $\max\{s'/(A[i] + B[k-i]), (A[i] + B[k-i])/s'\} \leq 1 + \varepsilon$. This yields

$$\begin{aligned} \max\{C[k]/s', s'/C[k]\} &\leq \max\{s'/(A[i] + B[k-i]), (A[i] + B[k-i])/s'\} \\ &\quad \cdot \max\{C[k]/(A[i] + B[k-i]), (A[i] + B[k-i])/C[k]\} \\ &\leq (1 + \varepsilon)\text{OPT}_k. \end{aligned}$$

Observe that the number $s \in S[k]$ minimizing the ratio $\max\{C[k]/s, s/C[k]\}$ is either $s = s_0$ or $s = s_1$. Hence, by picking $s \in \{s_0, s_1\}$ to minimize the ratio $\max\{C[k]/s, s/C[k]\}$, we compute the number $s \in S$ minimizing the ratio $\max\{C[k]/s, s/C[k]\}$. In particular, s is at least as good as s' , meaning that we have $\max\{C[k]/s, s/C[k]\} \leq \max\{C[k]/s', s'/C[k]\} \leq (1 + \varepsilon)\text{OPT}_k$. Finally, observe that the reported witness $i_{k,s}$ for s satisfies $A[i_{k,s}] + B[k - i_{k,s}] = s$ and thus $\max\{C[k]/(A[i_{k,s}] + B[k - i_{k,s}]), (A[i_{k,s}] + B[k - i_{k,s}])/C[k]\} = \max\{C[k]/s, s/C[k]\} \leq (1 + \varepsilon)\text{OPT}_k$. This proves the approximation guarantee.

The running time bound $\tilde{O}(n/\varepsilon)$ is clear from Lemma 4.2. ◀

4.2 Hardness of Approximation

We obtain a matching hardness result under the Strong 3SUM Conjecture.

► **Theorem 4.4** (Lower Bound for Two-Sided ApxConv3SUM). *Consider the following decision version of Two-Sided ApxConv3SUM: Given sequences $A, B, C \in [W]^n$ and $\varepsilon \in (0, 1]$,*

- *output YES if $\exists i, j: A[i] + B[j] = C[i + j]$*
- *output NO if $\nexists i, j: (1 - \varepsilon)C[i + j] \leq A[i] + B[j] \leq (1 + \varepsilon)C[i + j]$*
- *otherwise, both YES and NO are admissible.*

This problem cannot be solved in time $\tilde{O}((n/\varepsilon)^{1-\delta})$ for any $\delta > 0$, assuming the Strong 3SUM Conjecture. This statement even holds restricted to $\varepsilon = \Theta(1/n^\alpha)$ for any $\alpha \in (0, 1]$.

Proof. Assuming the Strong 3SUM Conjecture, by Theorem 3.1 a Conv3SUM instance $A, B, C \in [W]^n$ can in general not be solved in time $\tilde{O}((nW)^{1-\delta})$ for $W = \lfloor n^\alpha \rfloor$. We argue that the Conv3SUM instance A, B, C is equivalent to the Two-Sided Apx3SUM instance A, B, C with $\varepsilon := 1/(10n^\alpha)$. It follows that Two-Sided ApxConv3SUM cannot be solved in time $\tilde{O}((nW)^{1-\delta}) = \tilde{O}((n/\varepsilon)^{1-\delta})$ for $\varepsilon = \Theta(1/n^\alpha)$.

To see the equivalence, observe that $(1 - \varepsilon)C[i + j] \leq A[i] + B[j] \leq (1 + \varepsilon)C[i + j]$ is equivalent to $A[i] + B[j] = C[i + j]$, since $A[i], B[j], C[i + j] \in [W]$ are integers and $\varepsilon \leq 1/(10W)$. Therefore, an algorithm for Two-Sided ApxConv3SUM must output YES if there are i, j with $A[i] + B[j] = C[i + j]$ and NO otherwise, i.e., the algorithm solves Conv3SUM. This finishes the proof. \blacktriangleleft

References

- 1 Amir Abboud and Arturs Backurs. Towards hardness of approximation for polynomial time problems. In *ITCS*, volume 67 of *LIPICs*, pages 11:1–11:26, 2017.
- 2 Amir Abboud, Arturs Backurs, Karl Bringmann, and Marvin Künnemann. Fine-grained complexity of analyzing compressed data: Quantifying improvements over decompress-and-solve. In *FOCS*, pages 192–203. IEEE Computer Society, 2017.
- 3 Amir Abboud, Arturs Backurs, Karl Bringmann, and Marvin Künnemann. Impossibility results for grammar-compressed linear algebra. In *NeurIPS*, 2020.
- 4 Amir Abboud, Karl Bringmann, and Nick Fischer. Stronger 3-SUM lower bounds for approximate distance oracles via additive combinatorics. In *STOC*, pages 391–404. ACM, 2023.
- 5 Amir Abboud, Karl Bringmann, Danny Hermelin, and Dvir Shabtay. SETH-based lower bounds for subset sum and bicriteria path. *ACM Trans. Algorithms*, 18(1):6:1–6:22, 2022.
- 6 Amir Abboud, Karl Bringmann, Seri Khoury, and Or Zamir. Hardness of approximation in P via short cycle removal: cycle detection, distance oracles, and beyond. In *STOC*, pages 1487–1500. ACM, 2022.
- 7 Amir Abboud and Kevin Lewi. Exact weight subgraphs and the k-sum conjecture. In *ICALP (1)*, volume 7965 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2013.
- 8 Amir Abboud, Aviad Rubinfeld, and R. Ryan Williams. Distributed PCP theorems for hardness of approximation in P. In *FOCS*, pages 25–36. IEEE Computer Society, 2017.
- 9 Amir Abboud, Virginia Vassilevska Williams, and Oren Weimann. Consequences of faster alignment of sequences. In *ICALP (1)*, volume 8572 of *Lecture Notes in Computer Science*, pages 39–51. Springer, 2014.
- 10 Amihod Amir, Timothy M. Chan, Moshe Lewenstein, and Noa Lewenstein. On hardness of jumbled indexing. In *ICALP (1)*, volume 8572 of *Lecture Notes in Computer Science*, pages 114–125. Springer, 2014.
- 11 Arturs Backurs, Piotr Indyk, and Ludwig Schmidt. Better approximations for tree sparsity in nearly-linear time. In *SODA*, pages 2215–2229. SIAM, 2017.
- 12 Arturs Backurs, Liam Roditty, Gilad Segal, Virginia Vassilevska Williams, and Nicole Wein. Towards tight approximation bounds for graph diameter and eccentricities. In *STOC*, pages 267–280. ACM, 2018.
- 13 Ilya Baran, Erik D. Demaine, and Mihai Pătraşcu. Subquadratic algorithms for 3SUM. *Algorithmica*, 50(4):584–596, 2008.
- 14 David Bremner, Timothy M. Chan, Erik D. Demaine, Jeff Erickson, Ferran Hurtado, John Iacono, Stefan Langerman, Mihai Pătraşcu, and Perouz Taslakian. Necklaces, convolutions, and X+Y. *Algorithmica*, 69(2):294–314, 2014.
- 15 Karl Bringmann and Vasileios Nakos. A fine-grained perspective on approximating subset sum and partition. In *SODA*, pages 1797–1815. SIAM, 2021.

- 16 Karl Bringmann and André Nusser. Translating Hausdorff is hard: Fine-grained lower bounds for Hausdorff distance under translation. In *SoCG*, volume 189 of *LIPICs*, pages 18:1–18:17, 2021.
- 17 Timothy M. Chan. More logarithmic-factor speedups for 3SUM, (median, +)-convolution, and some geometric 3SUM-hard problems. *ACM Trans. Algorithms*, 16(1):7:1–7:23, 2020.
- 18 Timothy M. Chan and Qizheng He. Reducing 3SUM to Convolution-3SUM. In *SOSA*, pages 1–7. SIAM, 2020.
- 19 Timothy M. Chan, Virginia Vassilevska Williams, and Yinzhan Xu. Hardness for triangle problems under even more believable hypotheses: reductions from real APSP, real 3SUM, and OV. In *STOC*, pages 1501–1514. ACM, 2022.
- 20 Lin Chen, Jiayi Lian, Yuchen Mao, and Guochuan Zhang. A nearly quadratic-time FPTAS for knapsack. *CoRR*, abs/2308.07821, 2023.
- 21 Lin Chen, Jiayi Lian, Yuchen Mao, and Guochuan Zhang. Approximating partition in near-linear time. *CoRR*, abs/2402.11426, 2024.
- 22 T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction to Algorithms, Fourth Edition*. MIT Press, 2022.
- 23 Marek Cygan, Marcin Mucha, Karol Wegrzycki, and Michal Włodarczyk. On problems equivalent to (min, +)-convolution. *ACM Trans. Algorithms*, 15(1):14:1–14:25, 2019.
- 24 Mina Dalirrooyfard, Ce Jin, Virginia Vassilevska Williams, and Nicole Wein. Approximation algorithms and hardness for n-pairs shortest paths and all-nodes shortest cycles. In *FOCS*, pages 290–300. IEEE, 2022.
- 25 Erik D. Demaine and Joseph O’Rourke. Open problems: Open problems from CCCG 2005. In *CCCG*, 2006.
- 26 Bartłomiej Dudek, Pawel Gawrychowski, and Tatiana Starikovskaya. All non-trivial variants of 3-LDT are equivalent. In *STOC*, pages 974–981. ACM, 2020.
- 27 Ari Freund. Improved subquadratic 3SUM. *Algorithmica*, 77(2):440–458, 2017.
- 28 Anka Gajentaan and Mark H. Overmars. On a class of $O(n^2)$ problems in computational geometry. *Comput. Geom.*, 5:165–185, 1995.
- 29 George Gens and Eugene Levner. Computational complexity of approximation algorithms for combinatorial problems. In *MFCS*, volume 74 of *Lecture Notes in Computer Science*, pages 292–300. Springer, 1979.
- 30 George V. Gens and Eugene V. Levner. Approximation algorithm for some scheduling problems. *Engng. Cybernetics*, 6:38–46, 1978.
- 31 Beat Gfeller. Finding longest approximate periodic patterns. In *WADS*, volume 6844 of *Lecture Notes in Computer Science*, pages 463–474. Springer, 2011.
- 32 Omer Gold and Micha Sharir. Improved bounds for 3SUM, k-SUM, and linear degeneracy. In *ESA*, volume 87 of *LIPICs*, pages 42:1–42:13, 2017.
- 33 Allan Grønlund and Seth Pettie. Threesomes, degenerates, and love triangles. *J. ACM*, 65(4):22:1–22:25, 2018.
- 34 Chloe Ching-Yun Hsu and Chris Umans. On multidimensional and monotone k-SUM. In *MFCS*, volume 83 of *LIPICs*, pages 50:1–50:13, 2017.
- 35 Ce Jin and Yinzhan Xu. Removing additive structure in 3SUM-based reductions. In *STOC*, pages 405–418. ACM, 2023.
- 36 Hans Kellerer, Renata Mansini, Ulrich Pferschy, and Maria Grazia Speranza. An efficient fully polynomial approximation scheme for the subset-sum problem. *J. Comput. Syst. Sci.*, 66(2):349–370, 2003.
- 37 Tomasz Kociumaka and Adam Polak. Bellman-Ford is optimal for shortest hop-bounded paths. In *ESA*, volume 274 of *LIPICs*, pages 72:1–72:10, 2023.
- 38 Marvin Künnemann and André Nusser. Polygon placement revisited: (degree of freedom + 1)-SUM hardness and an improvement via offline dynamic rectangle union. In *SODA*, pages 3181–3201. SIAM, 2022.

- 39 Marvin Künnemann, Ramamohan Paturi, and Stefan Schneider. On the fine-grained complexity of one-dimensional dynamic programming. In *ICALP*, volume 80 of *LIPICs*, pages 21:1–21:15, 2017.
- 40 Eduardo Sany Laber, Wilfredo Bardales Roncalla, and Ferdinando Cicalese. On lower bounds for the maximum consecutive subsums problem and the $(\min, +)$ -convolution. In *ISIT*, pages 1807–1811. IEEE, 2014.
- 41 Xiao Mao. $(1 - \epsilon)$ -approximation of knapsack in nearly quadratic time. *CoRR*, abs/2308.07004, 2023.
- 42 Marcin Mucha, Karol Wegrzycki, and Michal Włodarczyk. A subquadratic approximation scheme for partition. In *SODA*, pages 70–88. SIAM, 2019.
- 43 Mihai Pătraşcu. Towards polynomial lower bounds for dynamic problems. In *STOC*, pages 603–610. ACM, 2010.
- 44 Liam Roditty and Virginia Vassilevska Williams. Fast approximation algorithms for the diameter and radius of sparse graphs. In *STOC*, pages 515–524. ACM, 2013.
- 45 Virginia Vassilevska Williams. On some fine-grained questions in algorithms and complexity. In *Proc. of the International Congress of Mathematicians, ICM'18*, pages 3447–3487, 2018.
- 46 Virginia Vassilevska Williams and Ryan Williams. Finding, minimizing, and counting weighted subgraphs. *SIAM J. Comput.*, 42(3):831–854, 2013.