ORIGINAL ARTICLE

Approximating subset sum ratio via partition computations

Giannis Alonistiotis[1] · **Antonis Antonopoulos[1]** · **Nikolaos Melissinos[2]** ·
Aris Pagourtzis[1,3] · **Stavros Petsalakis[1]** · **Manolis Vasilakis[4]**

Abstract

We present a new FPTAS for the SUBSET SUM RATIO problem, which, given a set of integers, asks for two disjoint subsets such that the ratio of their sums is as close to 1 as possible. Our scheme makes use of exact and approximate algorithms for PARTITION, and clearly showcases the close relationship between the two algorithmic problems. Depending on the relationship between the size of the input set n and the error margin ε, we improve upon the best currently known algorithm of Melissinos and Pagourtzis [COCOON 2018] of complexity $\mathcal{O}(n^4/\varepsilon)$. In particular, the exponent of n in our proposed scheme may decrease down to 2, depending on the PARTITION algorithm used.

1 Introduction

One of Karp's 21 NP-complete problems [20], SUBSET SUM has seen astounding progress over the last few years. Koiliaris and Xu [23], Bringmann [10] and Jin and Wu [19] have

✉ Aris Pagourtzis
pagour@cs.ntua.gr

Giannis Alonistiotis
ialonistiotis@corelab.ntua.gr

Antonis Antonopoulos
aanton@corelab.ntua.gr

Nikolaos Melissinos
nikolaos.melissinos@fit.cvut.cz

Stavros Petsalakis
spetsalakis@corelab.ntua.gr

Manolis Vasilakis
emmanouil.vasilakis@dauphine.eu

[1] School of Electrical and Computer Engineering, National Technical University of Athens, 15780 Zografou, Greece

[2] Department of Theoretical Computer Science, Faculty of Information Technology, Czech Technical University in Prague, Prague, Czech Republic

[3] Archimedes Unit, Athena Research Center, 15125 Marousi, Greece

[4] Université Paris-Dauphine, CNRS UMR7243, LAMSADE, PSL University, 75016 Paris, France

presented pseudopolynomial algorithms resulting in substantial improvements over the long-standing standard approach of Bellman [7] and the improvement by Pisinger [31]. Moreover, the latter two algorithms [10, 19] match the SETH-based lower bounds proved in [1]. Additionally, recently there has been progress in the approximation scheme of SUBSET SUM, the first such improvement in over 20 years, with a new algorithm introduced by Bringmann and Nakos [8], as well as corresponding lower bounds obtained through the lens of fine-grained complexity.

A thoroughly studied special case of SUBSET SUM is the PARTITION problem, which asks for a partition of the input set to two subsets such that the difference of their sums is minimum. Any algorithm solving the first applies to the latter, though recent progress [8, 16, 28] has shown that PARTITION may be solved more efficiently in the approximation setting. On the other hand, regarding exact solutions, no better algorithm has been developed, therefore SUBSET SUM algorithms remain the state of the art.

The EQUAL SUBSET SUM problem, which, given an input set, asks for two disjoint subsets of equal sum, is closely related to SUBSET SUM and PARTITION. It finds applications in multiple different fields, ranging from computational biology [12, 13] and computational social choice [24], to cryptography [32], to name a few. In addition, it is related to important theoretical concepts such as the complexity of search problems in the class TFNP [30].

The centerpiece of this paper is the SUBSET SUM RATIO problem, the optimization version of EQUAL SUBSET SUM, which asks, given an input set $S \subseteq \mathbb{N}$, for two disjoint subsets $S_1, S_2 \subseteq S$, such that the following ratio is minimized

$$\frac{\max \left\{ \sum_{s_i \in S_1} s_i, \sum_{s_j \in S_2} s_j \right\}}{\min \left\{ \sum_{s_i \in S_1} s_i, \sum_{s_j \in S_2} s_j \right\}}.$$

This problem is known to be NP-hard, and many FPTASes have been proposed over the years [6, 25, 29], all of which rely on some kind of scaling of the input elements. The current state of the art [25] achieves a running time of $\mathcal{O}(n^4/\varepsilon)$, leaving a significant gap in comparison with known approximation algorithms for the closely related SUBSET SUM and PARTITION problems, especially with respect to n. This leads to the natural question of whether we can improve this performance and achieve an FPTAS with a running time $\mathcal{O}(n^{c_1}/\varepsilon^{c_2})$, where either $c_1 < 4$ or $c_1 + c_2 < 5$. We manage to answer both questions in the affirmative, by presenting a novel approximation scheme which utilizes exact or approximate PARTITION algorithms and achieves running time[1] $\tilde{\mathcal{O}}(n^{2.3}/\varepsilon^{2.6})$ or $\tilde{\mathcal{O}}(n^2/\varepsilon^3)$ respectively. Our proposed algorithm significantly differs from previous approaches while it is the first to associate these closely related problems.

1.1 Related work

EQUAL SUBSET SUM as well as its optimization version called SUBSET SUM RATIO[6] are closely related to problems appearing in many scientific areas. Some examples include the PARTIAL DIGEST problem, which comes from computational biology [12, 13], the allocation of individual goods [24], tournament construction [22], and a variation of SUBSET SUM, called Multiple Integrated Sets SSP, which finds applications in the field of cryptography [32]. Furthermore, it is related to important concepts in theoretical computer science; for example, a restricted version of EQUAL SUBSET SUM lies in a subclass of the complexity

---

[1] Standard $\mathcal{O}^\star$ and $\tilde{\mathcal{O}}$ notation is used to hide polynomial and polylogarithmic factors respectively.

class TFNP, namely in PPP [30], a class consisting of search problems that always have a solution due to some pigeonhole argument, and no polynomial time algorithm is known for this restricted version.

EQUAL SUBSET SUM has been proven NP-hard by Woeginger and Yu [33] (see also the full version of [27] for an alternative proof) and several variations have been proven NP-hard by Cieliebak et al. [11, 14]. A 1.324-approximation algorithm has been proposed for SUBSET SUM RATIO in [33] and several FPTASes appeared in [6, 25, 29], the fastest so far being the one in [25] of complexity $\mathcal{O}(n^4/\varepsilon)$, the complexity of which also applies to various meaningful special cases, as shown in [26].

As far as exact algorithms are concerned, recent progress has shown that EQUAL SUBSET SUM can be solved probabilistically in $\mathcal{O}^\star(1.7088^n)$ time [27], faster than a standard "meet-in-the-middle" approach yielding an $\mathcal{O}^\star(3^{n/2}) \leq \mathcal{O}^\star(1.7321^n)$ time algorithm.

These problems are tightly connected to SUBSET SUM, which has seen impressive advances recently, due to Koiliaris and Xu [23] who gave a deterministic $\tilde{\mathcal{O}}(\sqrt{n}t)$ algorithm, where n is the number of input elements and t is the target, and Bringmann [10] who gave a $\tilde{\mathcal{O}}(n+t)$ randomized algorithm, which is essentially optimal under SETH [1]. See also [3] for an extension of these algorithms to a more general setting. Jin and Wu subsequently proposed a simpler randomized algorithm [19] achieving the same bounds as [10], which however seems to only solve the decision version of the problem. Recently, Bringmann and Nakos [9] have presented an $\mathcal{O}\left(|\mathcal{S}_t(Z)|^{4/3}\mathrm{poly}(\log t)\right)$ algorithm, where $\mathcal{S}_t(Z)$ is the set of all subset sums of the input set Z that are smaller than t, based on top-k convolution.

PARTITION shares the complexity of SUBSET SUM regarding exact solutions, where the meet-in-the-middle approach [18] from the 70's remains the state of the art as far as algorithms dependent on n are concerned. On the other hand, one can approximate PARTITION more efficiently than SUBSET SUM unless the min-plus convolution conjecture [15] is false. In particular, Bringmann and Nakos [8] have presented the first improvement for the latter in over 20 years, since the scheme of [21] had remained the state of the art. Moreover, in their paper they have shown that developing a significantly better algorithm would contradict said conjecture. Furthermore, they develop an approximation scheme for PARTITION utilizing min-plus convolution computations, improving upon the recent work of Mucha et al. [28] and circumventing the lower bounds established for SUBSET SUM in their work. Very recently, Deng, Jin and Mao [16] presented an even faster approximation algorithm for PARTITION, further widening the gap between the complexities of both problems in the approximation setting.

1.2 Our contribution

We present a novel approximation scheme for the SUBSET SUM RATIO problem, which, depending on the relationship between n and ε, improves upon the best existing approximation scheme of [25]. Our algorithm significantly differs from previous approaches, which in most cases rely on some kind of scaling of the input elements, and instead makes use of either exact or approximation algorithms for PARTITION. In particular, we first partition the input elements into small and large, and then prove that we can either easily find an approximate solution involving only large elements or there are at most $\log(n/\varepsilon^2)$ of them. In the latter case, in order to approximate SUBSET SUM RATIO it suffices to solve instances of PARTITION on all the subsets of large elements, i.e., polynomially many instances, each of size at most $\log(n/\varepsilon^2)$. By leveraging known PARTITION algorithms in the second case, we manage to improve upon previous FPTASes. In the case of exact computations, we show that by employing

such a PARTITION algorithm of complexity $\mathcal{O}^\star(2^{\alpha n})$, our proposed scheme runs in time $\tilde{\mathcal{O}}(n \cdot (n/\varepsilon^2)^{\log(1+2^\alpha)})$, for some constant $\alpha > 0$. It is already known that such an algorithm exists for $\alpha = 1/2$ [18], and any further improvements will positively affect our FPTAS. On the other hand, using the approximation algorithm of Kellerer et al. [21] we achieve a running time of $\tilde{\mathcal{O}}(n^2/\varepsilon^3)$, while any improvement over it (e.g., [8, 16]) will only affect polylogarithmic factors of our scheme, as is further discussed in Sect. 5.

We start by presenting some necessary background in Sect. 2. Afterward, in Sect. 3 we introduce an FPTAS for a restricted version of the problem. Then, in Sect. 4, we explain how to make use of the algorithm presented in Sect. 3, in order to obtain an approximation scheme for the SUBSET SUM RATIO problem. The complexity of the final scheme is thoroughly analyzed in Sect. 5, followed by some possible directions for future research in Sect. 6.

Prior work In the current paper we improve upon the results of the preliminary version [2], by using approximate and exact PARTITION algorithms instead of SUBSET SUM computations.

2 Preliminaries

Let, for $x \in \mathbb{N}$, $[x] = \{z \in \mathbb{N} \mid 1 \leq z \leq x\}$ denote the set of integers in the interval $[1, x]$. Given a set $S \subseteq \mathbb{N}$, denote its *largest element* by $\max(S)$ and the sum of its elements by $\Sigma(S) = \sum_{s \in S} s$. If we are additionally given a value $\varepsilon \in (0, 1)$, define the following *partition* of its elements:

- The set of its *large* elements as $L(S, \varepsilon) = \{s \in S \mid s \geq \varepsilon \cdot \max(S)\}$. Note that $\max(S) \in L(S, \varepsilon)$, for any $\varepsilon \in (0, 1)$.
- The set of its *small* elements as $M(S, \varepsilon) = \{s \in S \mid s < \varepsilon \cdot \max(S)\}$.

In the following, since the values of the associated parameters will be clear from the context, they will be omitted and we will refer to these sets simply as L and M.

Definition 1 (PARTITION) Given a set X, compute a subset $X_p^* \subseteq X$, such that $\Sigma(X_p^*) = \max\{\Sigma(Z) \mid Z \subseteq X, \Sigma(Z) \leq \Sigma(X)/2\}$. Moreover, let $\overline{X_p^*} = X \backslash X_p^*$.

Definition 2 (Approximate PARTITION, from [28]) Given a set X and error margin ε, compute a subset $X_p \subseteq X$ such that $(1-\varepsilon) \cdot \Sigma(X_p^*) \leq \Sigma(X_p) \leq \Sigma(X_p^*)$. Moreover, let $\overline{X_p} = X \setminus X_p$.

3 Scheme for a restricted version

In this section, we present an FPTAS for the constrained version of the SUBSET SUM RATIO problem where we are only interested in approximating solutions that involve the largest element of the input set. In other words, one of the subsets of the optimal solution contains $\max(A) = a_n$ (assuming that $A = \{a_1, \ldots, a_n\}$ is the *sorted* input set); let r_{opt} denote the subset sum ratio of such an optimal solution. Our FPTAS will return a solution of ratio r, such that $1 \leq r \leq (1 + \varepsilon) \cdot r_{\text{opt}}$, for a given error margin $\varepsilon \in (0, 1)$; however, we allow that the sets of the returned solution do not necessarily satisfy the aforementioned constraint (i.e., a_n may not be involved in the approximate solution).

3.1 Outline of the algorithm

We now present a rough outline of Algorithm 1:

- At first, we search for approximate solutions involving exclusively large elements from $L(A, \varepsilon)$.
- To this end, we produce the subset sums formed by these large elements. If their number exceeds n/ε^2, then we can find an approximate solution.
- Otherwise, there are at most n/ε^2 subsets of large elements. In this case, we can find a solution by running an exact or an approximate PARTITION algorithm for each subset.
- In the case that the optimal solution involves small elements, we show that it suffices to add elements of $M(A, \varepsilon)$ in a greedy way.

Algorithm 1 ConstrainedSSR(A, ε, T)

Input : Set $A = \{a_1, \ldots, a_n\}$, error margin ε and table of partial sums T.
Output : $(1 + \varepsilon)$-apx of the optimal solution respecting the constraint.
1: Partition A to $M = \{a_i \in A \mid a_i < \varepsilon \cdot a_n\}$ and $L = \{a_i \in A \mid a_i \geq \varepsilon \cdot a_n\}$.
2: Split interval $[0, n \cdot a_n]$ to n/ε^2 bins of size $\varepsilon^2 \cdot a_n$.
3: **while** filling the bins with the subset sums of L **do**
4:     **if** two subset sums correspond to the same bin **then**
5:         **return** an apx solution based on these. $\triangleright$ $\mathcal{O}(n/\varepsilon^2)$ complexity.
6:     **end if**
7: **end while**
8: $2^{|L|} \leq n/\varepsilon^2 \iff |L| \leq \log(n/\varepsilon^2)$.
9: **for** each subset of large elements containing a_n **do** $\triangleright$ $\mathcal{O}(n/\varepsilon^2)$ subsets.
10:     Solve corresponding PARTITION instance. $\triangleright$ Complexity in Section 5.
11:     Add small elements. $\triangleright$ $\mathcal{O}(\log n)$ complexity, see Section 3.3.
12: **end for**

3.2 Solution involving exclusively large elements

We firstly search for a $(1 + \varepsilon)$-approximate solution with $\varepsilon \in (0, 1)$, without involving any of the elements that are smaller than $\varepsilon \cdot a_n$. Let $M = \{a_i \in A \mid a_i < \varepsilon \cdot a_n\}$ be the set of small elements and $L = A \backslash M = \{a_i \in A \mid a_i \geq \varepsilon \cdot a_n\}$ be the set of large elements.

After partitioning the input set, we split the interval $[0, n \cdot a_n]$ into smaller intervals, called bins, of size $l = \varepsilon^2 \cdot a_n$ each, as depicted in Fig. 1.

Thus, there are a total of $B = n/\varepsilon^2$ bins. Notice that each possible subset of the input set will belong to a respective bin constructed this way, depending on its sum. Additionally, if two sets correspond to the same bin, then the difference of their subset sums will be at most l.

The next step of our algorithm is to generate all the possible subset sums, occurring from the set of large elements L. The complexity of this procedure is $\mathcal{O}\left(2^{|L|}\right)$, where $|L|$ is the cardinality of set L. Notice however, that it is possible to bound the number of the produced subset sums by the number of bins B, since if two sums belong to the same bin they constitute a solution, as shown in Lemma 1, in which case the algorithm terminates in time $\mathcal{O}(n/\varepsilon^2)$.
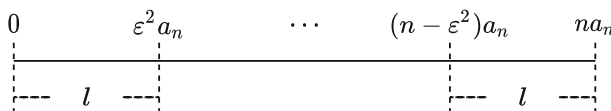

Fig. 1 Split of the interval $[0, n \cdot a_n]$ to bins of size l

Lemma 1 *If two subsets correspond to the same bin, we can find a $(1 + \varepsilon)$-approximation solution.*

Proof Suppose there exist two sets $L_1, L_2 \subseteq L$ whose sums correspond to the same bin, with $\Sigma(L_1) \leq \Sigma(L_2)$. Notice that there is no guarantee regarding the disjointness of said subsets, thus consider $L_1' = L_1 \setminus L_2$ and $L_2' = L_2 \setminus L_1$, for which it is obvious that $\Sigma(L_1') \leq \Sigma(L_2')$. Additionally, assume that $L_1' \neq \emptyset$. Then it holds that

$$\Sigma(L_2') - \Sigma(L_1') = \Sigma(L_2) - \Sigma(L_1) \leq l.$$

Therefore, the sets L_1' and L_2' constitute a $(1 + \varepsilon)$-approximation solution, since

$$\frac{\Sigma(L_2')}{\Sigma(L_1')} \leq \frac{\Sigma(L_1') + l}{\Sigma(L_1')} = 1 + \frac{l}{\Sigma(L_1')}$$

$$\leq 1 + \frac{\varepsilon^2 \cdot a_n}{\varepsilon \cdot a_n} = 1 + \varepsilon$$

where the last inequality is due to the fact that $L_1' \subseteq L$ is composed of elements $\geq \varepsilon \cdot a_n$, thus $\Sigma(L_1') \geq \varepsilon \cdot a_n$.

It remains to show that $L_1' \neq \emptyset$. Assume that $L_1' = \emptyset$. This implies that $L_1 \subseteq L_2$ and since we consider each subset of L only once and the input is a set and not a multiset, it holds that $L_1 \subset L_2 \implies L_2' \neq \emptyset$. Since L_1 and L_2 correspond to the same bin, it holds that

$$\Sigma(L_2) - \Sigma(L_1) \leq l \implies \Sigma(L_2') - \Sigma(L_1') \leq l \implies \Sigma(L_2') \leq l$$

which is a contradiction, since L_2' is a non empty subset of L, which is comprised of elements greater than or equal to $\varepsilon \cdot a_n$, hence $\Sigma(L_2') \geq \varepsilon \cdot a_n > \varepsilon^2 \cdot a_n = l$, since $\varepsilon < 1$.   □

Consider an ε' such that $(1 + \varepsilon')/(1 - \varepsilon') \leq 1 + \varepsilon$ for all $\varepsilon \in (0, 1)$ (the exact value of ε' will be computed in Sect. 5).

If every produced subset sum of the previous step belongs to a distinct bin, then, we can infer that the number of subsets of large elements is bounded by n/ε^2. Moreover, we can prove the following lemma.

Lemma 2 *If the optimal ratio r_{opt} involves sets S_1^*, S_2^* consisting of only large elements, with $S_1^* \cup S_2^* = S^* \subseteq L$ and $a_n \in S^*$, then $\Sigma(\overline{S_p})/\Sigma(S_p) \leq (1 + \varepsilon) \cdot r_{opt}$, where S_p is a $(1 - \varepsilon')$-apx solution to the PARTITION problem on input S^*.*

Proof Assume that $\Sigma(S_1^*) \leq \Sigma(S_2^*)$. Note that sets S_1^*, S_2^* are also the optimal solution of the PARTITION problem on input S^*. By running a $(1 - \varepsilon')$ approximate PARTITION algorithm on input set S^*, we obtain the sets S_1, S_2 with $\Sigma(S_1) \leq \Sigma(S_2)$, where $S_1 = S_p$ and $S_2 = \overline{S_p}$. Then,

$$\frac{\Sigma(S_2)}{\Sigma(S_1)} \leq \frac{\Sigma(S_2^*) + \varepsilon' \cdot \Sigma(S_1^*)}{(1 - \varepsilon')\Sigma(S_1^*)}$$

$$\leq \frac{\Sigma(S_2^*) + \varepsilon' \cdot \Sigma(S_2^*)}{(1 - \varepsilon')\Sigma(S_1^*)}$$

$$= \frac{1 + \varepsilon'}{1 - \varepsilon'} \cdot \frac{\Sigma(S_2^*)}{\Sigma(S_1^*)}$$

$$\leq (1 + \varepsilon) \cdot r_{opt}$$

where we used the fact that $(1 - \varepsilon') \cdot \Sigma(S_1^*) \leq \Sigma(S_1)$ as well as $\Sigma(S_2) \leq \Sigma(S_2^*) + \varepsilon' \cdot \Sigma(S_1^*)$.   □

Therefore, we have proved that when the optimal solution consists of sets comprised of only large elements, it is possible to find a $(1 + \varepsilon)$-approximation solution for the constrained SUBSET SUM RATIO problem by running a $(1 - \varepsilon')$-approximation algorithm for PARTITION with input the union of said large elements. In order to do so, it suffices to consider as input all the $2^{|L|-1}$ subsets of L containing a_n and each time run a $(1 - \varepsilon')$-approximation PARTITION algorithm. The total cost of this procedure will be thoroughly analyzed in Sect. 5 and depends on the algorithm used.

It is important to note that by utilizing an (exact or approximation) algorithm for PARTITION, we establish a connection between the complexities of PARTITION and approximating SUBSET SUM RATIO in a way that any future improvement in the first carries over to the second.

3.3 General $(1 + \varepsilon)$-approximate solutions

Whereas we previously considered optimal solutions involving exclusively large elements, here we will search for approximations for those optimal solutions that use all the elements of the input set, hence include small elements, and satisfy our constraint (i.e. a_n belongs to the optimal solution sets). We will prove that in order to approximate those optimal solutions, it suffices to consider only the $(1 - \varepsilon')$-apx solutions of the PARTITION problem corresponding to each subset of large elements and add small elements to them. In other words, instead of considering any two random disjoint subsets consisting of large elements[2] and subsequently adding to these the small elements, we can consider only the $(1 - \varepsilon')$-approximate solutions to the PARTITION problem computed in the previous step, ergo, at most $B = n/\varepsilon^2$ configurations regarding the large elements. Moreover, we will prove that it suffices to add the small elements to our solution in a greedy way.

Since the algorithm has not detected a solution so far, due to Lemma 1 every computed subset sum of set L belongs to a different bin. Thus, their total number is bounded by the number of bins B, i.e.

$$2^{|L|} \leq \left(\frac{n}{\varepsilon^2}\right) \iff |L| \leq \log\left(\frac{n}{\varepsilon^2}\right)$$

We proceed by additionally involving small elements into our solutions in order to reduce the difference between the sums of the sets, thus reducing their ratio.

Lemma 3 *Assume that we are given the $(1 - \varepsilon')$-apx solutions for the PARTITION problem on every subset of large elements containing a_n. Then, a $(1 + \varepsilon)$-approximation solution for the constrained version of SUBSET SUM RATIO can be found, when the optimal solution involves small elements.*

Proof Let S_1^*, S_2^* be disjoint subsets that form an optimal solution for the constrained version of SUBSET SUM RATIO, where:

- $\Sigma(S_1^*) \leq \Sigma(S_2^*)$ and $a_n \in S^* = S_1^* \cup S_2^*$.
- $S_1^* = L_1^* \cup M_1^*$ and $S_2^* = L_2^* \cup M_2^*$, where $L_1^*, L_2^* \subseteq L$ and $M_1^*, M_2^* \subseteq M$.
- $M_1^* \cup M_2^* \neq \emptyset$.

Moreover, let L_p^* and $\overline{L_p^*}$ be the optimal solution of the PARTITION problem on input $L^* = L_1^* \cup L_2^*$, while L_p and $\overline{L_p}$ be the sets returned by a $(1 - \varepsilon')$-apx algorithm. Then, it holds that:

---

[2] Note that the number of these random pairs is $2 \cdot 3^{|L|-1}$, since a_n is necessarily part of the solution.

- $\Sigma(L_p^*) \le \Sigma(\overline{L_p^*})$ and $\Sigma(\overline{L_p^*}) - \Sigma(L_p^*) \le |\Sigma(L^* \backslash X) - \Sigma(X)|, \forall X \subseteq L^*$.
- $(1 - \varepsilon') \cdot \Sigma(L_p^*) \le \Sigma(L_p) \le \Sigma(L_p^*)$.
- $\Sigma(\overline{L_p^*}) \le \Sigma(\overline{L_p}) \le \Sigma(\overline{L_p^*}) + \varepsilon' \cdot \Sigma(L_p^*) \le (1 + \varepsilon') \cdot \Sigma(\overline{L_p^*})$.
- $a_n \le \Sigma(\overline{L_p^*})$, since $a_n \in L^*$.

Case 1. Suppose that $\Sigma(L_p) + \Sigma(M) \ge \Sigma(\overline{L_p})$. In this case, there exists k such that $M_k = \{a_i \in M \mid i \in [k]\} \subseteq M$ and $0 \le \Sigma(L_p \cup M_k) - \Sigma(\overline{L_p}) \le \varepsilon \cdot a_n$, since all elements of M have value less than $\varepsilon \cdot a_n$. Hence,

$$1 \le \frac{\Sigma(L_p \cup M_k)}{\Sigma(\overline{L_p})} \le 1 + \frac{\varepsilon \cdot a_n}{\Sigma(\overline{L_p})} \le 1 + \frac{\varepsilon \cdot a_n}{a_n} = 1 + \varepsilon.$$

Case 2. Alternatively, it holds that $\Sigma(L_p) + \Sigma(M) < \Sigma(\overline{L_p})$. Then,

$$
\begin{aligned}
\frac{\Sigma(\overline{L_p})}{\Sigma(L_p \cup M)} &= \frac{\Sigma(\overline{L_p})}{\Sigma(L_p) + \Sigma(M)} \\
&\le \frac{(1 + \varepsilon') \cdot \Sigma(\overline{L_p^*})}{(1 - \varepsilon') \cdot \Sigma(L_p^*) + \Sigma(M)} \\
&\le \frac{1 + \varepsilon'}{1 - \varepsilon'} \cdot \frac{\Sigma(\overline{L_p^*})}{\Sigma(L_p^*) + \Sigma(M)} \\
&\le (1 + \varepsilon) \cdot \frac{\Sigma(\overline{L_p^*})}{\Sigma(L_p^*) + \Sigma(M)}.
\end{aligned}
$$

If $\Sigma(L_p^*) + \Sigma(M) \ge \Sigma(\overline{L_p^*})$, then it follows that $\frac{\Sigma(\overline{L_p})}{\Sigma(L_p \cup M)} \le 1 + \varepsilon$. On the other hand, if $\Sigma(L_p^*) + \Sigma(M) < \Sigma(\overline{L_p^*})$, then it follows that $\Sigma(S_1^*) = \Sigma(L_p^* \cup M)$ and $\Sigma(S_2^*) = \Sigma(\overline{L_p^*})$, therefore $\frac{\Sigma(\overline{L_p})}{\Sigma(L_p \cup M)} \le (1 + \varepsilon) \cdot \frac{\Sigma(S_2^*)}{\Sigma(S_1^*)}$. □

3.4 Adding small elements efficiently

Here, we will describe a method to efficiently add small elements to our sets. In particular, we search for some k such that $0 \le \Sigma(L_p \cup M_k) - \Sigma(\overline{L_p}) \le \varepsilon \cdot a_n$, where $M_k = \{a_i \in M \mid i \in [k]\}$. Notice that if $\Sigma(M) \ge \Sigma(\overline{L_p}) - \Sigma(L_p)$, there always exists such a set M_k, since by definition, each element of set M is smaller than $\varepsilon \cdot a_n$. In order to determine M_k, we make use of an array of partial sums $T[k] = \Sigma(M_k)$, where $k \le |M|$. Since T is sorted, we can find k in $\mathcal{O}(\log|L|) = \mathcal{O}(\log n)$ using binary search.

4 Final algorithm

The algorithm presented in the previous section constitutes an approximation scheme for SUBSET SUM RATIO when one of the solution subsets contains the maximum element of the input set. Thus, in order to solve the SUBSET SUM RATIO problem, it suffices to run the previous algorithm n times, where n depicts the cardinality of the input set A, while each time removing the max element of A.

    In particular, suppose that the optimal solution involves disjoint sets S_1^* and S_2^*, where $a_k = \max(S_1^* \cup S_2^*)$. There exists an iteration for which the algorithm considers as input

the set $A_k = \{a_1, \ldots, a_k\}$. In this iteration, the element a_k is the largest element and the algorithm searches for an approximation of the optimal solution for which a_k is contained in one of the solution subsets. The optimal solution of the unconstrained version of SUBSET SUM RATIO has this property so the ratio of the approximate solution that the algorithm of the previous section returns is at most $(1 + \varepsilon)$ times the optimal.

Consequently, n repetitions of the algorithm suffice to construct an FPTAS for SUBSET SUM RATIO. Notice that if at some repetition, the sets returned due to the algorithm of Sect. 3 have ratio at most $1 + \varepsilon$, then this ratio successfully approximates the optimal ratio $r_{\text{opt}} \geq 1$, since $1 + \varepsilon \leq (1 + \varepsilon) \cdot r_{\text{opt}}$, therefore they constitute an approximate solution.

Algorithm 2 SSR(A, ε)

Input : Sorted set $A = \{a_1, \ldots, a_n\}$ and error margin ε.
Output : $(1 + \varepsilon)$-apx of the optimal solution for SUBSET SUM RATIO.
1: Create array T such that $T[k] = \sum_{i=1}^{k} a_i$.   ▷ $\mathcal{O}(n)$ time.
2: **for** $i = n, \ldots, 1$ **do**
3:     ConstrainedSSR$(\{a_1, \ldots, a_i\}, \varepsilon, T)$
4: **end for**

---

5 Complexity

The total complexity of the final algorithm is determined by three distinct operations, over the n iterations of the algorithm:

1. The cost to compute all the possible subset sums occurring from large elements. It suffices to consider the case where this is bounded by the number of bins $B = n/\varepsilon^2$, due to Lemma 1.
2. The cost to compute an exact or $(1 - \varepsilon')$-apx PARTITION solution on each subset of large elements. The cost of this operation will be analyzed in the following subsection.
3. The cost to include small elements to the PARTITION solutions. There are B such solutions, and each requires $\mathcal{O}(\log n)$ time, and thus the total time required is $\mathcal{O}\left(\frac{n}{\varepsilon^2} \cdot \log n\right)$.

5.1 Complexity of partition computations

Using exact partition computations

First, we will consider the case where we compute the optimal solution of the PARTITION problem. In order to do so, we will use the standard meet-in-the-middle algorithm [18] for SUBSET SUM, and in the following we analyze its complexity.

Let subset $L' \subseteq L$ such that $|L'| = k$, and suppose we are given an exact PARTITION algorithm of complexity $\mathcal{O}(2^{\alpha k} \cdot k^{\beta})$, for some constants α, β. Notice that the number of subsets of L of cardinality k is $\binom{|L|}{k}$ and that $|L| \leq \log(n/\varepsilon^2)$. Then, it holds that

$$\sum_{k=0}^{|L|} \binom{|L|}{k} \cdot 2^{\alpha k} \cdot k^{\beta} \leq |L|^{\beta} \cdot \sum_{k=0}^{|L|} \binom{|L|}{k} \cdot 2^{\alpha k}$$

$$= |L|^{\beta} \cdot \left(1 + 2^{\alpha}\right)^{|L|}$$

$$= |L|^{\beta} \cdot 2^{|L| \log(1+2^{\alpha})}$$
$$\leq \log^{\beta}(n/\varepsilon^2) \cdot (n/\varepsilon^2)^{\log(1+2^{\alpha})}$$

where we used the binomial theorem. By employing the meet in the middle algorithm [18], where $\alpha = 1/2$ and $\beta = 1$, it follows that $\log(1 + 2^{\alpha}) = 1.271\ldots < 1.3$. Consequently, the complexity to solve the PARTITION problem for all the subsets of large elements is

$$\mathcal{O}\left(\frac{n^{1.3}}{\varepsilon^{2.6}} \cdot \log(n/\varepsilon^2)\right) = \tilde{\mathcal{O}}\left(\frac{n^{1.3}}{\varepsilon^{2.6}}\right)$$

Using approximate partition computations

Here we will analyze the complexity in the case we run an approximate PARTITION algorithm in order to compute the $(1 - \varepsilon')$-approximation solutions.

For subset $L' \subseteq L$, we run an approximate PARTITION algorithm with error margin ε' such that

$$\frac{1 + \varepsilon'}{1 - \varepsilon'} \leq 1 + \varepsilon \iff \varepsilon' \leq \frac{\varepsilon}{2 + \varepsilon}$$

and by choosing the maximum such ε', it holds that

$$\varepsilon' = \frac{\varepsilon}{2 + \varepsilon} \implies \frac{1}{\varepsilon'} = \frac{2 + \varepsilon}{\varepsilon} = \frac{2}{\varepsilon} + 1 \implies \frac{1}{\varepsilon'} = \mathcal{O}\left(\frac{1}{\varepsilon}\right)$$

Since there are at most n/ε^2 subsets of large elements, we will need to run said algorithm at most n/ε^2 times on $|L'| \leq |L|$ elements and with error margin ε'.

Note that any approximate SUBSET SUM algorithm could be used in order to approximate PARTITION, such as the one presented by Kellerer et al. [21] of complexity $\mathcal{O}\left(\min\left\{\frac{n}{\varepsilon}, n + \frac{1}{\varepsilon^2} \cdot \log(1/\varepsilon)\right\}\right)$. In our case, with $|L| = \log(n/\varepsilon^2)$ and error margin ε', the total complexity is

$$\mathcal{O}\left(\frac{n}{\varepsilon^2} \cdot \min\left\{\frac{|L|}{\varepsilon'}, |L| + \frac{1}{(\varepsilon')^2} \cdot \log(1/\varepsilon')\right\}\right) =$$
$$\mathcal{O}\left(\frac{n}{\varepsilon^2} \cdot \min\left\{\frac{\log(n/\varepsilon^2)}{\varepsilon}, \log(n/\varepsilon^2) + \frac{1}{\varepsilon^2} \cdot \log(1/\varepsilon)\right\}\right) =$$
$$\tilde{\mathcal{O}}\left(\frac{n}{\varepsilon^3}\right).$$

Using the state of the art $\tilde{\mathcal{O}}(n + (1/\varepsilon)^{1.25})$ algorithm of Deng et al. [16] for approximating PARTITION, one could, in some cases, further improve the last term of the previous minimum. However, since the PARTITION instances that we are solving involve $|L| = \log(n/\varepsilon^2)$ elements, any improvement resulting from said approximation algorithm would only affect polylogarithmic factors. Due to this, the algorithm of Kellerer et al. has a better performance compared to other PARTITION approximation algorithms, in case we choose to ignore those factors. On the other hand, if one takes them into account, it might be preferable to use the aforementioned algorithm of Deng et al. (always depending on the relation between n and ε).

5.2 Total complexity

The total complexity of the algorithm occurs from the n distinct iterations required and depends on the algorithm chosen to find the (exact or approximate) solution to the PARTITION problem, since all of the presented algorithms dominate the time of the rest of the operations. Thus, by choosing the fastest one (depending on the relationship between n and ε), the final complexity is

$$\tilde{\mathcal{O}}\left(\min\left\{\frac{n^{2.3}}{\varepsilon^{2.6}}, \frac{n^2}{\varepsilon^3}\right\}\right)$$

6 Conclusion and future work

The main contribution of this paper, apart from the introduction of a new FPTAS for the SUBSET SUM RATIO problem, is the establishment of a connection between PARTITION and approximating SUBSET SUM RATIO. In particular, our scheme employs PARTITION computations, and any improvement in the latter will have an effect to its complexity.

Additionally, we establish that the complexity of approximating SUBSET SUM RATIO, expressed in the form $\tilde{\mathcal{O}}(n^{c_1}/\varepsilon^{c_2})$, has $c_1 < 2.3$ and $c_1 + c_2 < 5$, which is an improvement over all the previously presented FPTASes for the problem. Moreover, the exponent of n may go down to 2 if we employ approximation PARTITION algorithms, which is a significant improvement over the $\mathcal{O}(n^4/\varepsilon)$ algorithm of [25].

It is important to note however, that there is a distinct limit to the complexity that one may achieve for the SUBSET SUM RATIO problem using the techniques discussed in this paper, since although of polylogarithmic size, the number of PARTITION instances required to be solved is $\mathcal{O}(n^2/\varepsilon^2)$ in total. Consequently, an interesting natural question arising from our work, is whether one can further improve the complexity of the problem, possibly developing a $\mathcal{O}(n^{c_1}/\varepsilon^{c_2})$ algorithm, where $c_1 < 2$ or even $c_1 + c_2 < 4$.

As another direction for future research, we consider the use of exact SUBSET SUM or PARTITION algorithms parameterized by a *concentration* parameter β, as described in [4, 5], where they solve the decision version of SUBSET SUM. See also [17] for a use of this parameter under a pseudopolynomial setting. It would be interesting to investigate whether analogous arguments could be used to solve the optimization version.

Author Contributions All authors had an equal contribution in all parts of the paper. The name order is alphabetical.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Abboud, A., Bringmann, K., Hermelin, D., Shabtay, D.: Seth-based lower bounds for subset sum and bicriteria path. ACM Trans. Algorithms **18**(1), 6–1622 (2022). https://doi.org/10.1145/3450524
2. Alonistiotis, G., Antonopoulos, A., Melissinos, N., Pagourtzis, A., Petsalakis, S., Vasilakis, M.: Approximating subset sum ratio via subset sum computations. In: Combinatorial Algorithms - 33rd International Workshop, IWOCA 2022. Lecture Notes in Computer Science, vol. 13270, pp. 73–85. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-06678-8_6
3. Antonopoulos, A., Pagourtzis, A., Petsalakis, S., Vasilakis, M.: Faster algorithms for k-subset sum and variations. J. Comb. Optim. **45**(1), 24 (2023). https://doi.org/10.1007/s10878-022-00928-0
4. Austrin, P., Kaski, P., Koivisto, M., Nederlof, J.: Dense subset sum may be the hardest. In: 33rd Symposium on Theoretical Aspects of Computer Science, STACS 2016. LIPIcs, vol. 47, pp. 13–11314. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2016). https://doi.org/10.4230/LIPIcs.STACS.2016.13
5. Austrin, P., Kaski, P., Koivisto, M., Nederlof, J.: Subset sum in the absence of concentration. In: 32nd International Symposium on Theoretical Aspects of Computer Science, STACS 2015. LIPIcs, vol. 30, pp. 48–61. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2015). https://doi.org/10.4230/LIPIcs.STACS.2015.48
6. Bazgan, C., Santha, M., Tuza, Z.: Efficient approximation algorithms for the SUBSET-SUMS EQUALITY problem. J. Comput. Syst. Sci. **64**(2), 160–170 (2002). https://doi.org/10.1006/jcss.2001.1784
7. Bellman, R.E.: Dynamic Programming. Princeton University Press, Princeton (1957)
8. Bringmann, K., Nakos, V.: A fine-grained perspective on approximating subset sum and partition. In: Proceedings of the 2021 ACM-SIAM symposium on discrete algorithms, SODA 2021, pp. 1797–1815. SIAM, USA (2021). https://doi.org/10.1137/1.9781611976465.108
9. Bringmann, K., Nakos, V.: Top-k-convolution and the quest for near-linear output-sensitive subset sum. In: Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, pp. 982–995. ACM, New York, NY, USA (2020). https://doi.org/10.1145/3357713.3384308
10. Bringmann, K.: A near-linear pseudopolynomial time algorithm for subset sum. In: Proceedings of the twenty-eighth annual ACM-SIAM symposium on discrete algorithms, SODA 2017, pp. 1073–1084. SIAM, USA (2017). https://doi.org/10.1137/1.9781611974782.69
11. Cieliebak, M., Eidenbenz, S.J., Pagourtzis, A.: Composing equipotent teams. In: Fundamentals of computation theory, 14th international symposium, FCT 2003. Lecture Notes in Computer Science, vol. 2751, pp. 98–108. Springer, Berlin, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45077-1_10
12. Cieliebak, M., Eidenbenz, S.J., Penna, P.: Noisy data make the partial digest problem NP-hard. In: Algorithms in bioinformatics, third international workshop, WABI 2003. Lecture Notes in Computer Science, vol. 2812, pp. 111–123. Springer, Berlin, Heidelberg (2003). https://doi.org/10.1007/978-3-540-39763-2_9
13. Cieliebak, M., Eidenbenz, S.J.: Measurement errors make the partial digest problem np-hard. In: LATIN 2004: theoretical informatics, 6th Latin American symposium. Lecture Notes in Computer Science, vol. 2976, pp. 379–390. Springer, Berlin, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24698-5_42
14. Cieliebak, M., Eidenbenz, S.J., Pagourtzis, A., Schlude, K.: On the complexity of variations of equal sum subsets. Nord. J. Comput. **14**(3), 151–172 (2008)
15. Cygan, M., Mucha, M., Wegrzycki, K., Wlodarczyk, M.: On problems equivalent to (min, +)-convolution. ACM Trans. Algorithms **15**(1), 14–11425 (2019). https://doi.org/10.1145/3293465
16. Deng, M., Jin, C., Mao, X.: Approximating knapsack and partition via dense subset sums. In: Proceedings of the 2023 ACM-SIAM symposium on discrete algorithms, SODA 2023, pp. 2961–2979. SIAM, USA (2023). https://doi.org/10.1137/1.9781611977554.ch113

17. Dutta, P., Rajasree, M.S.: Algebraic algorithms for variants of subset sum. In: Algorithms and Discrete Applied Mathematics - 8th International Conference, CALDAM 2022. Lecture Notes in Computer Science, vol. 13179, pp. 237–251. Springer, Cham (2022). https://doi.org/10.1007/978-3-030-95018-7_19

18. Horowitz, E., Sahni, S.: Computing partitions with applications to the knapsack problem. J. ACM **21**(2), 277–292 (1974). https://doi.org/10.1145/321812.321823

19. Jin, C., Wu, H.: A simple near-linear pseudopolynomial time randomized algorithm for subset sum. In: 2nd symposium on simplicity in algorithms, SOSA 2019. OASIcs, vol. 69, pp. 17–1176. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2019). https://doi.org/10.4230/OASIcs.SOSA.2019.17

20. Karp, R.M.: Reducibility among combinatorial problems. In: Proceedings of a symposium on the complexity of computer computations. The IBM Research Symposia Series, pp. 85–103. Springer, Boston, MA (1972). https://doi.org/10.1007/978-1-4684-2001-2_9

21. Kellerer, H., Mansini, R., Pferschy, U., Speranza, M.G.: An efficient fully polynomial approximation scheme for the subset-sum problem. J. Comput. Syst. Sci. **66**(2), 349–370 (2003). https://doi.org/10.1016/S0022-0000(03)00006-0

22. Khan, M.A.: Some problems on graphs and arrangements of convex bodies. PRISM (2017). https://doi.org/10.11575/PRISM/10182

23. Koiliaris, K., Xu, C.: Faster pseudopolynomial time algorithms for subset sum. ACM Trans. Algorithms **15**(3), 40–14020 (2019). https://doi.org/10.1145/3329863

24. Lipton, R.J., Markakis, E., Mossel, E., Saberi, A.: On approximately fair allocations of indivisible goods. In: Proceedings of the 5th ACM conference on electronic commerce (EC-2004), pp. 125–131. ACM, New York, NY, USA (2004). https://doi.org/10.1145/988772.988792

25. Melissinos, N., Pagourtzis, A.: A faster FPTAS for the subset-sums ratio problem. In: Computing and Combinatorics—24th International Conference, COCOON 2018. Lecture Notes in Computer Science, vol. 10976, pp. 602–614. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-94776-1_50

26. Melissinos, N., Pagourtzis, A., Triommatis, T.: Approximation schemes for subset-sums ratio problems. Theor. Comput. Sci. **931**, 17–30 (2022). https://doi.org/10.1016/j.tcs.2022.07.027

27. Mucha, M., Nederlof, J., Pawlewicz, J., Wegrzycki, K.: Equal-subset-sum faster than the meet-in-the-middle. In: 27th annual european symposium on algorithms, ESA 2019. LIPIcs, vol. 144, pp. 73–17316 (2019). https://doi.org/10.4230/LIPIcs.ESA.2019.73

28. Mucha, M., Wegrzycki, K., Wlodarczyk, M.: A subquadratic approximation scheme for partition. In: Proceedings of the thirtieth annual ACM-SIAM symposium on discrete algorithms, SODA 2019, pp. 70–88. SIAM, USA (2019). https://doi.org/10.1137/1.9781611975482.5

29. Nanongkai, D.: Simple FPTAS for the subset-sums ratio problem. Inf. Process. Lett. **113**(19–21), 750–753 (2013). https://doi.org/10.1016/j.ipl.2013.07.009

30. Papadimitriou, C.H.: On the complexity of the parity argument and other inefficient proofs of existence. J. Comput. Syst. Sci. **48**(3), 498–532 (1994). https://doi.org/10.1016/S0022-0000(05)80063-7

31. Pisinger, D.: Linear time algorithms for knapsack problems with bounded weights. J. Algorithms **33**(1), 1–14 (1999). https://doi.org/10.1006/jagm.1999.1034

32. Voloch, N.: Mssp for 2-d sets with unknown parameters and a cryptographic application. Contemp. Eng. Sci. **10**, 921–931 (2017). https://doi.org/10.12988/ces.2017.79101

33. Woeginger, G.J., Yu, Z.: On the equal-subset-sum problem. Inf. Process. Lett. **42**(6), 299–302 (1992). https://doi.org/10.1016/0020-0190(92)90226-L