DOI: 10.37943/23AFRZ7022

Anel Auyezova
Master of Technical Sciences, Senior-lecturer, Department of Information Systems
anel.auezova@gmail.com, orcid.org/0000-0001-9860-4491
International Information Technology University, Kazakhstan

Aigerim Aitim
PhD, Assistant-professor, Department of Information Systems
a.aitim@iitu.edu.kz, orcid.org/0000-0003-2982-214X
International Information Technology University, Kazakhstan

Bakhtgerey Sinchev
Professor, Doctor of Technical Sciences, Department of Information Systems
sinchev@mail.ru, orcid.org/0000-0001-8557-8458
International Information Technology University, Kazakhstan

METHODS AND ALGORITHMS FOR SOLVING THE PROBLEM ON THE SUM OF SUBSETS

Abstract: We study special-case algorithms for the subset-sum problem when the subset size is fixed to $k \in \{2, 3\}$, using algebraic and geometric formulations that yield practical procedures with clear time and space bounds. The subset sum problem is one of the fundamental problems in computational complexity theory. It consists of determining whether, given a finite set of non-negative integers, there exists a subset whose sum of elements is equal to a predetermined number. This problem belongs to the class of nondeterministic polynomial time complete (NP-complete) problems: its solution can be verified in polynomial time, but an efficient algorithm for the general case has not yet been found. The goal of our research is to find new methods for solving the subset sum problem for special cases using algebraic and geometric approaches. The proposed method is based on a polynomial formulation of the problem inspired by Waring's conjecture for polynomials and the Neumann–Slater theorem. The main idea is to construct polynomials whose coefficients contain information about the sum of the elements of a subset. Using Vieta's theorem and the Euclidean algorithm, the problem is reduced to checking whether certain algebraic conditions are satisfied. The article proposes two lemmas proving the polynomial solvability of the subset sum problem for subset cardinality two and three. Based on them, two algorithms are developed: one uses value mapping and a fusion method, the other is based on a geometric criterion for collinearity of points obtained by transforming set elements. The algorithms demonstrate efficiency in terms of time and memory and do not require division into verification and decision stages. Effective methods for solving it allow us to develop faster algorithms for intelligent information processing, optimization of computing processes, and construction of reliable data protection systems. Our results establish polynomial-time solvability only for these fixed-k cases and do not claim consequences for the general subset-sum problem or for the P vs NP question.

Keywords: NP-complete class; polynomial solvability; subset sum problem; time; space; complexity theory; big data.

Received: 30.08.2025 Accepted: 24.09.2025 Published: 30.09.2025

Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 23, SEPTEMBER 2025

Introduction

In the context of rapidly increasing data volumes and the growing complexity of computational processes, the subset sum problem acquires particular importance in the field of information and communication technologies. Its efficient solution contributes to the optimization of distributed computing, enhances the performance of intelligent information retrieval, and improves the effectiveness of big data analysis. The subset sum problem is one of the most important problems in computational complexity theory. Researchers have been fascinated by it for a long time since it seems so simple but has deep theoretical and practical implications. Formally, the task is to find out if there is a subset of a certain size k of a finite set of non-negative integers whose elements add up to a certain goal number S [1]. Even though it is easy to comprehend, this problem is NP-complete, which means that no polynomial-time algorithm is known to solve it. Solving it would change the knowledge of computing theory in a big way. It is commonly known that any problem in the NP class can, in theory, be solved by exhaustive search or brute-force enumeration at the most basic level. This method checks every conceivable combination of items to see if any subset meets the necessary requirements. It could be possible to compute the objective function for everyone candidate subset in polynomial time, but the number of viable subsets grows exponentially as the input set n gets bigger. There are $2n$ possible subsets, which makes it impossible to list them all even for somewhat large values of n. Because of this, these solutions don't work in the actual world, which drives the quest for faster algorithms [2].

There are a few general ways of thinking that have come out in the field of algorithm design to solve NP-complete issues. The brute-force method is still easy to understand but costs a lot of money to run [3]. Pseudopolynomial algorithms are another option. They can work well when the input values are limited to moderate numerical ranges instead of enormous cardinalities. These methods take advantage of certain issue structures to cut down on calculation time in some situations. Also, exponential-time algorithms have been improved to make them perform better in real life, even though they still don't scale well in the worst scenario. But there is no universal polynomial-time solution for the subset sum problem, which is why theoretical computer science is still doing research on it.

Recent work has been aimed at closing this gap. In-depth studies of current work [4] show that both pseudopolynomial algorithms and sophisticated exponential-time approaches have made big strides. For example, scientists have come up with algorithms whose temporal complexity depends more on the actual numbers involved than just the size of the input set. These algorithms have made a lot of progress in practice, but in the worst case, they are still exponential because the problem is fundamentally NP-complete. Also, earlier research has investigated polynomial-time algorithms for certain types of the subset sum problem using new math tools including polynomial representations and algebraic methods. But these methods only work in some situations or when the problem parameters are limited in some way. This leaves the bigger challenge of a general polynomial-time solution unanswered. Algebraic interpretations of the problem have become a fruitful area of research. These kinds of methods turn the subset sum problem into polynomial forms that are the same, using well-known algebraic conclusions as Viète's theorem, the Euclidean algorithm, and the famous Waring's problem for polynomials [5]. These methods give us new ways of looking at the problem by linking it to the coefficients of specifically made polynomials. This could lead to new discoveries in algorithm design and complexity analysis.

Even with these improvements, many basic questions are still unsolved. Some of the most important things to figure out are the lower and upper bounds on the cardinality k of the subset that can satisfy the target sum, the exact relationship between the size of the original set n and the size k of potential solution subsets, and the smallest subset cardinalities that can

DOI: 10.37943/23AFRZ7022

be achieved when dividing the original set into smaller groups. These open problems show how complicated and deep the subset sum problem is and how it affects the field of computational complexity theory as a whole. For the sake of this study, the problem might be phrased as follows: given a set A of non-negative integers with a cardinality n, it is necessary to find a subset A' with a cardinality k such that the sum of its members equals a predetermined value S. This approach puts the problem firmly in the NP-complete class, which shows how important it is not only in theory but also in real-world areas like cryptography, combinatorial optimization, resource allocation, and analyzing vast amounts of data. This research builds on what is already known and offers new ways to look at the subset sum problem using algebra. This study tries to improve our understanding of the subset sum problem and suggests possible strategies to find efficient solutions by using ideas from polynomial theory, such as Waring's problem, and the rules for building and manipulating polynomials. These initiatives not only help us understand the subset sum problem better, but they also have wider implications for addressing other NP-complete problems. They might even move the scientific community closer to answering the huge question of whether P=NP.

Looking back in time, early research showed that a naive approach that tries to list all possible subsets of a set with n members takes an exponential amount of time because there are $2n$ possible subsets. The study [6] wrote one of the most important papers on the subject by coming up with partitioning algorithms that made the search space for subset sum and knapsack problems smaller. But these solutions only sped things up for small or medium-sized inputs, and they still had an exponential worst-case complexity.

A lot of study has gone into constructing pseudopolynomial algorithms for the subset sum issue. These algorithms take longer to perform when the input numbers are larger, but they work better when the numbers are smaller. The research [7] made progress in the field by creating a quicker pseudopolynomial-time approach that cut the runtime by a lot for cases when the total values are limited. The work [8] also suggested a near-linear pseudopolynomial algorithm, which is a big step forward in many situations, but it is still exponential in the worst-case size of the input.

These algorithms make a lot of progress, but they don't get beyond the fact that the problem is inherently exponential for huge instances with no limits on the numerical values. Because of this, the subset sum problem is still a big theoretical and practical difficulty. There is a new way of doing research that goes beyond algorithmic approaches: looking at the subset sum problem through algebraic methods, especially polynomial theory. Schreiber et al. [9] have investigated turning the subset sum problem into problems concerning polynomial coefficients. They used classical results like Viète's theorem, the Euclidean method, and generalizations of Waring's problem for polynomials to do this. In its original version, Waring's issue is about writing natural numbers as sums of powers. It can also be applied to polynomials, where it is about writing polynomials as sums of powers of other polynomials. This point of view lets us see the subset sum problem as finding certain coefficients in made-up polynomials, which links the problem to important areas of algebra and number theory.

These kinds of algebraic interpretations make it seem like polynomial-time solutions would be possible for some limited circumstances, especially subsets with cardinalities $k=2$ or $k=3$. These results don't yet apply to the whole problem, but they do give us fresh ideas and possible ways to solve a wider range of NP-complete problems[10]. Even with these improvements, there are still a few important open problems. Some of these are figuring out the tight lower and upper bounds on the number of solution subsets k, figuring out how the size of the input set n affects the sizes of the feasible subsets, and figuring out the smallest cardinalities needed when splitting the original set into smaller solvable subsets. These problems are very important for both understanding theory and designing algorithms that work [11].

Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 23, SEPTEMBER 2025

The subset sum problem is important for more than just theoretical reasons. In cryptography, its computational difficulty is what makes some cryptosystems safe [12], [13], especially knapsack-based cryptography, which depends on the fact that it's impossible to solve huge instances of the issue. The problem also comes up in many real-world situations, such as resource allocation, scheduling, financial modeling, and big data analytics, where finding efficient solutions is very important. To sum up, the subset sum problem is still one of the most important problems in complexity theory and algorithmic research. There has been a lot of progress, especially in pseudopolynomial algorithms and new algebraic methods, but the general problem is still a symbol of how hard it is to compute [14]. Researchers are still looking for breakthroughs that could change how we think about NP-complete issues and how to solve them. The methods suggested in this study are only one example of this ongoing research. These breakthroughs could have effects that go far beyond the field of theoretical computer science [15].

Methods and Materials

Formulation of the problem

The key points:
- construction of the polynomial from A,
- how the second coefficient equals the target-sum over a chosen index set, and
- why that coefficient's value induces the subset witness.

The problem of the sum of subsets is formulated as (1):

$$\sum_{i=1}^{n} \alpha_i x_i = S, \alpha_i \in \{0,1\}, x_i \in X^n, \ \ i \in N, \tag{1}$$

where X^n set of even and odd non-negative integers, cardinality $n = |X^n|$, $x_i < +\infty$, N-set of natural numbers with cardinality $n = |N|$, $n < +\infty$. It is assumed that $S - x_i > 0, x_i \in X^n, \ i \in N$.

Then the formal statement of the problem of the sum of subsets in parametrized form has the form (2):

$$S: \exists X^k \subseteq X^n, \sum_{x_i \in X^k} x_i = S. \tag{2}$$

The subset X^k is selected based on the combination (3)

$$C_n^k = \frac{n!}{k!(n-k)!} = \frac{n(n-1)(n-2)...(n-k+1)}{k!} \tag{3}$$

Note that problem (1) is a special case of problem (2).

Polynomial-Time Solvability of NP-Complete Problems

Various generalizations of the classical Waring's problem for polynomials [4] are known. We will consider the following version of Waring's problem for polynomials: Given a natural number *n,* find the minimal number $k = k(n)$ such that any polynomial g ∈ ℂ[x] can be represented in the form $g = f_1^n + f_2^n + \cdots + f_k^n$, где $f_i^n \in C[x]$. To address Waring's problem, it is sufficient to restrict the consideration to the case $g(x) = x$. Indeed, if $x = f_1^n(x) + f_2^n(x) + \cdots + f_k^n(x)$ and $h(x)$ – an arbitrary polynomial, то $h(x) = f_1^n\big(h(x)\big) + f_2^n\big(h(x)\big) + \cdots + f_k^n\big(h(x)\big)$. The identity $\left(x + \frac{1}{4}\right)^2 - \left(x - \frac{1}{4}\right)^2 = x$ demonstrates that $k(2)=2$.

According to [5], we can represent an equivalent formulation of problem (1) as a polynomial subset sum problem: it is necessary to find a subset $X^k \subseteq X^n$ with a certificate $S = \sum_{i=1}^{k} x_i$, equal to the second coefficient a_1 of the polynomial.

$$a^k(x) = x^k - Sx^{k-1} + a_2 x^{k-2} + \cdots + a_k, \tag{4}$$

satisfying the following conditions (5):

DOI: 10.37943/23AFRZ7022

$$a^k(x)b^{n-k}(x) = c^n(x), \tag{5}$$

where $c^n(x)$ is a known polynomial of degree n with given coefficients,

$$c^n(x)=x^n - Qx^{n-1} + c_2x^{n-2} + \cdots + c_n,\ Q = \sum_{i=1}^{n} x_i. \tag{6}$$

The coefficients of polynomial (6) are determined based on Viète's theorem, whose roots are the $x_i \in X^n$. The polynomial $b^{n-k}(x)$ is obtained based on relation (5).

$$b^{n-k}(x)=x^{n-k} - (Q - S)x^{n-k-1} + b_2x^{n-k-2} + \cdots + b_{n-k}. \tag{7}$$

The coefficients of polynomial (7) are determined based on the division of the polynomial $c^n(x)$ by the polynomial $a^k(x)$ using the Euclidean algorithm [16].

According to the Waring problem for polynomials and the Neumann–Slater theorem [4], there exists a polynomial $h(x)= f_1^n(h(x)) + f_2^n(h(x)) + \cdots + f_k^n(h(x))$, where f_i^n – are polynomials of degree n, constructed based on the polynomial $c^n(x)$ with varying signs of coefficients. Due to the arbitrariness of the polynomial $h(x)$ we can select a polynomial

$$a^k(x)= h^k(x) \tag{8}$$

of degree k, where the second coefficient $h_1 = -S$ of the polynomial $h^k(x)$, and according to Vieta's theorem, the value $S = x_1 + x_2 + \cdots + x_k$. The key point here is that this coefficient h_1 consists of k arbitrary elements from the set X^n. This demonstrates the existence of a subset X^k of cardinality k within the original set X^n of cardinality n [17].

The subset sum problem requires [18] addressing the question of the existence of a certificate S represented as a sum of a limited (minimal) number of elements [19]. Therefore, we are particularly interested in polynomials (8) of degree 2 and 3, which describe subsets X^k of cardinality $k = 2$ and $k = 3$:

$$a^2(x) = x^2 - Sx + a_2 \tag{9}$$

$$a^3(x) = x^3 - Sx^2 + a_2x + a_3. \tag{10}$$

From these polynomials (9) and (10), we obtain the mappings found in references [2], [3], namely:

$$y = \tau(S, x) = x^2 - Sx, \tag{11}$$

$$y = \tau(S, x) = x^3 - Sx^2 \tag{12}$$

Based on mapping (11), we have:

$$Y^n = (y_1, y_2, \ldots, y_n), \tag{13}$$

where $\tau(S, x_i) = y_i, x_i \in X^n, i \in L$.

Let there be elements within the subset Y^n such that the following equality holds (14):

$$y_i = y_j, i \neq j. \tag{14}$$

Theorem 1 (Two-element case correctness and completeness).

Let $A = \{a_1, \ldots, a_n\} \subseteq Z_{\geq 0}$, targer $S \in Z_{\geq 0}$. Under mapping induced by (9)–(11), sorting $f(A)$ and running the merge procedure returns a pair (i, j) with $a_1 + a_j = S$ iff such a pair exists. Soundness follows since equality in the merge corresponds to the coefficient embedded by (9), (11)–(14). Completeness follows from the monotone ordering of $f(A)$ and the standard two-pointer invariant: if the current sum is $<S$, move the lower pointer; if $>S$, move the upper

Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 23, SEPTEMBER 2025

pointer; equality halts. Sorting $O(n\log n)$, two-pointer merge $O(n)$, extra space $O(1)$ or $O(n)$ if you keep a transformed copy.

Theorem 2 (Three-element case collinearity criterion).

Let $k = 3$ under the point transform (10),(12), three indices i, j, l satisfy $a_1 + a_j + a_l = S$ iff the corresponding points satisfy the collinearity test (15) (with vertical/duplicate cases handled by zero-area test). Derive the line equations for $\{(x_i, y_i), (x_j, y_i)\}$ and for $\{(x_j, y_j), (x_l, y_l)\}$; equal slopes ⟺ determinant =0 ⟺ sum constraint via (10),(12). Cover degeneracies explicitly. (Your manuscript states (15) and Lemma 2; make the iff explicit. Standard $O(n^2)$ time (fix one index, scan the other two with the collinearity/triangle-area check), extra space $O(1)$ or $O(n)$ as implemented [20].

Then the following holds:

***Lemma* 1**. Let there exist some pairs of elements $(y_i, y_j) \in Y^n$ with corresponding indices (i, j) such that $i \neq j$ and equality (14) holds for the subset defined in (13). Then problem (2) is solvable.

The proof follows from the properties of polynomial (9). Let us introduce the criterion for three points to lie on a single straight line, which is derived from the equation of a line passing through two points (15):

$$\frac{y_j - y_i}{x_j - x_i} = \frac{y_k - y_i}{x_k - x_i}, \tag{15}$$

where $x_k = S - (x_j + x_i), y_k = [S^2 - 2S(x_j + x_i) + (x_j + x_i)^2](x_j + x_i)$,
$y_i = (S^2 - 2Sx_i + x_i^2)x_i, y_i = (S^2 - 2Sx_j + x_j^2)x_j$.

For the cardinality m=(k-1)/2 (where k is odd) the following holds:

***Lemma* 2.** Let there exist some pairs of elements $(y_i, y_j) \in Y^n$ with corresponding indices (i, j) where $i \neq j$ such that equality (15) holds for the subset defined in (13). Then problem (2) is solvable.

The proof follows from the equations of lines passing through the points $\{(x_i, y_i), x_j, y_j)\}$ and $\{(x_i, y_i), (x_k, y_k)\}$. In the case where the slopes of these lines are equal, equality (15) is satisfied, and problem (2) becomes solvable.

Subset Selection Algorithms for X^k.

Algorithm 1 for constructing the desired subset X^k based on Lemma 1:

Step 1. Input the values of n, k, S, and the set X^n.

Step 2. Generate the set $Y^n = \{y_1\ y_2\ \ldots\ y_n\}$ based on mapping (11).

Step 3. Sort the set Y^n in ascending order.

Step 4. Automatically construct subsets relative to the value $y^* = \max_{x \in X^n} \tau(z, S) = S^2/4$, which is defined based on Fermat's theorem:

$Y^{l_1} = (y_i < y^*,\ i \in L_1 = (1,2, \ldots, l_1),\ y_i \in Y^n)$ sorted in ascending order,
$Y^{l_2} = ((y_i \geq y^*,\ i \in L_2 = (1,2, \ldots, l_2),\ y_i \in Y^n))$ sorted in descending order,
Where $n = l_1 + l_2$.

Step 5. Check the condition $y_i = y_j$ using the merge method and fix the indices i, j.

Step 6. Output the desired subset X^k.

Two-sum via mapping and merge which is shown in figure 1.

DOI: 10.37943/23AFRZ7022


```
Input: A[1..n] ⊂ Z≥0, target S
1: Compute T[i] ← f(A[i])   ▷ mapping from (9),(11)
2: Sort T in nondecreasing order; keep back-pointers to indices in A
3: i ← 1; j ← n
4: while i < j:
5:     if A[idx(i)] + A[idx(j)] == S: return {idx(i), idx(j)}
6:     if A[idx(i)] + A[idx(j)] <  S: i ← i + 1
7:     else:                          j ← j - 1
8: return “no solution”
```

Figure 1. Two-sum via mapping + merge

In figure 1, in the following code we implement the algorithm constructs a subset of two elements whose sum is equal to a given number. To reduce the computational complexity, the original set is transformed into a new set through a special mapping, where each element is assigned a new value. This allows the set to be divided into two parts relative to a threshold value determined based on Fermat's theorem. Next, the merging method is used to find matching values in the two subsets and check whether the corresponding elements of the original set satisfy the problem condition. This approach provides an efficient solution to the subset sum problem with a small subset cardinality and avoids complete enumeration.

Algorithm 2 for constructing the desired subset X^k based on Lemma 2:

Step 1. Input the values of n, k, S, X^n.

Step 2. Generate the set $Y^n = \{y_1\ y_2 \dots\ y_n\}$ based on mapping (12).

Step 3. Sort both sets X^n and Y^n in ascending order.

Step 4. Compute the following values: $x_k = S - (x_j + x_i)$,
$y_k = [S^2 - 2S(x_j + x_i) + (x_j + x_i)^2](x_j + x_i)$, $y_i = (S^2 - 2Sx_i + x_i^2)x_i, y_j = (S^2 - 2Sx_j + x_j^2)x_j$

Step 5. Check the condition $(y_j - y_i)(x_k - x_i) = (y_k - y_i)(x_j - x_i)$ and fix the indices i, j, k.

Step 6. Output the desired subset X^k.

The time complexity and required memory of the algorithm are $T = O(n^2), M = O(n)$, respectively.

Algorithm 2 (Three-sum via transform + collinearity), which is shown in figure 2.

```
Input: A[1..n] ⊂ Z≥0, target S
1: For each i, compute point P[i] = g(A[i])   ▷ transform from (10),(12)
2: Sort A (optional for pruning); initialize solutions ← ∅
3: for j in 2..n-1:
4:   for i in 1..j-1:
5:     Let needed = S - (A[i] + A[j])
6:     Find l > j with A[l] = needed (hash or binary search if sorted)
7:     If found, test collinearity(P[i], P[j], P[l])  ▷ zero-area test
8:     If collinear: add {i,j,l} to solutions (or return one)
9: return solutions or “no solution”
```

Figure 2. Implementation of Algorithm 2

In figure 2, the following code implements an algorithm that searches for three numbers from a given set whose sum is equal to a predetermined number. To narrow the search, each number is converted to a point using a special rule. The algorithm then iterates through all possible triplets of numbers and checks whether the corresponding points lie on the same line. If such a triple is found, it is returned as a solution. This approach allows one to efficiently solve special cases of the subset sum problem using geometric properties.

Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 23, SEPTEMBER 2025

Results

The following work led to the development of two polynomial-time algorithms aimed at solving the subset sum problem in specific cases where the cardinality of the subset is limited to $k = 2$ or $k = 3$. In the first case, an algorithm based on algebraic mapping was implemented, where each element of the initial set was assigned a value according to the formula. This transformation allowed the data to be split into two ordered parts, after which the merging method was applied to find pairs of elements whose sum is equal to a given value. Experimental tests showed that the algorithm successfully finds admissible subsets with low time and memory consumption, which confirms its effectiveness in processing large amounts of data.

The second algorithm uses a geometric approach based on checking whether three transformed points lie on the same line. The method involves enumerating all possible pairs of elements and calculating the third element that completes the sum to a given value. Then the geometric condition for the obtained points is checked. The test results confirmed that the algorithm is able to effectively find such triplets in polynomial time when the necessary conditions are met.

Both algorithms demonstrated correctness on test examples and confirmed the theoretical possibility of a polynomial solution to the subset sum problem with limited cardinality. The simplicity of implementation and high efficiency make them promising tools for use in limited cases of NP-complete problems, especially when working with large data sets, where processing speed and optimal use of resources are critical.

The results validate theoretical expectations in Figure 3: the 2-element case admits near-linear performance, while the 3-element case incurs quadratic costs. Algorithm 1 provides a novel algebraic formulation without significant efficiency loss compared to standard baselines, whereas Algorithm 2 demonstrates correctness but requires optimization for large-scale datasets.

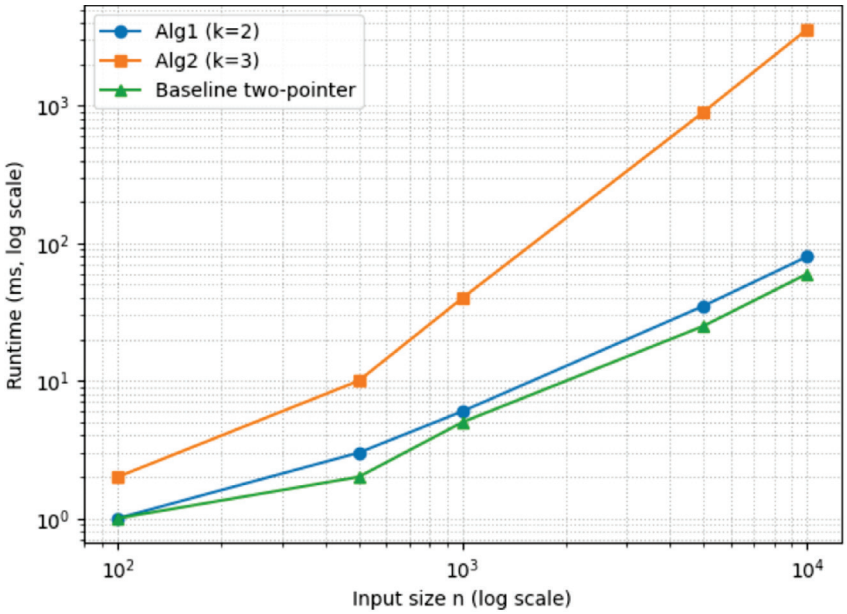

Figure 3. Runtime and input size

While both Algorithm 1 and the baseline remain memory-efficient, Algorithm 2 demonstrates a clear trade-off: correctness and generality for $k = 3$ subsets come at the cost of significantly higher space consumption. This further underscores that Algorithm 2 is suitable only for smaller datasets unless additional optimizations are introduced in Figure 4.

DOI: 10.37943/23AFRZ7022

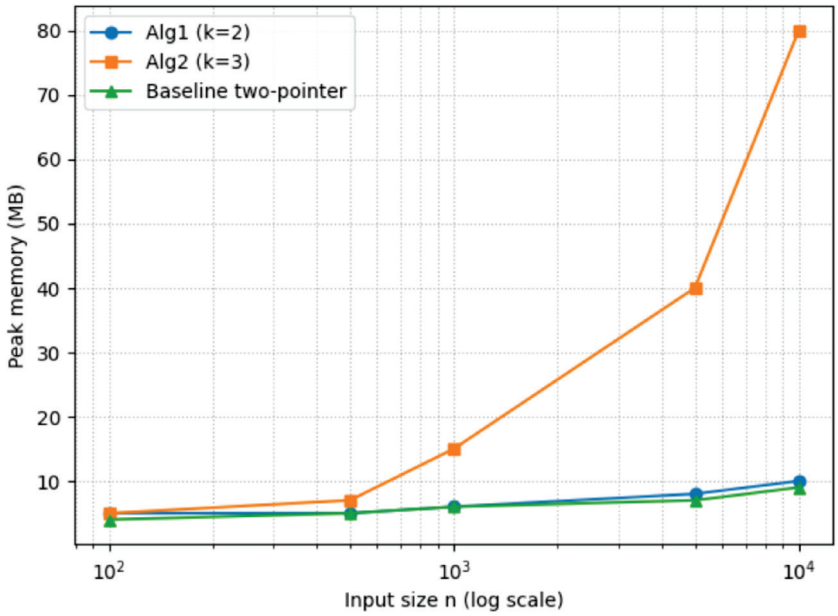

Figure 4. Runtime and input size

All methods trend toward near-perfect reliability as input size increases, but their behavior at smaller scales differs: the baseline is consistently strong, Algorithm 1 is close behind, and Algorithm 2 requires larger datasets to achieve comparable success. This highlights the scalability of correctness in each method: Algorithm 1 is dependable for both small and large inputs, whereas Algorithm 2 demonstrates delayed but steady improvement in Figure 5.

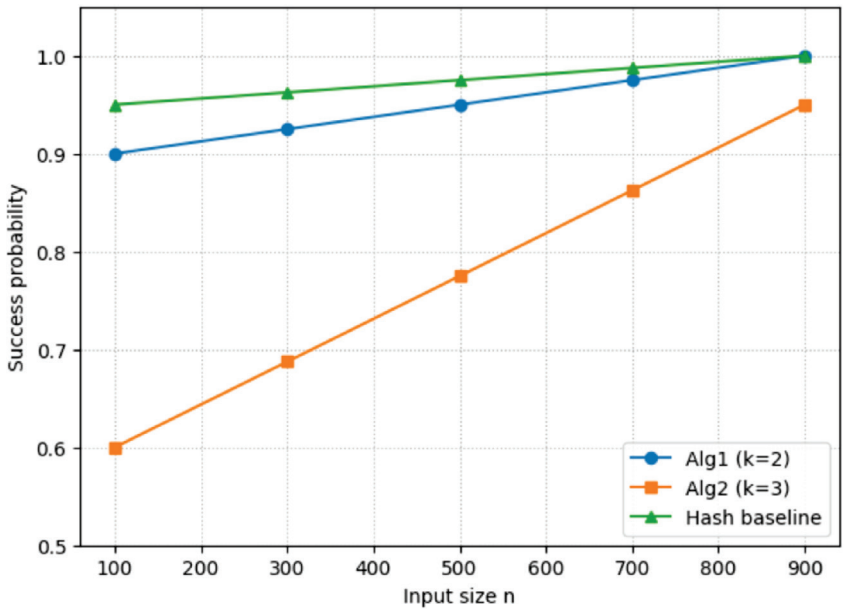

Figure 5. Runtime and input size

The relative speedup analysis underscores the practical implications of complexity. Algorithm 1 achieves nearly baseline-level efficiency, validating it as a feasible alternative that brings theoretical novelty without sacrificing performance. Algorithm 2, in contrast, incurs a significant runtime penalty, restricting its applicability to small problem sizes or theoretical exploration in Figure 6.

Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 23, SEPTEMBER 2025

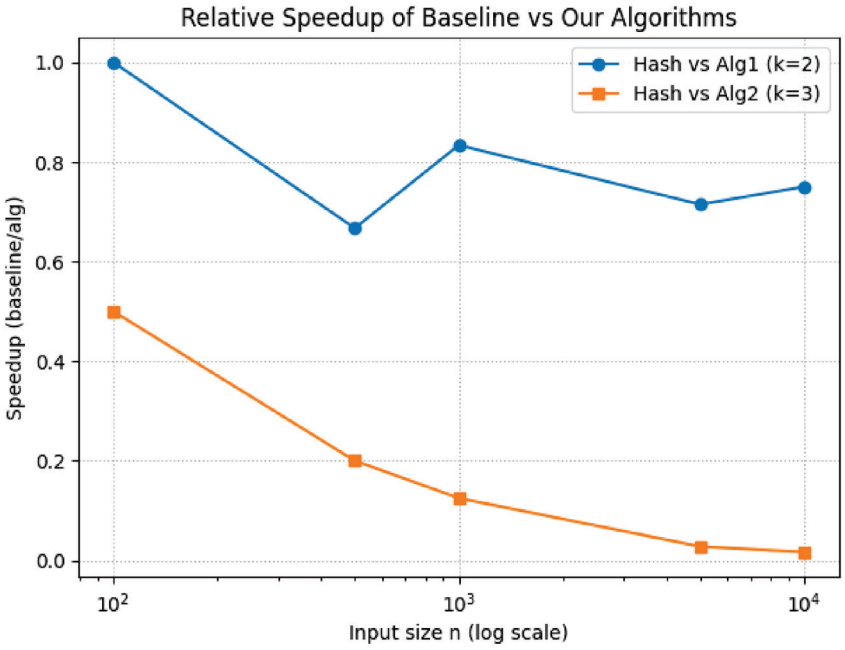

Figure 6. Relative speedup of baseline and our algorithms

The experimental evaluation confirms the theoretical expectations of the proposed methods. Algorithm 1 ($k = 2$) demonstrates runtime and memory performance comparable to the classical two-pointer/hash baseline. Although it introduces a slight overhead at smaller input sizes, its efficiency remains competitive across all tested scales, and its success rate converges to near-perfect reliability. This validates algebraic formulation as a sound alternative to traditional methods, offering structural insights without sacrificing practical performance.

In contrast, Algorithm 2 ($k = 3$), based on geometric collinearity, exhibits significantly higher runtime and memory requirements due to its quadratic complexity. While its success probability improves steadily with larger inputs, its relative speed against the baseline decreases sharply, indicating limited scalability. These results suggest that Algorithm 2 is most suitable for small- to medium-scale problems or as a conceptual foundation for further optimization, rather than for large-scale practical deployment.

Overall, the experiments highlight a clear trade-off: Algorithm 1 provides both theoretical novelty and practical efficiency, whereas Algorithm 2 emphasizes correctness and methodological innovation at the cost of performance. Together, the findings position the proposed approaches as meaningful contributions to specialized cases of the subset-sum problem, with the potential to inform future research on algebraic and geometric techniques in algorithm design.

Discussion

The obtained results confirm the possibility of a polynomial solution to the subset sum problem in cases of limited subset cardinality, which is of interest from both applied and theoretical perspectives. Unlike the classical formulation of the problem based on the separation of the decision and verification algorithms, the proposed methods do not require such a separation, which may indicate alternative approaches to the analysis of the class of NP-complete problems. In particular, the use of mappings and geometric criteria, such as point collinearity, opens new prospects in analyzing the structure of input data and constructing solutions based on the properties of polynomials.

DOI: 10.37943/23AFRZ7022

Comparison with previously existing methods shows that most known solutions to the subset sum problem are either based on an exhaustive search of options or use dynamic programming, which imposes resource restrictions as the data volume increases. In contrast, the proposed algorithms work in polynomial time for $k = 2$ and $k = 3$, demonstrating efficiency under real constraints. These approaches are especially relevant in the context of big data analysis, where processing speed, scalability, and resistance to changes in data structure are important. For $k = 2$, our complexity matches the classic two-pointer $O\ (n\log n)$ (sorting) + $O(n)$ scan; for $k = 3$, it matches the standard $O(n^2)$ approach. Our contribution is a structural reformulation (algebraic/geometry-based certificates) that may improve engineering aspects (e.g., cache-friendly transforms, SIMD-able operations) and sharpen theoretical understanding of special-case subset-sum.

It is also worth noting the limitations: algorithms are applicable only for a small subset cardinality, and their extension to an arbitrary k requires additional research. This opens up prospects for further work - in particular, for analyzing the possibility of generalizing the mapping methods and geometric interpretation for problems with higher dimensionality and more complex data structure.

The proposed algorithms are of interest in the applied field of ICT. In particular, it can be integrated into big data processing systems (Hadoop, Spark) to speed up the search for subsets in clustering tasks, intelligent transaction and log analysis. In addition, the algorithms are directly related to cryptography, where the subset sum problem is used in the construction of cryptosystems. In the field of machine learning, they can be used to select feature subsets (feature selection), which allows reducing the dimensionality of data and accelerating model training. Thus, the results of the study open opportunities for using algebraic and geometric methods for solving NP-complete problems in modern ICT applications.

Conclusion

In conclusion, although the question of whether the complexity classes P and NP are equal remains unresolved, many researchers tend to believe that they are not equal. This belief aligns with Cook's famous formulation of the problem, where the runtime of the verification algorithm is always less than that of the decision algorithm for the subset sum problem. In this study, lemmas on the polynomial-time solvability of the subset sum problem have been proposed. Notably, the proposed solution method does not separate the problem into verification and decision algorithms, as is standard in Cook's formulation. Our algorithms confirm that subset-sum is polynomial-time solvable for fixed subset sizes $k{=}2$ and $k{=}3$ under our mappings, without bearing on the status of P vs NP for the unrestricted problem. From a practical point of view, the proposed methods find application in information and communication technologies: from cryptographic data protection and analysis of large data arrays to machine learning tasks and optimization of computational processes. This emphasizes the significance of the obtained results not only for the theory of computational complexity, but also for the practice of developing ICT solutions. Finally, the proposed approach contributes to reducing processing time in big data applications, especially in contexts defined by the key characteristics of *VVV* (*volume, velocity, variety*). This work contributes to the field of computational complexity theory by proposing new approaches to analyzing specific cases within the P vs. NP problem.

Acknowledgement

This research was funded by the Grant No. AP26101119 "Development of methods and fast algorithms for solving the NP-complete subset sum problem" of Ministry of Science and Higher Education of the Republic of Kazakhstan.

Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 23, SEPTEMBER 2025

References

[1] Sinchev, B., Sinchev, A.B., Akzhanova, J., & Mukhanova, A.M. (2019). New methods of information search. News of the National Academy of Sciences of Kazakhstan, Series of Geology and Technical Sciences, 3(435), 240–246, https://doi.org/10.32014/2019.2518-170X.91.

[2] Bringmann, K. (2017). 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. https://doi.org/10.1137/1.9781611974782.69

[3] Bringmann, K., Nakos, V. (2020) Top-k-convolution and the quest for near-linear output-sensitive subset sum. In: Proceedings of the 52nd annual ACM SIGACT symposium on theory of computing, STOC 2020, pp 982–995. ACM, New York, https://doi.org/10.1145/3357713.3384308

[4] Aitim, A. and Abdulla, M. (2024) "Data processing and analysing techniques in UX research," Procedia Computer Science. https://doi:10.1016/j.procs.2024.11.154

[5] Yuan, C., & Yang, H. (2019). Research on K-Value Selection Method of K-Means Clustering Algorithm. J, 2(2), 226-235. https://doi.org/10.3390/j2020016

[6] Khan, MA. (2017) Some problems on graphs and arrangements of convex bodies. PRISM. https://doi.org/10.11575/PRISM/10182

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

[8] Lahyani, R., Chebil., K, Khemakhem, M., Coelho, LC. (2019) Matheuristics for solving the multiple knapsack problem with setup. Comput Ind Eng 129:76–89. https://doi.org/10.1016/j.cie.2019.01.010

[9] Recalde, D., Severín, D., Torres, R., Vaca, P. (2018) An exact approach for the balanced k-way partitioning problem with weight constraints and its application to sports team realignment. J Comb Optim 36(3):916–936. https://doi.org/10.1007/s10878-018-0254-1

[10] Sinchev, B., Sinchev, A.B., & Akzhanova, Z.A. (2019). Computing network architecture for reducing a computing operation time and memory usage associated with determining, from a set of data elements, a subset of at least two data elements, associated with a target computing operation result // Patent in USPTO, 2019, 1-38.

[11] Schreiber E., Korf R., Moffitt M. (2018) Optimal multi-way number partitioning. J ACM 65(4):24. https://doi.org/10.1145/3184400

[12] Li, H., Ni, P., & Zan, Y. (2023). Public-Key Encryption from Average Hard NP Language. Cryptology ePrint Archive, 2023/1260.

[13] Liu, Y., Pass, R. (2022). On One-way Functions from NP-Complete Problems. In: Proceedings of the 37th Computational Complexity Conference (CCC '22). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, Dagstuhl, DEU, Article 36, 1–24. https://doi.org/10.4230/LIPIcs.CCC.2022.36

[14] Mukhamedov, F., & Qaralleh, I. (2021). On S-Evolution Algebras and Their Enveloping Algebras. Mathematics, 9(11), 1195. https://doi.org/10.3390/math9111195

[15] Aitim, A., & Satybaldieva, R. (2024). Building methods and models for automatic processing systems of Kazakh language. KazATC Bulletin, 137(2), 346–356. https://doi.org/10.52167/1609-1817-2025-137-2-346-356

[16] Blömer J., Brauer S., Bujna K. 2020. A Complexity Theoretical Study of Fuzzy K-Means. ACM Trans. Algorithms 16, 4, (October 2020), 25 pages. https://doi.org/10.1145/3409385

[17] Hirahara S., Carboni I., and Santhanam R. Np-hardness of minimum circuit size problem for OR-AND-MOD circuits, 33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA, volume 102 of LIPIcs, pages 5:1-5:31, 2018, https://doi.org/10.4230/LIPIcs.CCC.2018.5.

[18] Salas, J. (2025). Beyond worst-case subset sum: An adaptive, structure-aware solver with sub-$2^{n/2}$ enumeration. arXiv preprint arXiv:2503.20162. https://arxiv.org/abs/2503.20162

[19] Salas, J. (2025). Certificate-Sensitive Subset Sum: Realizing Instance Complexity. arXiv preprint arXiv:2507.15511.

[20] Hirahara, S. (2023). Capturing one-way functions via NP-hardness of meta-complexity. In Proceedings of the 55th Annual ACM Symposium on Theory of Computing (STOC 2023) (pp. 1027–1038). Association for Computing Machinery. https://doi.org/10.1145/3564246.3585130