УДК 004.421.2
МРНТИ 27.33.19, 27.35.15

https://doi.org/10.55452/1998-6688-2026-23-1-82-93

[1]*Ауезова А.С.,
магистр, ассистент-профессор, ORCID ID: 0000-0001-9860-4491,
*e-mail: a.auyezova@iitu.edu.kz
[2]Муханова А.М.,
к.т.н., старший преподаватель, ORCID ID: 0000-0001-6781-5501,
e-mail: nuraksulu72@mail.ru
[3]Синчев А.,
магистр, ORCID ID: 0000-0002-7333-2255,
e-mail: askar.sinchev@gmail.com

[1]Международный университет информационных технологий,
г. Алматы, Казахстан
[2]Q-university, г. Алматы, Казахстан
[3]National Information Technologies JSC, г. Астана, Казахстан

АЛГОРИТМЫ NP-ПОЛНЫХ ЗАДАЧ. ЧАСТЬ I

Аннотация

В работе рассматривается параметризованная постановка задачи о сумме подмножеств для n-элементного множества положительных целых чисел, сумма элементов которого равна заданному сертификату S. Предлагается метод формирования подмножеств фиксированной мощности k на основе индексных сертификатов и двумерных массивов, позволяющий эффективно выполнять выборку подмножеств без полного перебора. Основная идея заключается в переходе от значений элементов к индексному пространству и использовании структурированных диагоналей массивов, что снижает практическую трудоемкость вычислений при фиксированном k. Представлены алгоритмы формирования подмножеств четной мощности и показана их применимость в задачах поиска информации и обработки больших данных, а также на задачи поиска неструктурированной информации, где существенную роль играют скорость обработки и рациональное использование памяти. Подчеркивается, что полученные результаты относятся к ограниченной и параметризованной постановке задачи и не противоречат известной NP-полноте общей формулировки.

Ключевые слова: NP-полные задачи, задача о сумме подмножеств, полиномиальные алгоритмы, динамическое программирование, большие данные (Big Data), поиск информации.

Введение

В n-элементном множестве целых чисел X^n найти подмножество, сумма элементов которого равна сертификату S. Эта задача о сумме подмножеств связана с проблемой ранца, которая решена Р. Беллманом [1] в 1956 г. Им предложен псевдополиномиальный алгоритм с временем выполнения T=O(nS) на основе метода динамического программирования.

Спустя 43 года Д. Писинжер [2] улучшил решение этой задачи за время T=O(nS/logS), применяя битовое представление сертификата S.

Недавно Килильяс и Сюй представили в [3] новый псевдополиномиальный алгоритм, основанный на парадигме «разделяй и властвуй», который вычисляет все реализуемые суммы до целого числа $u \geq S$ в оценке времени O(min{ $\sqrt{n}\,u$, $u^{4/3}$, σ) выполнения алгоритма, где σ – сумма всех элементов исходного множества. В их подходе применяется хеширование для ускорения решения задачи о сумме подмножеств в случаях, когда элементы множества лежат в ограниченном числовом интервале. Общая постановка задачи сводится к такому случаю пу-

тем разбиения исходного множества на несколько меньших подмножеств. Авторы утверждают, что предложенный алгоритм является на сегодняшний день самым быстрым общим детерминированным алгоритмом для решения задачи о сумме подмножеств.

В 2017 г. К. Биргман представил псевдополиномиальный рандомизированный алгоритм [4], работающий за время O(n+S). Точнее, учитывая экземпляр задачи о сумме подмножеств (X^n, S), вычисляется множество всех сумм s(X^n; S), генерируемых небольшими подмножествами Y ⊆ X^n размерности | Y | ≤ k с постоянной вероятностью принадлежности сертификата S множеству сумм s(X^n; t) и малой ошибкой. В дальнейшем данный параметр k используется при определении трудоемкости алгоритма.

Важно отметить, что детальный обзор современных результатов, содержащихся более чем в 60 научных трудах, по псевдополиномиальным алгоритмам решения задачи о сумме подмножеств приведен в работах [3, 4].

Наряду с псевдополиномиальными алгоритмами имеются точные алгоритмы с экспоненциальным временем работы, когда в n-элементном множестве X^n существует хотя бы одно подмножество, сумма элементов которого равна S. В первую очередь, к ним относятся работы Горовиц, Санни [5] с временем выполнения алгоритма $T = O(2^{n/2})$ и требуемой памятью $\$ = O(2^{n/2})$ и Шреппеля, Шамира [6] с временем выполнения алгоритма $T = O(2^{n/2})$ и требуемой памятью $\$ = O\left(2^{\frac{n}{4}}\right)$.

В работах [7, 8, 9] введены понятия полиномиального (практического) алгоритма и его трудоемкости. Трудоемкость алгоритма напрямую используется при определении времени работы алгоритмов, разработанных на основе известного метода перебора, табличных методов (табличная к-сумма) и методов поискового запроса неструктурированной информации по многим ключевым словам [10, 11] и методов обработки и анализа больших данных [12].

Важно подчеркнуть, что анализ трудоемкости алгоритмов решения задачи о сумме подмножеств является фундаментальной проблемой теоретической информатики. Данная задача широко используется в качестве стандартной модели при изучении алгоритмов, допускающих полиномиальное время решения, в различных областях информационных технологий. В рамках теоретической информатики на ее основе формулируется ряд типовых практических задач, которые рассматриваются ниже [13–15] и другие.

Задача 1а. Имеется таблица 2xn и заданное число S. Необходимо найти два числа из разных строчек (по одному из каждой строки, k=2), дающих в сумме S.

Задача 1б. Даны множество из n чисел и число S. Требуется выяснить, существует одно или несколько подмножеств из двух чисел (k=2), сумма элементов которых равна S.

Алгоритм А. Полный перебор. Время работы $-O(n^2)$.

Алгоритм Б. Перебор с сортировкой. Отсортировать первую строку, для каждого элемента из второй строки вычесть его из S и искать эту разность в первой строке. Время работы $-O(nlogn)$. Требование к памяти $O(n)$.

Эти алгоритмы преподаются в течение последних нескольких десятилетий для студентов и IT-специалистов, и, таким образом, они оказали большое влияние на теоретическую информатику.

Важнейшая задача теоретической информатики о равенстве классов P и NP была сформулирована в 1971 г. и до сих пор остается нерешенной. В настоящее время доказана полнота более 3000 задач из класса NP.

К открытым проблемам вышеуказанных простых задач относятся нахождение времени выполнения алгоритмов T< $O(nlogn)$ и T< $O(n^2logn)$. Однако так называемые стандартные задачи не решены до настоящего времени. Вообще, возникает вопрос существования решения простых задач за меньшее время и выборки подмножества $X^k, (k = 2m < \frac{n}{2})$ за время T ≤ $O(n^2)$ и требуемое пространство $\$ \leq O(n^2)$.

Поэтому исследование каждой проблемы из класса NP-complete является актуальным.

Наши вклады

В n-элементном множестве X^n положительных целых чисел найти k-элементное подмножество X^k ($k < n$), сумма элементов которого равна сертификату S^k. Мощность k определяет трудоемкость предлагаемых алгоритмов решения задачи о сумме подмножества. В полученных патентах USPTO приведена компьютерная система сверхбыстрой обработки больших данных с объемом $n < +\infty$. Для мощности k=2 скоростью обработки, пропорциональной времени выполнения T $= O(n)$ с требуемой памятью $\mathbb{S} = O(n^2)$. Предложенный подход основан на отображении $y = \tau(S^k, x) = (S^k - x)x, \forall x \in X^n$, аргументами которого являются сертификат S^k и элементы x множества X^n , теореме Ферма и методе слияния. Предыдущие лучшие полиномиальные алгоритмы имеют характеристики T $\leq O(nlogn), M \leq O(n)$ либо T $\leq O(n^2 logn), \mathbb{S} \leq O(n^2)$.

Применения полученных результатов

В работе предложены новые методы решения задачи о сумме подмножеств и соответствующие алгоритмы, ориентированные на задачи обработки и анализа больших данных и поиска неструктурированной информации. Методы применимы к поисковым запросам с двумя и более ключевыми словами и отличаются сниженной трудоёмкостью и рациональным использованием памяти. Время выполнения и требуемая память для таких запросов пропорциональны $O(n^2)$. Использование индексного представления данных обеспечивает повышение скорости обработки и анализа информации в реальном времени в задачах, характеризуемых признаками Big Data (volume, velocity, variety).

Методология

В основе предлагаемого подхода лежит параметризованная постановка задачи о сумме подмножеств, в которой фиксируется мощность искомого подмножества. Такой подход позволяет исследовать структурные свойства задачи без рассмотрения полного пространства всех возможных сочетаний.

Метод основан на введении индексного сертификата, представляющего сумму индексов элементов подмножества. Для фиксированной мощности подмножества формируется двумерный массив, элементы которого соответствуют возможным комбинациям индексов. Каждому значению индексного сертификата соответствует диагональ этого массива, что позволяет свести поиск подмножеств к анализу ограниченного числа элементов.

Формирование подмножеств большей мощности осуществляется путем объединения подмножеств меньшей мощности при условии непересечения их индексов. Такой механизм слияния обеспечивает структурированный перебор и позволяет контролировать вычислительную сложность на каждом этапе.

Следует подчеркнуть, что предложенные алгоритмы ориентированы на параметризованную форму задачи и практические приложения, такие как обработка больших данных и поиск информации. Полиномиальная трудоемкость достигается за счет фиксированной мощности подмножеств и использования структурированных массивов индексов, что не противоречит известной NP-полноте общей постановки задачи.

Материалы и методы

Постановка NP-полной задачи (задачи о сумме подмножества (ЗСП)) и методы их решения

Введем основные определения и обозначения.

Определение1. Трудоемкостью алгоритма называется функция

$f(n) = O(g(n)) \leftrightarrow \exists(C > 0), n_0 : \forall(n > n_0) f(n) \leq Cg(n)$. Функция $f(n)$ асимптотически ограничена сверху функцией $g(n)$ с точностью до множителя С.

Определение 2. Алгоритм называется полиномиальным, если для трудоемкости функции $f(n)$ найдется такое $k \in N$, что $f(n) = O(n^k)$, для некоторой константы k, не зависящей от длины входных данных n.

Тогда формальная постановка задачи о сумме подмножеств в параметризованной форме имеет вид:

$$S: \exists \subseteq X^n, \sum_{x_i \in X^k} x_i = S \quad (1)$$

Подмножества X^k выбираются на базе функции сочетания

$$C_n^k = \frac{n!}{k!(n-k)!} = \frac{n(n-1)(n-2)\ldots(n-k+1)}{k!} \quad (2)$$

На основе свойств функции сочетания (2) найдем дискретный диапазон

$$[S_{min}^k, S_{max}^k] \quad (3)$$

где $S_{min}^k = \sum_{i=1}^k x_i$, $S_{max}^k = \sum_{i=n-k+1}^n x_i$, $x_i \in X^n$. Здесь верхний индекс всех переменных и других величин связан с мощностью подмножества X^k.

Введем отсортированное (не обязательное условие) множество натуральных чисел N^n ={1,2,3,…,n} мощности $n = |N^n|$. Без потерь общности в множество N^n можно включить число ноль, тогда N^n ={0,1,2,3,…,n-1}. Тогда постановка о сумме подмножеств $N^k \subseteq N^n$ мощности $k = |N^k|$ с заданным индексным сертификатом s_k в параметризованной форме имеет вид:

$$s^k: \exists N^k \subseteq N^n, \sum_{n_i \in N^k} n_i = s^k. \quad (4)$$

Вспомогательная задача (4) исключает из вычислительной сложности задачи (1) параметр точности p (определяется как число двоичных разрядов в числах, составляющих исходное множество) и тем самым облегчает решение задачи (1), но и имеет самостоятельный научный интерес. Набор элементов подмножества N^k определяется на основе функции сочетания (2). Каждое подмножество N^kсостоит из k элементов множества N^n.

Поэтому найдем величины $s_{min}^k = \sum_1^k n_i$, $n_i \in N^n$, $s_{max}^k = \sum_{n-k+1}^n n_i$, $n_i \in N^n$. Приведем возможный дискретный диапазон изменения индексного сертификата s^k, соответствующий некоторому подмножеству из набора подмножеств $N^k \subseteq N^n$,

$$s^k \in [s_{min}^k, s_{max}^k] \quad (5)$$

Заметим, что диапазон (5) описывает только уникальные индексные сертификаты s_i^k. Далее найдем величину

$$m^k = s_{max}^k - s_{min}^k + 1 = kn - \frac{(k-1)k}{2} - \frac{k(k+1)}{2} + 1 = kn - k^2 + 1 \quad (6)$$

Формула(6)определяетколичествоуникальныхиндексныхсертификатов$s_i^k, i = 1,2, \ldots, m_k$.

На основе функции сочетания (2) подмножества X^k с мощностью $k = 2$ представляются в виде двумерного треугольного массива порядка (n-1)x(n-1):

$$X^2 = \begin{Bmatrix} x_1 + x_2 \; x_1 + x_3 \ldots\ldots\ldots\ldots\ldots\ldots x_1 + x_{n-1} \; x_1 + x_n \\ x_2 + x_3 \; x_2 + x_4 \ldots\ldots x_2 + x_{n-1} \; x_2 + x_n \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_{n-2} + x_{n-1} \; x_{n-2} + x_n \\ x_{n-1} + x_n \end{Bmatrix}, X^2 = \{X_1^2, X_2^2, \ldots, X_l^2\}, \; l = C_n^2 \quad (7)$$

Здесь каждое подмножество X^2 состоит из двух элементов: $X^2 = \{x_i, x_j\}$.

Алгоритм генерации массива (7): достаточно сложить элемент x_1 с элементами множества X^n (данное множество представляет собой одномерный массив), начиная со второго элемента, получим $(x_1 + x_2) \in X^2$ и до конца- $(x_1 + x_n) \in X^2$; затем сложить элемент x_2 с элементами этого множества, начиная с третьего элемента $(x_2 + x_3) \in X^2$ и до конца-($x_{n-2} + x_{n-1}\, x_{n-2} + x_n) \in X^2$, и так далее, пока не получим последний элемент $(x_{n-1} + x_n) \in X^2$.

Дискретные значения диапазона (4) непосредственно состоят из значений элементов массива (7). Количество элементов в массиве (7) равно $\frac{(n-1)n}{2}$. В частности, $x_{12} = x_1 + x_2, i = 1, j = 2$,…, $x_{ij} = x_{n-1} + x_n, i = n-1, j = n, X^2 = \{x_1, x_2\}, \dots, X^2 = \{x_{n-1}, x_n\}$.

Массив (7) относительно индексов i, j элементов x_{ij} подмножества X^2 имеет вид:

$$N^2 = \begin{Bmatrix} 12\ 13 \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots 1\ n-1\ 1\ n \\ 23\ 24 \dots \dots \dots \dots \dots \dots \dots 2\ n-1\ 2\ n \\ \dots \dots \dots \dots \dots \dots \dots \dots \dots \\ n-2\ n-1\ n-2\ n \\ n-1\ n \end{Bmatrix} \quad (8)$$

Здесь индексы двумерного треугольного массива N^2 выбираются из множества последовательных натуральных чисел $N^n = \{1,2, \dots, n\}$ с мощностью $n = |N^n|$. Отметим, что между массивами (7) и (8) существует взаимно однозначное соответствие. Согласно условию $S^2 \in [z^k_{min}, z^k_{max}]$ леммы имеем, что сертификат S^2 принадлежит дискретному диапазону (4), так как функция (2) генерирует все сочетания, необходимые для формирования всего набора подмножеств X^2. Это означает, что для данной мощности k найдется элемент $x_i + x_j$ из массива (7), равный сертификату $x_{ij} = S^2$, и при этом фиксируются индексы i, j. Тогда для этого элемента выполняется условие $\sum_{x_i \in X^2} x_i = x_{ij} = x_i + x_j = S^2$. Задача о сумме подмножеств (1) решена.

На базе массива (7) введем двумерный треугольный массив индексных сертификатов:

$$S^2 = \begin{Bmatrix} 1+2\ 1+3 \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots 1+(n-1)\ 1+n \\ 2+3\ 2+4 \dots \dots \dots \dots \dots \dots 2+(n-1)\ 2+n \\ \dots \dots \dots \dots \dots \dots \dots \dots \dots \\ n-2+(n-1)\ (n-2)+n \\ (n-1)+n \end{Bmatrix} \quad (9)$$

Из массива (9) выделим уникальные индексные сертификаты:

3, 4,… $1 + (n-1), 1 + n, 2 + n, \dots, (n-1) + n.$

Таким образом, это соотношение включает первую строку и последний столбец массива (9), так как другие элементы по диагонали повторяются.

Выше доказали разрешимость задачи (1) и существование подмножества X^2. Это означает, что существует элемент $x_{ij} = x_i + x_j = S^2$ при найденных значениях индексов i и j, тогда необходимый индексный сертификат $s^2 = i + j$. Последнее дает возможность введения диафантова уравнения для нахождения элементов одной из диагоналей массива (9) либо требуемых элементов подмножества N^2:

$$N^2 : n_i + n_j = s^2, (n_i, n_j) \in N^n \quad (10)$$

Важно отметить, что количество решений диафантова уравнения (10) равно количеству подмножеств N^2 при всех индексных сертификатах s^2. Максимальное количество решений диафантова уравнения (10) будет меньше или равно $n/2$ – наибольшему числу элементов в диагонали массива (8) с индексным сертификатом $s^2 = 1 + n$ и $T \leq O\left(\frac{n}{2}\right)$. Другими словами,

время выборки подмножества X^2, описывающего подмножеством N^2, определяется количеством элементов найденной диагонали массива (9) при заданной величине индексного сертификата s^2.

Введем отображение при k=2

$$y_i = \tau(x_i, S^k) = (S^k - x_i)x_i, x_i \in X^n \quad (11)$$

и проверка условия

$$\tau(x_i, S^k) = \tau(x_j, S^k) \quad (12)$$

Формирование всех подмножеств N_m^2 и X_m^2 осуществляется с использованием индексного сертификата s^k при объединении подмножеств (7) при учете условия (12)

$$N^k = \bigcup_m N_m^2, X^k = \bigcup_m X_m^2, m = 1,2,\dots,m, k = 2m < \frac{n}{2}, \quad (13)$$

где индексы выбираемых подмножеств N_m^2 и X_m^2 не должны совпадать.

Метод решения задачи о сумме подмножеств в краткой математической форме (последовательность операций) можно записать так:

$$X^n \to \tau(x_i, S^k) = \tau(S^k - x_i, S^k) \to i \neq j \to X_i^2 \cap X_j^2 = \emptyset, i, j = 1,2,\dots,m \to X^k = \bigcup_m X_m^2, k = 2m \leq \frac{n}{2} \quad (14)$$

На базе соотношений (14) установим алгоритм решения NP-полной ЗСП. Первоначально установим алгоритм для задачи (4) – задачи о сумме подмножества для натуральных чисел.

Первый алгоритм определения мощности k подмножества N^k.

Шаг 1. Ввод множества N^n, n, s^k

Шаг 2. Сортировка одномерного массива N^n

Шаг 3. Определение границ $s_{\min}^k, s_{max}^k$ диапазона (5)

Шаг 4. Проверка принадлежности s^k диапазону $s_{\min}^m, s_{max}^m]$

Шаг 5. Вывод мощности k.

Второй алгоритм формирования искомого подмножества N^k для четной мощности k.

Шаг 1. Ввод n, k, s^k, N^n

Шаг 2. Формирование двумерного массива (8)

Шаг 3. Выборка подмножеств N_i^2, N_j^2 на основе отображения (11)

Шаг 4. Проверка условия (11) с целью объединения этих множеств и других

Шаг 5. Формирование подмножества $N^k = \bigcup_m N_m^2, k = 2m \leq \frac{n}{2}$, с несовпадающими индексами этих $N_i^2 \cap N_j^2 = \emptyset, i, j = 1,2,\dots,m$

Шаг 6. Вывод искомого подмножества N^k.

В заключение отметим, что подмножество N^k полностью описывает подмножество X^k.

Далее несложно установить алгоритмы решения задачи (1).

Третий алгоритм определения мощности k подмножества X^k.

Шаг 1. Ввод множества X^n, n, S^k

Шаг 2. Сортировка одномерного массива X^n

Шаг 3. Определение границ $S_{\min}^k, S_{max}^{mk}$ диапазона (3)

Шаг 4. Проверка принадлежности S^k диапазону $[S_{\min}^m, S_{max}^m]$

Шаг 5. Вывод мощности k.

Четвертый алгоритм формирования искомого подмножества X^k для четной мощности k.

Шаг 1. Ввод n, k, S^k, X^n

Шаг 2. Формирование двумерного массива (7)

Шаг 3. Выборка подмножеств X_i^2, X_j^2 на основе отображения (11)

Шаг 4. Проверка условия (11) с целью объединения этих множеств и других

Шаг 5. Формирование подмножества $X^k = \bigcup_m X_m^2, k = 2m \leq \frac{n}{2}$, с несовпадающими индексами этих $X_i^2 \cap X_j^2 = \emptyset, i, j = 1,2, \dots, m$

Шаг 6. Вывод искомого подмножества X^k.

Пример. Достоверность полученных теоретических и практических результатов с применением двумерных массивов (7), (8), (9) и диафантова уравнения (10) для множеств $X^8 = \{10,14,17,20,36,38,43,47\}, N^8 = \{1,2, \dots .8\}$ следует из матрицы (7) с элементом $x_{ij} = x_i + x_j = S^2$, матрицы (8) с элементом $n_{ij} = (n_i, n_j)$, матрицы (9) с суммой двух индексов, равной $n_i + n_j = s^2$:

$$X^2 = \begin{Bmatrix} 24\ 27\ 30\ 46\ 48\ 53\ 57 \\ 31\ 34\ 50\ 52\ 57\ 61 \\ 37\ 53\ 55\ 60\ 64 \\ 56\ 58\ 63\ 67 \\ 74\ 79\ 83 \\ 81\ 85 \\ 90 \end{Bmatrix}, N^2 \begin{Bmatrix} 1,2\ 1,3\ 1,4\ 1,5\ 1,6\ 1,7\ 1,8 \\ 2,3\ 2,4\ 2,5\ 2,6\ 2,7\ 2,8 \\ 3,4\ 3,5\ 3,6\ 3,7\ 3,8 \\ 4,5\ 4,6\ 4,7\ 4,8 \\ 5,6\ 5,7\ 5,8 \\ 6,7\ 6,8 \\ 7,8 \end{Bmatrix},$$

$$s^2 = \begin{Bmatrix} 3\ 4\ 5\ 6\ 7\ 8\ 9 \\ 5\ 6\ 7\ 8\ 9\ 10 \\ 7\ 8\ 9\ 10\ 11 \\ 9\ 1011\ 12 \\ 11\ 12\ 13 \\ 13\ 14 \\ 15 \end{Bmatrix}. \quad (15)$$

Действительно на основе матриц (15) облегчается применение предложенного алгоритма для сертификатов $S^8 = 225$ и $S^6 = 169$, и в том числе других. Тогда сумма индексов множества N^8 равна $s^8 = \frac{n(n+1)}{2} = 36$ и решениями диафантова уравнения (10) с индексным сертификатом 36/4=9 являются подмножества $N_m^2 = \{1,8\} \vee \{2,7\} \vee \{3,6\} \vee \{4,5\}$. Тогда их комбинация имеет вид: $N^8 = \bigcup_{m=1}^4 N_m^2$, согласно которому решение задачи (1) равно $X^8 = \bigcup_{m=1}^4 X_m^2$ и из первой матрицы матриц (15) при последовательном просмотре имеем $S^8 = 57 + 57 + 55 + 56 = 225$. Для S^6 с использованием соотношений (13) $X^6 = \bigcup_{m=1}^3 X_m^2 = 225 - 56 = 169$. Здесь $T \leq O(m^k) = O(kn - k^2 + 1) = O(13), T \leq O(n^2) = O(64), \$ \leq O\left(\frac{(n-1)*n}{2}\right) = 28.$

Пусть сертификат $S^4 = 105$, которому соответствует индексный сертификат $s^4 = 16$, $N_m^2 = \{1,7\} \vee \{2,6\} \vee \{3,5\}$, ответ $\{1,7\} \vee \{2,6\}$ либо $\{2,6\} \vee \{3,5\}$. Здесь $T \leq O(m^k) = O(kn - k^2 + 1) = O(17), T \leq O(n^2) = O(64), \$ \leq O\left(\frac{(n-1)*n}{2}\right) = 28.$

Возможны другие комбинации элементов подмножеств N^2.

Методы уменьшения трудоемкости предложенных алгоритмов и их времени работы. Уменьшение времени выполнения предлагаемых простых алгоритмов решения задачи о сумме подмножеств определяется отношением $\frac{C_n^k}{C_n^2}$. В силу простоты алгоритмов имеется возможность дальнейшего уменьшения трудоемкости и времени работы алгоритма.

Метод расщепления множества на подмножества. Расщепление множества X^n на d подмножеств с размерностью $d = |X^n|/|X_i^d|$, причем $d \geq k, i = (1,2, \dots, d)$. Если размерности подмножеств больше k, то первоначально ищутся два подмножества $\{X_i^m\}, \{X_j^m\}$ с размерностью $m = |X_i^m| = |X_j^m|$ при k=2m из каждого подмножества X^d, $(i,j) \in (1,2, \dots, d), i \neq j$ и формируется подмножество $X^k = \{X_i^m\} \cup \{X_j^m\}$ с несовпадающими индексами. В слу-

чае пустоты подмножества X^k для каждого подмножества X_i^d рассматриваются парное объединение двух различных подмножеств X_i^d, X_j^d для формирования подмножества $X^k = \{X_i^m\} \cup \{X_j^m\}, X_i^m \subseteq X_i^d, X_j^m \subseteq X_j^d$ и более. Индексы элементов этих подмножеств $\{X_i^m\}, \{X_j^m\}$ заведомо не пересекаются. Трудоемкость алгоритмов снижается скачкообразно.

Метод распараллеливания операций. Распараллеливание необходимых операций возможно при применении сверки Вандермонда, порождаемых операторами программного продукта для простых алгоритмов решения задачи о сумме подмножеств с целью использования всех возможностей вычислительных устройств и аппаратных средств, снижает трудоемкость алгоритмов.

Результаты и обсуждение

Разработанные алгоритмы позволили сформировать новые методы решения задачи о сумме подмножеств, относящейся к классу NP-полных задач. На основе предложенного отображения, методов слияния и применения теоремы Ферма была построена система полиномиальных алгоритмов, которые обеспечивают значительное снижение трудоёмкости по сравнению с известными экспоненциальными и псевдополиномиальными методами.

Полученные результаты подтверждаются как теоретическими выкладками, так и практическими примерами. В частности, показано, что при мощности подмножеств k=2 вычислительная сложность сводится к пропорциональности времени выполнения T и памяти, что сопоставимо с требованиями к современным системам обработки больших данных. Использование двумерных массивов и индексных сертификатов позволило обеспечить быстрый поиск и выборку подмножеств, что открывает возможность применения метода в реальных задачах анализа данных.

Сравнение с алгоритмами Беллмана, Писингера, Килиляса–Сюя и Биргмана показывает, что предложенный метод требует меньше памяти и проще реализуется при больших входных данных. В отличие от переборных подходов, трудоёмкость определяется размером диагоналей массивов, что обеспечивает ускорение выборки.

Ограничением метода является рост диапазона сертификатов при высокой плотности данных, что может увеличивать время выборки. Тем не менее, алгоритмы эффективно применимы в задачах анализа больших данных и поиска информации, где критичны скорость и рациональное использование памяти.

Заключение

Сделаем вывод, хотя вопрос о равенстве классов P и NP до сих пор не решен, многие ученые склонны считать, что они не равны. Это утверждение справедливо для поставленной Куком знаменитой задачи, в которой время работы проверочного алгоритма всегда меньше времени работы решающего алгоритма для задачи о сумме подмножеств. Но окончательную точку в споре поставит лишь строгое математическое доказательство. Особо отметим, что предлагаемый общий метод решения задачи о сумме подмножеств не разделяет на проверочные и решающие алгоритмы, которые имеются в самой постановке задачи Кука, а предлагается семейство полиномиальных алгоритмов решения задачи о сумме подмножеств, основанное на предложенном отображении и двумерных массивах, аргументами которых являются сертификат и входные данные. При этом предложен простой механизм работы с индексами элементов исходного множества и для выборки необходимых подмножеств. В основе интеллектуальной системы лежат задачи о сумме подмножеств из n-мерного множества, которые

относятся к NP-полным задачам. Множество представляет собой целые числа, натуральные числа, простые числа, числа Фибоначчи и числа, обладающие другими свойствами. Здесь n непосредственно связан с признаком volume BIG DATA. У экспоненциального алгоритма размерность исходного множества возрастает до $2^{n/2}$. Алгоритмы выделяют k-мерные подмножества из n-мерного множества. Время работы алгоритмов и требуемая память находятся на основе функции сочетания C_n^k, причем сочетание C_n^k намного меньше, чем сочетания C_n^m $(k < m, C_n^k \ll C_n^m)$. Уменьшение времени определяется отношением C_n^m / C_n^k, и тем самым увеличивается скорость (velocity) обработки больших данных (BIG DATA). При этом предложен изящный аппарат работы индексами обеспечивает простоту реализации и расширяет область практического применения предложенного подхода в задачах анализа данных и поиска информации.

Благодарность

Данное исследование финансировалось Комитетом науки Министерства науки и высшего образования Республики Казахстан в рамках научного проекта «Разработка методов и быстрых алгоритмов разрешимости NP-complete задачи о сумме подмножеств» (ИРН AP26101119).

ЛИТЕРАТУРА

1 Bellman R. Notes on the theory of dynamic programming IV – maximization over discrete sets // Naval Research Logistics Quarterly. – 1956. – Vol. 3, № 1–2. – P. 67–70.

2 Pisinger D. Linear time algorithms for knapsack problems with bounded weights // Journal of Algorithms. – 1999. – Vol. 33, № 1. – P. 1–14.

3 Koiliaris K., Xu C. A faster pseudopolynomial time algorithm for subset sum // To appear in SODA '17. – 2017. – 18 p. – URL: https://arxiv.org/abs/1610.04712v2.

4 Bringmann K. A near-linear pseudopolynomial time algorithm for subset sum // To appear in SODA'17. – 2017. – 18p. – URL: https://arxiv.org/abs/1610.04712v2.

5 Horowitz E., Sanni S. Computing partitions with application to the knapsack problem // Journal of the ACM (JACM). – 1974. – Vol. 21. – P. 277–292.

6 Schroeppel R., Shamir A. A T=O(2^n/2), S=O(2^n/4) algorithm for certain NP-complete problem // SIAM Journal on Computing. – 1981. – Vol. 10, № 3. – P. 456–464.

7 Cobham A. The intrinsic computational difficulty of functions // Proceedings of the Congress for Logic, Methodology and Philosophy of Science. – North-Holland, 1964. – P. 24–30.

8 Egmonds J. Paths, trees and flowers // Canadian Journal of Mathematics. – 1965. – Vol. 17. – P. 449–467.

9 Николаев А.В. Геометрический подход к задаче о разрезе. – Ярославль: ЯрГУ, 2014. – 68 с.

10 Ivan Rijsbergen C.J. Information Retrieval. – Glasgow: Dept. of Computer Science, University of Glasgow, 1979.

11 Adamansky A. Overview of full-text search methods and algorithms. – Novosibirsk: Novosibirsk State University, 2018. – 26 p.

12 Chen M., Mao S., Zhang Y., Leung V.C.M. Big Data. Related Technologies, Challenges, and Future Prospects. – Springer, 2014. – 100 p.

13 Лифшиц Ю. Точные алгоритмы и открытые проблемы // Современные задачи теоретической информатики. – URL: http://download.yandex.ru/class/...

14 Куликов А. Алгоритмы NP-трудных задач. Лекции. – СПб: Computer Science клуб СПб отделения МИАН, 2009.

15 Sinchev B., Sinchev A.B., Akzhanova J., Mukhanova A.M. New methods of information search. I // News of the National Academy of Sciences of Kazakhstan. Series of Geology and Technical Sciences. – 2019. – Vol. 3, № 435. – P. 240–246.

16 Sinchev B., Sinchev A.B., Akzhanova Z.A. Computing network architecture for reducing computing operation time and memory usage… // Патент USPTO. – 2019. – 38 p.

17 Akl S.G. Adaptive and optimal algorithms for enumerating permutations and combinations // The Computer Journal. – 1987. – Vol. 30. – P. 433–436.

18 Hoare C.A.R. Quicksort // The Computer Journal. – 1962. – Vol. 5. – P. 10–16.

19 Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ. – М.: Вильямс, 2005.

REFERENCES

1 Bellman, R. Notes on the Theory of Dynamic Programming IV – Maximization over Discrete Sets. Naval Research Logistics Quarterly 3 (1–2), 67–70 (1956).

2 Pisinger, D. Linear Time Algorithms for Knapsack Problems with Bounded Weights. Journal of Algorithms 33 (1), 1–14 (1999).

3 Koiliaris, K., and Xu C. A Faster Pseudopolynomial Time Algorithm for Subset Sum. In Proceedings of SODA ’17, 2017. Preprint, arXiv:1610.04712v2.

4 Bringmann, K. A Near-linear Pseudopolynomial Time Algorithm for Subset Sum. In Proceedings of SODA ’17, 2017. Preprint, arXiv:1610.04712v2.

5 Horowitz, E., and Sanni S. Computing Partitions with Application to the Knapsack Problem. Journal of the ACM, 21, 277–292 (1974).

6 Schroeppel, R., and Shamir A. A T=O(2^n/2), S=O(2^n/4) Algorithm for Certain NP-Complete Problems. SIAM Journal on Computing, 10 (3), 456–464 (1981).

7 Cobham, A. The Intrinsic Computational Difficulty of Functions. In Proceedings of the Congress for Logic, Methodology and Philosophy of Science, 24–30. Amsterdam: North-Holland, 1964.

8 Egmonds, J. Paths, Trees and Flowers. Canadian Journal of Mathematics, 17, 449–467 (1965).

9 Nikolaev, A.V. Geometricheskiy podkhod k zadache o razreze. Yaroslavl: Yaroslavl State University, 2014. (in Russian).

10 Ivan Rijsbergen, C.J. Information Retrieval. Glasgow: Department of Computer Science, University of Glasgow, 1979.

11 Adamansky, A. Overview of Full-text Search Methods and Algorithms. Novosibirsk: Novosibirsk State University, 2018.

12 Chen, Min, Shiwen Mao, Yin Zhang, and Victor C.M. Leung. Big Data: Related Technologies, Challenges, and Future Prospects. Springer, 2014.

13 Lifshitz, Y. Tochnye algoritmy i otkrytye problemy. In Sovremennye zadachi teoreticheskoy informatiki. Accessed September 18, 2025. URL: http://download.yandex.ru/class/... (in Russian).

14 Kulikov, A. Algoritmy NP-trudnykh zadach. Lektsii. Saint Petersburg: Computer Science Club of St. Petersburg Department of Steklov Mathematical Institute, 2009. (in Russian).

15 Sinchev, B., Sinchev, A.B., Akzhanova, J., and Mukhanova, A.M. New Methods of Information Search. I. News of the National Academy of Sciences of Kazakhstan. Series of Geology and Technical Sciences, 3 (435), 240–246 (2019).

16 Sinchev, B., Sinchev, A.B., and Akzhanova, Z.A. Computing Network Architecture for Reducing a Computing Operation Time and Memory Usage… USPTO Patent, 2019.

17 Akl, S.G. Adaptive and Optimal Algorithms for Enumerating Permutations and Combinations. The Computer Journa, l 30, 433–436 (1987).

18 Hoare, C.A.R. Quicksort. The Computer Journal, 5, 10–16 (1962).

19 Cormen, Th., Leiserson, Ch., Rivest, R., and Stein, C. Algoritmy: postroenie i analiz. Moscow: Williams, 2005. (in Russian).

[1]*Ауезова А.С.,
магистр, ассистент-профессор, ORCID ID: 0000-0001-9860-4491,
*e-mail: a.auyezova@iitu.edu.kz
[2]Муханова А.М.,
т.ғ.к., аға оқытушы, ORCID ID: 0000-0001-6781-5501,
e-mail: nuraksulu72@mail.ru
[3]Синчев А.,
магистр, ORCID ID: 0000-0002-7333-2255,
e-mail: askar.sinchev@gmail.com

[1]Халықаралық ақпараттық технологиялар университеті, Алматы қ., Қазақстан
[2]Q-university, Алматы қ., Қазақстан
[3]National Information Technologies JSC, Астана қ., Қазақстан

NP-ТОЛЫҚ ЕСЕПТЕР АЛГОРИТМДЕРІ. І-БӨЛІМ

Андатпа

Жұмыста элементтерінің қосындысы берілген S сертификатына тең, оң бүтін сандардан тұратын n-элементті жиын үшін ішкі жиындар қосындысы есебінің параметрленген тұжырымдамасы қарастырылады. Индекстік сертификаттар мен екі өлшемді массивтер негізінде қуаттылығы (элемент саны) k-ға тең ішкі жиындарды құру әдісі ұсынылған. Бұл әдіс толық іріктеусіз (перебор) ішкі жиындарды тиімді таңдауға мүмкіндік береді. Негізгі идея элементтердің мәндерінен индекстік кеңістікке өтуге және массивтердің құрылымдалған диагональдарын пайдалануға негізделген. Бұл k мәні бекітілген жағдайда есептеулердің практикалық күрделілігін азайтады. Жұмыста қуаттылығы жұп болатын ішкі жиындарды құру алгоритмдері ұсынылып, олардың ақпаратты іздеу, үлкен деректерді өңдеу, сондай-ақ өңдеу жылдамдығы мен жадты ұтымды пайдалану маңызды рөл атқаратын құрылымдалмаған ақпаратты іздеу есептеріндегі тиімділігі көрсетілген. Алынған нәтижелер есептің шектеулі әрі параметрленген нұсқасына ғана қатысты екені және жалпы тұжырымдамадағы белгілі NP-толықтық теориясына қайшы келмейтіні атап өтілген.

Тірек сөздер: NP-толық есептер, ішкіжиындар қосындысының есебі, полиномиалдық алгоритмдер, динамикалық бағдарламалау, үлкен деректер (Big Data), ақпарат іздеу.

[1]*Auyezova A.S.,
MSc., Assistant Professor,
ORCID ID: 0000-0001-9860-4491,
*e-mail: a.auyezova@iitu.edu.kz
[2]Mukhanova A.M.,
Cand.Tech. Sc., Senior Lecturer,
ORCID ID: 0000-0001-6781-5501,
e-mail: nuraksulu72@mail.ru
[3]Sinchev A.,
MSc., ORCID ID: 0000-0002-7333-2255,
e-mail: askar.sinchev@gmail.com

[1]International Information Technology University, Almaty, Kazakhstan
[2]Q-university, Almaty, Kazakhstan
[3]National Information Technologies JSC, Astana, Kazakhstan

ALGORITHMS OF NP-COMPLETE PROBLEMS. PART I

Abstract

The paper considers a parameterized formulation of the subset sum problem for an n-element set of positive integers, where the sum of the selected elements equals a given certificate S. A method for constructing subsets of

fixed cardinality k based on index certificates and two-dimensional arrays is proposed, enabling efficient subset selection without exhaustive enumeration. The core idea is to shift from the value space to the index space and to exploit structured array diagonals, which reduces the practical computational complexity for fixed k. Algorithms for constructing subsets of even cardinality are presented, and their applicability to information retrieval and big data processing tasks is demonstrated, including unstructured information search, where processing speed and efficient memory usage are critical. It is emphasized that the obtained results correspond to a restricted and parameterized formulation of the problem and do not contradict the known NP-completeness of the general subset sum problem.

Keywords: NP-complete problems, subset sum problem, polynomial algorithms, dynamic programming, Big Data, and information retrieval.

Received: September 20, 2025; revised: December 12, 2025; accepted: January 17, 2026.