max planck institut informatik

Improved Pseudopolynomial Time Algorithms for Subset Sum

Karl Bringmann

Simons Institute, Berkeley, December 12, 2016

Subset Sum

Given a set Z of n positive integers and a target t,

is there a subset Y of Z summing to exactly t?

$t = 19$ $\quad Z = \{2,5,6,11,15\}$

$Y = \{2, \quad 6,11 \quad\}$ $\quad \Sigma(Y) = 2 + 6 + 11 = 19$

note that $n \leq t$

well-studied, classic NP-hard problem, $O(2^{n/2})$ algorithm [Horowitz,Sahni'72]

at the core of many other problems: knapsack, constraint shortest path, ...

Subset Sum

Given a set Z of n positive integers and a target t,

is there a subset Y of Z summing to exactly t?

pseudopolynomial time algorithm by dynamic programming: [Bellman'57]

$$T[i,s] \coloneqq T[i-1,s] \vee T[i-1,s-z_i]$$

$$T[0,s] \coloneqq [s=0]$$

$$Z = \{z_1, \dots, z_n\}$$

then $T[n,t]$ decides the Subset Sum instance (Z,t)

time $O(nt)$, space $O(t)$

max planck institut
informatik

Conditional Lower Bounds

Is time $O(nt)$ optimal? Can we prove a conditional lower bound?

any $t^{1-\varepsilon}n^{O(1)}$ algorithm would imply an $2^{(1-\delta)n}(nm)^{O(1)}$ algorithm for Set Cover

[CDLMNOPSW'12]

any combinatorial $t^{1-\varepsilon}n^{O(1)}$ algorithm would imply
a combinatorial $O\left(n^{(1-\delta)k}\right)$ algorithm for k-Clique, for some large constant k

follows from [Abboud,Lewi,Williams'14]

conditional lower bound: $t^{1-o(1)}$

Is there an $\tilde{O}(n+t) = \tilde{O}(t)$ algorithm?

Attempts to break O(nt)

use basic Word RAM parallelism, word size w: $O(nt/w)$ [Pisinger'03]

consider $s := \max Z$; we can assume $s \leq t$: $O(ns)$ [Pisinger'99]

breakthrough: $\tilde{O}(\sqrt{n} \cdot t)$ [Koiliaris,Xu Arxiv'15/SODA'17]

all previous algorithms are deterministic

Is there an $\tilde{O}(t)$ algorithm?

Thm: [B. SODA'17]

Subset Sum is in randomized time $\tilde{O}(t)$.

one-sided error probability $1/n$, time $O(t \log t \log^5 n)$

Preliminaries

Sumset Computation

A, B sets of non-negative integers

sumset: $A \oplus B := \{ a + b \mid a \in A \cup \{0\},\ b \in B \cup \{0\} \}$

***t*-capped sumset:** $A \oplus_t B := (A \oplus B) \cap \{0, \dots, t\}$

Fact: $A \oplus_t B$ can be computed in time $O(t \log t)$

(on Word RAM with word size $\Omega(\log t)$)

1) reduce to **Boolean convolution**: $z_i = \bigvee_j x_j \wedge y_{i-j}$ for vectors $x, y \in \{0,1\}^t$

Boolean conv. of characteristic vectors of A, B yields characteristic vector of $A \oplus B$

capping at t yields $A \oplus_t B$

2) reduce Boolean convolution to **multiplying polynomials** of degree t

set $a := \sum_i x_i \cdot X^i$ and $b := \sum_i y_i \cdot X^i$, and consider their product $c = \sum_i c_i \cdot X^i$

then we can infer z as $z_i = [c_i > 0]$

3) polynomial multiplication is in time $O(t \log t)$ on the Word RAM (using **FFT**)

From Multisets to Sets

Given a set Z of n positive integers and a target t,

is there a subset Y of Z summing to exactly t?

in general Z can be a **multiset**, i.e., every integer z has some multiplicity in Z

reducing to multiplicities ≤ 2: $2k+2$ [Lawler'79]

if $z \in Z$ has multiplicity $2k+1$ 2

then decrese the multiplicity of z to 1 and increase the multiplicity of $2z$ by k

this generates the same subset sums

do this for all $z \in Z$ in increasing order

this is a linear time preprocessing that reduces all multiplicities to ≤ 2

From Multisets to Sets

Given a set Z of n positive integers and a target t,

is there a subset Y of Z summing to exactly t?

in general Z can be a **multiset**, i.e., every integer z has some multiplicity in Z

linear time preprocessing that reduces to **multiplicities** $\leq \mathbf{2}$

reducing to sets: [Koiliaris,Xu'17]

split Z into two **sets** Z_1, Z_2 s.t. $Z = Z_1 \cup Z_2$

new goal is to compute: $S(Z,t) = \left\{ \sum_{y \in Y} y \mid Y \subseteq Z \right\} \cap \{0, \ldots, t\}$

compute $S(Z_1,t)$ and $S(Z_2,t)$ and combine to $S(Z,t) = S(Z_1,t) \oplus_t S(Z_2,t)$

we reduced subset sum on ***multisets*** *to computing $S(Z,t)$ on* ***sets*** *Z*

thus in the remainder: **assume that Z is a set**

Warmup: Unbounded Subset Sum

a variant of Subset Sum:

given a set Z of n positive integers and target t

is there any **sequence** of elements of Z summing to t?

i.e., each item can be picked multiple times

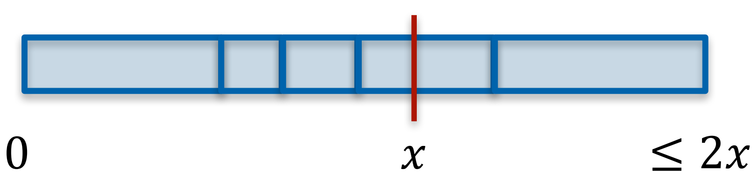
want to compute $S^{unb}(Z,t) = \{ a_1 + \cdots + a_k \mid k \geq 0, a_1, \ldots, a_k \in Z \} \cap \{0, \ldots, t\}$

compute $S^{unb}(Z,2x)$ from $S^{unb}(Z,x)$ via

$$S^{unb}(Z,2x) = S^{unb}(Z,x) \oplus_{2x} S^{unb}(Z,x) \oplus_{2x} Z$$

time $O(x \log x)$

proof: each sequence summing to $\leq 2x$ can be split into two sequences summing to $\leq x$ and at most one additional element



total time $O(t \log t)$

Using Sumset Computation for Subset Sum

notation: $A \oplus_t B := (\{a + b \mid a \in A \cup \{0\},\ b \in B \cup \{0\}\}) \cap \{0, \dots, t\}$ .. sumset

$\Sigma(Y) := \sum_{y \in Y} y$ .. sum of a set

goal is to compute $S(Z, t) = \{\, \Sigma(Y) \mid Y \subseteq Z \,\} \cap \{0, \dots, t\}$

---

how to use „$\oplus_t$": $Z \oplus_t Z$ contains forbidden sums $z + z$ ☹

however, for a **partitioning** $Z = Z_1 \cup Z_2$:

$Z_1 \oplus_t Z_2$ contains only valid subset sums of Z

more generally, if S_1 and S_2 contain only valid subset sums of Z_1 and Z_2, then $S_1 \oplus_t S_2$ contains only valid subset sums of Z

max planck institut informatik

Algorithm for Subset Sum

1. Color-Coding:

computes all $\Sigma(Y)$ for $Y \subseteq Z$ with $|Y| \leq k$

2. Two-level Partitioning:

computes all $\Sigma(Y)$ for $Y \subseteq Z$, assuming $Z \subseteq [t/m, 2t/m]$

3. FasterSubsetSum:

computes all $\Sigma(Y)$ for $Y \subseteq Z$

Color-Coding

.. is a technique from FPT algorithms [Alon,Yuster,Zwick'95]

we use color-coding to detect sums of **small** subsets:

we want to compute all $\Sigma(Y)$ for $Y \subseteq Z$ with $|Y| \leq k$

consider a **random** partitioning $Z = Z_1 \cup \cdots \cup Z_{k^2}$

compute $S := Z_1 \oplus_t \ldots \oplus_t Z_{k^2}$

then S contains only valid sums of Z

we say that the partitioning *splits* Y if $|Y \cap Z_i| \leq 1$ for all i

if the partitioning splits $\boldsymbol{Y}$ then $\boldsymbol{S}$ contains $\boldsymbol{\Sigma(Y)}$

since we can choose the element in $Y \cap Z_i$ (or 0) in each Z_i to obtain $\Sigma(Y)$

Color-Coding

.. is a technique from FPT algorithms [Alon,Yuster,Zwick'95]

we use color-coding to detect sums of **small** subsets:

we want to compute all $\Sigma(Y)$ for $Y \subseteq Z$ with $|Y| \leq k$

consider a **random** partitioning $Z = Z_1 \cup \cdots \cup Z_{k^2}$

compute $S := Z_1 \oplus_t \dots \oplus_t Z_{k^2}$

$\Pr$[random partitioning splits Y] = ?

balls into bins process: at most k balls (Y) into k^2 bins ($Z_1, \dots, Z_{k^2}$)

„birthday paradox":

$\Pr$[no bin is used twice] = $\frac{k^2-1}{k^2} \cdot \frac{k^2-2}{k^2} \cdot \dots \cdot \frac{k^2-(|Y|-1)}{k^2} \geq \left(1 - \frac{1}{k}\right)^{k-1} \geq 1/e$

→ constant success probability

Color-Coding

complete color-coding algorithm:

ColorCoding(Z, t, k):

for $r = 1, \dots, O(\log n)$:

consider a **random** partitioning $Z = Z_1 \cup \dots \cup Z_{k^2}$

compute $S_r := Z_1 \oplus_t \dots \oplus_t Z_{k^2}$

return $\bigcup_r S_r$

this returns a set S containing only valid subset sums of Z

for any $Y \subseteq Z$ with $\Sigma(Y) \leq t$ and $|Y| \leq k$:

$\Sigma(Y) \in S$ with high probability ($\geq 1 - n^{-\Omega(1)}$)

running time: $O(t \log t \cdot k^2 \cdot \log n) = \tilde{O}(t \cdot k^2)$

Two-Level Partitioning

for computing all subset sums, we use a two-level partitioning approach

assume: $Z \subseteq [t/m, 2t/m]$ for some m

fix $Y \subseteq Z$, $\Sigma(Y) \leq t$, note that $|Y| \leq m$

consider a **random** partitioning $Z = Z_1 \cup \cdots \cup Z_m$

this does not split Y (we would need m^2 bins)

but it **almost splits** Y: whp we have $|Y \cap Z_i| \leq O(\log n)$ for all i

Proof: $|Y \cap Z_i| = \text{Bin}(|Y|, 1/m)$ is binomially distributed

use Chernoff

Two-Level Partitioning

for computing all subset sums, we use a two-level partitioning approach

assume: $Z \subseteq [t/m, 2t/m]$ for some m

fix $Y \subseteq Z$, $\Sigma(Y) \leq t$, note that $|Y| \leq m$

consider a **random** partitioning $Z = Z_1 \cup \cdots \cup Z_m$

this does not split Y (we would need m^2 bins)

but it **almost splits** Y: whp we have $|Y \cap Z_i| \leq O(\log n)$ for all i

$O(\log n \cdot t/m)$ .. can work with smaller target

thus $S_i :=$ ColorCoding($Z_i, \cancel{t}, O(\log n)$) contains $\Sigma(Y \cap Z_i)$ whp

thus $S_1 \oplus_t \ldots \oplus_t S_m$ contains $\Sigma(Y)$ whp

running time $\tilde{O}(m \cdot t)$, so what did we gain?

Two-Level Partitioning

assume $Z \subseteq [t/m, 2t/m]$ for some m

1) $S_i \coloneqq \text{ColorCoding}(Z_i, \underbrace{O(\log n \cdot t/m)}_{t'}, \underbrace{O(\log n)}_{k'})$

2) $S_1 \oplus_t \ldots \oplus_t S_m$

1)

running time of color-coding is $\tilde{O}(t' \cdot k'^2) = \tilde{O}(t/m)$

we do this for m sets: $\tilde{O}(t)$

Two-Level Partitioning

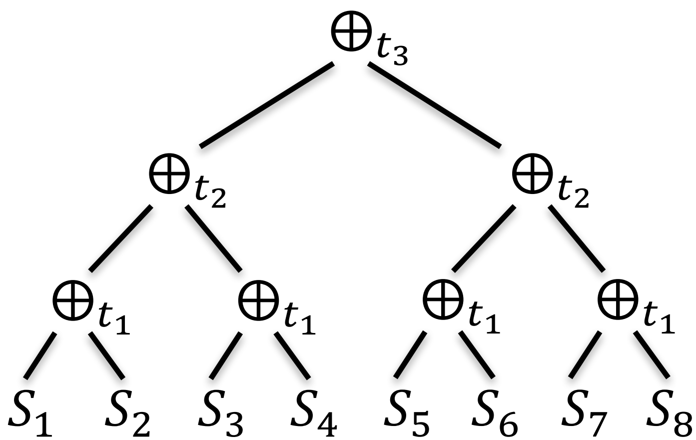
assume $Z \subseteq [t/m, 2t/m]$ for some m

1) $S_i \coloneqq \text{ColorCoding}(Z_i, O(\log n \cdot t/m), O(\log n))$

2) $S_1 \oplus_t \dots \oplus_t S_m$

2)

level



level	all elements are at most
3	$t_3 = 8 \cdot O(\log n \cdot t/m)$
2	$t_2 = 4 \cdot O(\log n \cdot t/m)$
1	$t_1 = 2 \cdot O(\log n \cdot t/m)$
0	$1 \cdot O(\log n \cdot t/m)$

in level i there are $m/2^i$ sets with elements bounded by $O(2^i \cdot \log n \cdot t/m)$

sumset computation in level i takes time $\tilde{O}(2^i \cdot t/m)$

total time $\tilde{O}(t)$ for computing $S_1 \oplus_t \dots \oplus_t S_m$

Two-Level Partitioning

assume: $Z \subseteq [t/m, 2t/m]$ for some m

Partitioning(Z, t):

consider a **random** partitioning $Z = Z_1 \cup \cdots \cup Z_m$

compute $S_i :=$ ColorCoding$(Z_i, O(\log n \cdot t/m), O(\log n))$ for all i

return $S := S_1 \oplus_t \ldots \oplus_t S_m$, computed in binary-tree-like way

this returns a set S containing only valid subset sums of Z

for any $Y \subseteq Z$ with $\Sigma(Y) \leq t$:

$\Sigma(Y) \in S$ with high probability ($\geq 1 - n^{-\Omega(1)}$)

running time: $\tilde{O}(t) = O(t \log t \log^4 n)$

Final algorithm

recall: we assumed $Z \subseteq [t/m, 2t/m]$ for some m

actually we only needed: (for some m)

1) *all items are small*: $Z \subseteq [0, O(t/m)]$

2) *all interesting subsets are small*: for any $Y \subseteq Z$ with $\Sigma(Y) \leq t$ we have $|Y| = O(m)$

this is satisfied for $Z \subseteq [t/m, 2t/m]$ as well as $Z \subseteq [0, t/n]$ (with $m \coloneqq n$)

recall: $n = |Z|$

FasterSubsetSum(Z, t):

split Z into $Z_i \coloneqq Z \cap [t/2^i, t/2^{i-1}]$ for $i = 1, \ldots, L = O(\log n)$

and $Z_{L+1} \coloneqq Z \cap [0, t/2^L]$

compute $S_i \coloneqq$ Partitioning(Z_i, t) for all i

return $S \coloneqq S_1 \oplus_t \ldots \oplus_t S_{L+1}$

running time $\tilde{O}(t) = O(t \log t \log^5 n)$, same whp-error bound as Partitioning

Polynomial Space

Is there an $\tilde{O}(t)$ algorithm?

Thm: Subset Sum is in randomized time $\tilde{O}(t)$. [B. SODA'17]

one-sided error probability $1/n$, time $O(t \log t \log^5 n)$

uses space $\tilde{O}(t)$

polynomial space is known: $\tilde{O}(n^3 t)$ time and $\tilde{O}(n^2)$ space [Lokshtanov,Nederlof'10]

Thm: Subset Sum has a randomized algorithm with [B. SODA'17]

- time $\tilde{O}(nt)$ and space $\tilde{O}(n \log t)$, assuming ERH
- time $\tilde{O}(nt \cdot \min\{n, t^{\varepsilon}\})$ and space $\tilde{O}(n \cdot \min\{n, t^{\varepsilon}\})$, unconditional

Algorithm by Lokshtanov and Nederlof

interpret a Subset Sum algorithm as a circuit:

→ **circuit over** $\cup$ and $\oplus$, each gate computes a subset of $\{0, \dots, f(n,t)\}$

→ output gate computes the set of all subset sums

translate to characteristic vectors:

→ **circuit over** $\vee$ and Boolean conv., each gate computes a vector of length $f(n,t)$

translate to integer vectors „counting“ the number of solutions: up to $2^{\Theta(n)}$

→ **circuit over** $+$ and convolution, each gate computes a vector of length $f(n,t)$

go to Fourier domain:

→ **circuit over** $+$ and $\times$ (pointwise operations!) with $g(n)$ gates, length $f(n,t)$

evaluating an entry of the output vector of the circuit: $\tilde{O}(g(n))$ time and space

inverse Fourier transform is a simple sum,
so we can evaluate all entries independently

total number of arithmetic operations $\tilde{O}(f(n,t) \cdot g(n))$, storing $\tilde{O}(g(n))$ numbers

need $O(n)$-bit numbers

Algorithm by Lokshtanov and Nederlof

total time $\tilde{O}\big(n \cdot f(n,t) \cdot g(n)\big)$ and space $\tilde{O}\big(n \cdot g(n)\big)$

they use a Subset Sum algorithm with $f(n,t) = \tilde{O}(nt)$ and $g(n) = \tilde{O}(n)$

1) plugging in our new Subset Sum algorithm: $f(n,t) = \tilde{O}(t)$ and $g(n) = \tilde{O}(n)$

2) work modulo a random prime p to decrease precision from $O(n)$ to $\mathrm{polylog}(n)$ bits

vectors have length $f(n,t) = \ell$

need a primitive ℓ-th root of unity for Fourier transform

this exists in $\mathbb{Z}_p$ if ℓ divides $p - 1$

so we want to choose p as a random prime in the arithmetic progression $1 + \ell \cdot \mathbb{N}$

need that the arithmetic progression contains many primes (see Dirichlet's Thm)

ERH helps with this

Polynomial Space

Thm: Subset Sum is in randomized time $\tilde{O}(t)$.

uses space $\tilde{O}(t)$

polynomial space is known: $\tilde{O}(n^3 t)$ time and $\tilde{O}(n^2)$ space [Lokshtanov,Nederlof'10]

Thm: Subset Sum has a randomized algorithm with [B. SODA'17]

- time $\tilde{O}(nt)$ and space $\tilde{O}(n \log t)$, assuming ERH
- time $\tilde{O}(nt \cdot \min\{n, t^{\varepsilon}\})$ and space $\tilde{O}(n \cdot \min\{n, t^{\varepsilon}\})$, unconditional

we plug our new algorithm into the framework by Lokshtanov and Nederlof

and work modulo a random prime (from an appropriate arithmetic progression)

OPEN: time $\tilde{O}(t)$ and space $n^{O(1)} \operatorname{polylog} t$

Summary

Thm: Subset Sum is in randomized time $\tilde{O}(t)$.

More open problems:

- derandomization

- approximation algorithms:

  if a subset sums to t, we compute a subset summing to a value in $[(1-\varepsilon)t, t]$

  best known running time $\tilde{O}(\min\{n/\varepsilon, n + 1/\varepsilon^2\})$ - improvements?

  [Lawler'79,Gens,Levner'80,KMPS'03]

- extensions:

  knapsack problem $O(nt)$, constraint shortest path $O(mt)$

  similar improvements possible?

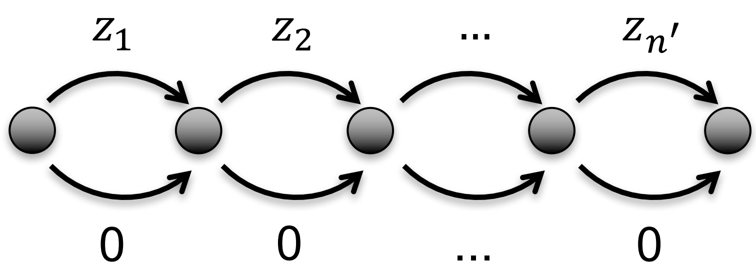
Extensions?

Subset Sum	$\leq$	Knapsack	$\leq$	Constraint Shortest Path
given n numbers, does any subset sum to t?		given n items, pick subset of total weight $\leq t$ and with largest value		given a graph, find a path from u to v with total delay $\leq t$ and with smallest length
$\tilde{O}(nt)$		$\tilde{O}(nt)$		$\tilde{O}(mt)$
$\tilde{O}(t)$		?		$(nt)^{1-o(1)}$
$t^{1-o(1)}$		$t^{1-o(1)}$		$t^{1-o(1)}$

Extensions?

start with **hard Subset Sum instance**
with target t and $n' = t^{o(1)}$,
e.g. from Set Cover or combinatorial k-Clique

build graph with lengths $= -$ delays:



lower bound $t^{1-o(1)}$

Constraint Shortest Path

given a graph, find a path from u to v with total delay $\leq t$ and with smallest length

$$\tilde{O}(mt)$$

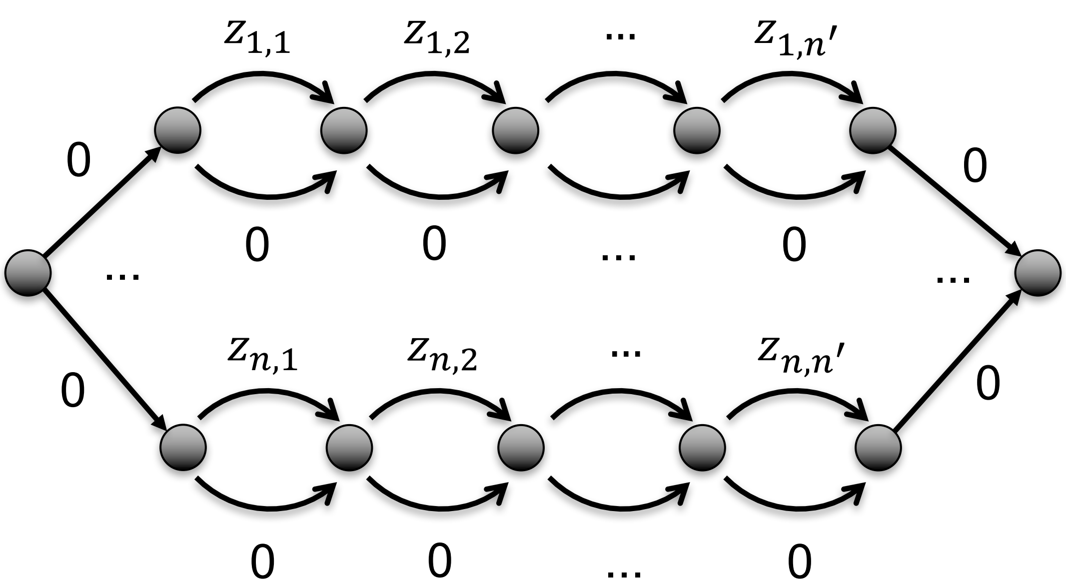
$$(nt)^{1-o(1)}$$

$$t^{1-o(1)}$$

Extensions?

start with n **hard Subset Sum instances**
with target t and $n' = t^{o(1)}$,
e.g. from Set Cover or combinatorial k-Clique

combine graphs as follows:



lower bound $(nt)^{1-o(1)}$

Constraint Shortest Path

given a graph, find a path from u to v with total delay $\leq t$ and with smallest length

$$\tilde{O}(mt)$$

$$(nt)^{1-o(1)}$$

$$t^{1-o(1)}$$