arXiv:2601.01390v1 [cs.DS] 4 Jan 2026

去随机化 子集和伪多项式算法

Timothy M. Chan*

2026年1月6日

摘要

我们重新审视经典的子集和问题：给定一个由 n 个正整数组成的集合 X 以及一个数 t，判断是否存在一个子集 X，其元素之和为 t；或者更一般地，计算所有数构成的集合 OUT，其中每个数 $y \in \{0, \ldots, t\}$ 都满足存在一个子集 X，其元素之和为 y。标准动态规划可在 $O(tn)$时间内解决该问题。在 SODA'17 上，有两篇论文给出了目前最优的确定性和随机化算法（忽略多对数因子）：Koiliaris 和 Xu 的确定性算法可在$O(t\sqrt{n})$时间内运行，而 Bringmann 的随机化算法可在 $O(t)$时间内运行。~我们提出了首个可在 $O(t)$时间内运行的确定性算法。

我们的技术还有许多其他应用：例如，我们也可以对 Bringmann 和 Nakos [STOC'20] 以及 Bringmann、Fischer 和 Nakos [SODA'25] 近期提出的、更具输出敏感性的算法进行去随机化；这些算法的运行时间分别为 $\widetilde{O}(|\text{OUT}|^{4/3})$ 和 $\widetilde{O}(|\text{OUT}|\sqrt{n})$。此外，我们还可以对 Cygan 等人¯[ICALP'17]提出的、从 0-1 背包问题到最小加卷积的先前细粒度归约进行去随机化。

1 引言

本文研究算法设计中的一个教科书问题——子集和 问题：给定一个由 n 个正整数构成的（多重）集合 X 和一个目标数 t，判断是否存在一个子集 X，其元素之和等于 t。该问题是弱 NP 完全的。标准算法的运行时间约为 $O(2^{n/2})$（采用折半相遇技巧 [HS74]）和 $O(tn)$（采用动态规划 [Bel57]）。尽管最坏情况下能否改进指数级界 $2^{n/2}$ 仍未有定论，但伪多项式界 $O(tn)$ 已得到改进：在同期 SODA'17 论文中，Koiliaris 和 Xu [KX19] 提出了一种运行时间为 $\widetilde{O}(t\sqrt{n})$ 的确定性算法，[1] 而 Bringmann [Bri17]提出了一种运行时间为 $O(t)$ 的随机化（蒙特卡罗）算法。随后，Jin 和 Wu [JW19]又提出了一种优雅的、运行时间为 $O(t)$ 的随机化（蒙特卡罗）算法。

所有这些伪多项式算法都能解决该问题一个更强的"全目标"版本：计算 OUT $= \{y \in \{0, \ldots, t\} :$ 是否存在一个 X 的子集，其元素之和为 $y\}$。它们还可以为每个 $y \in$ OUT 计算一个可行子集的表示（这样之后便可在每个元素耗时 $\sim$ $\sim O(1)$ 的情况下报告一个子集）。对于该问题的全目标版本，随机化的 $O(t)$ 时间算法显然已接近最优，因为在最坏情况下，输出大小为 $\Omega(t)$。对于该问题的原始版本，Abboud、Bringmann、Hermelin 和 Shabtay [ABHS22] 还证明，假设 SETH 成立，则对于任意常数 $\varepsilon > 0$，不存在时间上界为 $O(t^{1-\varepsilon}2^{o(n)})$ 的算法（另请参见 [CDL+16]，该文在不同假设下给出了更早的条件下界）。

*伊利诺伊大学厄巴纳-香槟分校 Siebel计算与数据科学学院，美国（tmc@illinois.edu）[1]本文中，$\widetilde{O}$ 记号省略了关于 n 和 t的多对数因子。

近年来，人们提出了许多针对子集和问题的其他算法，这些算法考虑了不同的参数（例如，输出大小 |OUT| [BN20、BFN25], 最大输入值 max(X) [PRW21、CLMZ24a、CLMZ24b], 、密度 [BW21], 、空间使用量 [LN10、Bri17、JVW21、NW21、BCKM24], ...）。人们还研究了许多变体（例如，无界子集和 [Bri17], 、模子集和 [ABJ$^+$19、ABB$^+$21、CI21], 、近似子集和 [KMPS03、CLMZ24a], ...）。该问题还与背包问题和换零钱问题等其他基础问题相关，已成为当前非常活跃的研究课题。

去随机化？ SODA'17 两篇论文中的结果自然引出了这样一个问题：是否存在一个运行时间为$\widetilde{}$确定性 $O(t)$的子集和算法，而这个问题在过去 8 年中一直悬而未决。Bringmann 的随机化算法使用了颜色编码技术 [AYZ95], ；尽管过去一些基于颜色编码的算法已经成功实现了去随机化（例如，用于计算图中的 k-简单路径/环），但对颜色编码进行去随机化所需的开销似乎对于子集和的应用来说过大（我们所期望的 $O(t)$ 上界几乎没有余地来吸收这种开销）。如果允许一个和中包含的元素数量 k 为次对数级，那么去随机化的代价是可以接受的；但当 k 较大时则不然。在不使时间上界急剧增大的情况下，如何应用有限独立性方法或条件概率/期望法等标准技术，目前尚不清楚。Jin 和 Wu 的另一种随机化算法 [JW19] 似乎更难实现去随机化——它对模一个随机素数的计数进行计算，而目前没有显而易见的高效方法来保证选出的素数是正确的。即使只是将这些蒙特卡罗随机算法转化为拉斯维加斯算法，也仍然是一个悬而未决的问题。另一方面，Koiliaris 和 Xu 的确定性方法 [KX19]（另一种方案也见 [KX18]）似乎难以进一步改进——他们确实给出了一个运行时间为 $\widetilde{O}(t^{4/3})$ 的变体，但没有明确的方法可以达到 $O(t^{1+\varepsilon})$。相比之下，对于无界子集和[2] [Bri17] 以及模子集和 [ABB$^+$21]等变体，高效的去随机化并不困难，而且已经有现成的方法。

$\widetilde{}$**新算法**。我们提出了首个运行时间为 $O(t)$的子集和确定性算法。它还解决了该问题的全目标版本。

我们的算法可以视为对 Bringmann 的某个变体 [Bri17] 的去随机化。我们不再通过颜色编码将输入集划分为多个互不相交的子集，而是递归地将输入集划分为两个子集，使得每个目标值对应的所需子集，按其基数计算，都大致均分为两半（如果该子集约有 k 个元素，误差约为 $\sqrt{k}$ 的加性误差）。将 Bringman 的算法重新诠释为通过"二分器"（或类似的随机划分）进行分治，这并非新鲜事，在之后其他用于子集和与背包问题的随机化算法中也已出现（例如，参见 [BN20、BDP24、BFN25]）。然而，尽管概率方法（通过标准的 Chernoff 界或差异界）保证了优质二分器的存在，但其确定性构造似乎需要知道所有目标值对应的子集，也就是说，事先就得知道所有答案！我们的新方法是通过降低二分器的质量来加快其构造。[3] 只要将加性误差控制为略低于线性（以 $k/\log k$ 为界），分治仍然是高效的。为此，我们证明，对于任意子集，都存在一个与之等价（总和相同）的子集，可以将其划分为次线性数量的小型"规范"子集；随后，我们针对这些子集直接构造二分器

[2]在无界子集和问题中，我们可以多次使用同一个元素来求和，得到目标值。这个问题通常更容易，因为原始问题的大部分难点在于防止一个元素被重复使用。[3]这个思路在很大程度上受到计算几何中关于"ε-网"和"ε-近似"的去随机化研究的启发 [Cha01, Mat00]。其中一个具体灵感来自作者对低维线性规划递归算法的去随机化 [Cha18], : 该方法只需将输入集划分为若干小子集，并对每个子集应用一个较慢的 ε-网算法，便能构造出质量较低的 ε-网（足以满足该递归算法的需要）。

规范子集。相对于原始子集而言，二分器的质量确实会下降，但由于规范子集的总数不多且规模较小，构造起来反而更省力。不过，如何生成优质的规范子集呢？我们的想法是调整 Koiliaris 和 Xu 的确定性算法来实现这一目的！以恰当的方式将这些部分组合起来后，整个方案相对简单，篇幅仅为 3 页（见第 2 节）。

其他推论。我们关于二分器构造的新思路也有助于对其他算法进行去随机化。这里介绍两个应用：

1. Bringmann 和 Nakos [BN20] 以及 Bringmann、Fischer 和 Nakos [BFN25] 最近分别提出了用于子集和问题全目标版本的随机化输出敏感算法，运行时间分别为 $\widetilde{O}(|\mathrm{OUT}|^{4/3})$ 和 $\widetilde{O}(|\mathrm{OUT}|\sqrt{n})$。（标准动态规划 [Bel57] 可以调整为在 $O(|\mathrm{OUT}|n)$ 时间内运行。）我们的技术可以对这两种算法进行去随机化，时间上界仅相差关于 n 和 t 的多对数因子。

2. Cygan、Mucha、W.egrzycki 和 Włodarczyk [CMWW19] 给出了一种随机化归约，表明如果能在 $O(n^{2-\varepsilon})$ 时间内解决含有 n 个元素的最小加卷积问题，那么对于具有正整数重量（上界为 t）和正实数收益的物品，0-1 背包问题可以在随机化的 $\widetilde{O}(t^{2-\varepsilon'})$ 时间内求解。我们的技术可以对这一结果进行去随机化。（近年来，对归约进行去随机化受到越来越多的关注；例如，参见 [CH20, CX24, FKP24]。）特别地，结合 Chan 和 Williams 的确定性最小加矩阵乘法算法 [CW21] 以及从最小加矩阵乘法到卷积的已知归约 [BCD+14]，可得到一个用于 0-1 背包问题的确定性 $O(t^2/2^{\Omega(\sqrt{\log t})})$ 时间算法。

上文的第二个应用可由我们的技术直接推出（见附录 A）。不过，第一个应用更有意思，也需要一些新的想法，因为我们不再能采用 Koiliaris 和 Xu 的方法来生成规范子集（见第 3 节）。

记为。令 $[u]$ 表示 $\{0,1,\ldots,u\}$。对于由整数构成的集合 S，令 $\sigma(S) = \sum_{s\in S} s$。定义

$$\Sigma_i(X) = \{y : \exists S \subseteq X \text{ with } |S| = i \text{ and } \sigma(S) = y\}.$$

令 $\Sigma_{\le k}(X) := \bigcup_{i\in[k]} \Sigma_i(X)$，且 $\Sigma(X) = \Sigma_{\le\infty}(X)$。子集和问题的全目标版本是：给定 t，计算 $\Sigma(X) \cap [t]$。

2 An $O(t)$ 时间算法

In t 本节中，我们给出一个用于全目标子集和的确定性 $O(t)$ 时间算法。

2.1 预备知识：回顾 Koiliaris 和 Xu 算法

许多关于子集和的先前工作，都是从考虑相关问题开始的：对于给定的 k，计算 $\Sigma_{\le k}(X)$（或一个 a 超集）。Bringmann [Bri17] 在随机化的 $O(ku)$ 时间内解决了这个问题，其中 $u = \max(X)$；而 Koiliaris 和 Xu [KX19] 则在确定性时间 $\widetilde{O}(k^2u)$ 内解决了该问题。我们先回顾 Koiliaris 和 Xu 的解法，因为稍后会用到。其方法采用二进制分治，并将卷积作为子程序。下面我们介绍一个简化版本，其运行时间为 $\widetilde{O}(k^3u)$，足以满足我们的应用需求；我们将用二进区间来重新诠释这种分治方法。

引理 2.1。 给定一个由 X 个整数构成的集合 $[u]$ 和一个数 k，我们可以在确定性时间 $\widetilde{O}(k^3u)$ 内，计算所有 $i \in [k]$ 对应的 $\Sigma_i(X)$ 。

之后，对于任意给定的 $i \in [k]$ 和 $y \in \Sigma_i(X)$，我们可以在 $O(k)$ 时间内，报告一个满足 $|S| = i$ 且 $\sigma(S) = y$ 的子集 $S \subseteq X$ 。

证明。 不失一般性，假设 u 是 2 的幂。二进区间 是形如 $(a, a + \ell]$ 的区间，其中长度 ℓ 是 2 的幂，且 a 可被 ℓ整除。对于每个二进区间 $I = (a, a + \ell] \subseteq (0, u]$，我们希望计算 $\{\Sigma_i(X \cap I)\}_{i\in[k]}$。

将 I分解为两个二进子区间 $I' = (a, a + \ell/2]$ 和 $I'' = (a + \ell/2, a + \ell]$。我们可以根据公式$C_i = \bigcup_{i',i''\colon i'+i''=i} (A_{i'} + B_{i''})$，由 $\{A_{i'} := \Sigma_{i'}(X \cap I')\}_{i'\in[k]}$ 和 $\{B_{i''} := \Sigma_{i''}(X \cap I'')\}_{i''\in[k]}$ 计算$\{C_i := \Sigma_i(X \cap I)\}_{i\in[k]}$。由于对于每个 $i', i'' \in [k]$，$A_{i'} \subseteq (ai', (a + \ell/2)i']$ 和 $B_{i''} \subseteq ((a + \ell/2)i'', (a + \ell)i'']$ 都包含在长度为 $O(k\ell)$的区间中，因此我们可以在 $O(k\ell \log(k\ell))$ 时间内，通过两个长度为 $O(k\ell)$的布尔向量的卷积来计算每个和集 $A_{i'} + B_{i''}$。对于每个元素 $z \in A_{i'} + B_{i''}$，我们还可以使用 Alon 和 Naor 的确定性见证查找技术 [AN96]，在 $O(k\ell \log^{O(1)}(k\ell))$ 时间内计算一个满足 $x + y = z$的见证 $(x, y) \in A_{i'} \times B_{i''}$。遍历所有 $O(k^2)$ 个 (i', i'') 选择的总成本为 $O(k^3\ell \log^{O(1)} \ell)$。对于每个元素 $z \in C_i$，我们还可以记录一个满足 $i' + i'' = i$ and$z \in A_{i'} + B_{i''}$的 (i', i'')。

沿着自底向上的顺序处理所有二进区间，这些区间的总长度为 $O(u \log u)$，我们得到总运行时间 $\widetilde{O}(k^3u)$。对于根区间 $I_0 = (0, u]$，我们可以通过利用卷积的见证进行回溯，在 $O(k)$ 时间内报告与给定的 $i \in [k]$ 和 $y \in \Sigma_i(X \cap I_0)$ 相对应的子集。 □

(注。Koiliaris 和 Xu 的算法原始版本 [KX19] 达到了 $\widetilde{O}(k^2u)$ 的运行时间，而不是 $\widetilde{O}(k^3u)$，其本质是注意到，C_i 可以通过对长度为 $O(k^2\ell)$ 的向量进行一次卷积来计算，而无需进行 $O(k^2)$ 次长度为 $O(k\ell)$ 的向量卷积。他们对分治的执行方式也略有不同：将 X 的元素 $x_1, \ldots, x_n$ 排序，并为二进区间 $I \subseteq (0, n]$ 计算 $\{\Sigma_i(\{x_j \colon j \in I\})\}_{i\in[k]}$。这样可以将一些 log u 因子替换为 log n，这对我们并不重要，不过这一替代方法稍后在第 3 节中会有用。)

2.2 二分器

我们的算法背后的关键思 路涉及构造一个二分器，其定义如下 w:

定义 2.2. 给定一个正整数**集合** X，以及数值 k 和 Δ，X 的(k, Δ)-二分器 是将 X 划分为两个不相交子集 X' 和 X''，并满足以下性质：

$\hat{}$ $\hat{}\hat{}$对于每个满足 $|S| \le k$的子集 $S \subseteq X$，都存在一个满足 $|S| \le k$、$\sigma(S) = \sigma(S)$的子集 $S \subseteq X$，且 $|\hat{}S \cap X'|, |\hat{}S \cap X''| \le |\hat{}S|/2 + \Delta$。

请注意上述定义中存在量词的一点微妙之处（因为可能有多个子集具有相同的基数和相同的和 S）。我们回顾差异理论中的一个著名事实：

引理 2.3. 给定一个集合 X 以及由 X 的子集组成的集合族 $\mathcal{C}$，其中每个子集的大小至多为 k，我们可以将 X 划分为两个子集 X'和 X''，使得对于所有$S \in \mathcal{C}$，$|S \cap X'|, |S \cap X''| \le |S|/2 + O(\sqrt{|S|\text{对数 } |\mathcal{C}|})$，并且可以在 $O(k|\mathcal{C}| + |X|)$ 确定性时间内完成。

[4]Alon 和 Naor 的巧思 数性寻找技术最初是为矩阵乘法描述的，但这项技术具有普适性，且 d 适用于卷积。

证明。 这种划分 (X', X'') 的存在是差异理论中的一个基本事实 [Cha01]。（通过简单应用 Chernoff界，随机划分以高概率有效。）可以通过条件期望法确定性地构造 (X', X'')；关于详细说明，参见 [Cha01, Theorem 1.2] [5]

□

上述引理意味着存在一个（k, $\widetilde{O}(\sqrt{k})$）-二分器：对于每个 $y \in [ku]$，如果存在，我们只需固定一个满足 $|S| \le k$ 和 $\sigma(S) = y$ 的子集 $S \subseteq X$，将其加入集合 $\mathcal{C}$，然后对该集合应用引理 2.3。然而，构造时间过长（我们必须先解决子集和问题，才能构造集合 $\mathcal{C}$！）。为缩短构造时间，对于参数 $b \le k$，我们将二分器的加性误差从$\widetilde{O}(\sqrt{k})$ 增大到 $\widetilde{O}(k/\sqrt{b})$。具体而言，我们利用 Koiliaris 和 Xu 算法生成的、大小至多为 b 的小型“规范”子集构造一个集合，然后对该集合应用引理 2.3，以构造一个二分器：

引理 2.4 （典范子集生成）。给定一个由 $[u]$ 中整数构成的集合 X 和一个数 b，我们可以在 $\widetilde{O}(b^3u)$ 确定性时间内，生成一个包含 $\widetilde{O}(b^2u)$ 个“典范”子集的集合 $\mathcal{C}$，这些子集均包含于 X，且大小至多为 b，并满足以下性质：

对于每个子集 $S \subseteq X$，都存在一个子集 $S \subseteq X$，使得 $|S| = |S|$，$\sigma(S) = \sigma(S)$，并且 S 可以表示为 $\mathcal{C}$ 中 $O((|S|/b+1)\log u)$ 个互不相交的典范子集的并。

证明。令 $k = b$，运行引理 2.1 证明中的算法。对于每个二进区间 $I = (a, a+\ell]$，$i \in [b]$，以及 $y \in \Sigma_i(X \cap I)$），找到一个满足 $|S_I[i,y]| = i$ 和 $\sigma(S_I[i,y]) = y$ 的子集 $S_I[i,y] \subseteq X \cap I$，并将该子集加入集合 $\mathcal{C}$。这类子集的数量为每个 I 有 $O(b^2\ell)$ 个（因为 $\sim$ 有 b 种选择，且 $y \in (ai, (a+\ell)i]$ 有 $O(\ell b)$ 种选择，而每种选择都可以在 $O(b)$ 时间内生成。所有二进区间 I（其总长度为 $O(u\log u)$）中，子集总数为 $\widetilde{O}(b^2u)$。

为证明 $\mathcal{C}$ 满足所述性质，考虑任意子集 $S \subseteq X$。首先将 $(0, u]$ 划分为一族互不相交的二进区间 $\mathcal{I}(S)$，其中每个区间包含的 S 中的点不超过 b 个：具体而言，我们从访问根二进制区间开始；每当访问到一个包含的 S 中的点多于 b 个的二进区间时，就递归访问它的两个子区间。每一层访问的二进区间数为 $O(|S|/b+1)$。因此，生成的二进区间总数为 $|\mathcal{I}(S)| \le O((|S|/b+1)\log u)$。对于每个区间 $I \in \mathcal{I}(S)$，令 $i_I = |S \cap I| \in [b]$ 和 $y_I = \sigma(S \cap I) \in \Sigma_{i_I}(X \cap I)$。定义 S 为所有 $I \in \mathcal{I}(S)$ 对应的 $S_I[i_I, y_I]$ 的并集；这些集合 $S_I[i_I, y_I] \subseteq X \cap I$ 显然互不相交，因为 $\mathcal{I}(S)$ 中的区间互不相交；S 且 $|S_I[i_I, y_I]| = |S \cap I|$ 和 $\sigma(S_I[i_I, y_I]) = \sigma(S \cap I)$。因此，$|S| = |S|$ 和 $\sigma(S) = \sigma(S)$。 □

引理 2.5 (快速二分器构造)。给定 X 中的整数集合，我们可以在 $[u]$ 和 $b \le k$中以确定性时间计算 X 的$(k, \widetilde{O}(k/\sqrt{b}))$-二分器。

证明。 应用引理 2.4，在 $\widetilde{O}(b^3u)$ 时间内构造一个包含 $\mathcal{C}$ 个 $\widetilde{O}(b^2u)$ 规范子集的集合。然后对该集合 $\mathcal{C}$ 应用引理 2.3，将 k 替换为 b，在 $\widetilde{O}(b^3u)$ 时间内计算划分 (X', X'')。

为证明 (X', X'') 满足 $(k, \widetilde{O}(k/\sqrt{b}))$-二分器性质，考虑任意子集 $S \subseteq X$，满足 $|S| \le k$。令 $S \subseteq X$ 满足 $|S| = |S| \le k$, $\sigma(S) = \sigma(S)$，且 S 可表示为 $O((k/b)\log u)$ 个互不相交的规范子集 $S_j \in \mathcal{C}$ 的并集。对于每个这样的子集 S_j，我们有 $|S_j \cap X'|$，$|S_j \cap X''| \le |S_j|/2 + \widetilde{O}(\sqrt{b})$。对 S_j 求和，得到 $|S \cap X'|$，$|S \cap X''| \le |S|/2 + \widetilde{O}(k/b \cdot \sqrt{b})$。

□

[5] 其中陈述的时间上界为 $O(|X||\mathcal{C}|)$，但由于我们的所有子集大小都至多为 k，算法的运行时间实际上为 $O(k|\mathcal{C}| + |X|)$。

2.3 将所有内容整合起来

最后，我们使用二分器，得到一个用于子集和的确定性分治算法（可将其视为 Bringmann 的随机化算法 [Bri17] 的一个变体）：

引理 2.6. 给定一个 X 中的整数集合，以及一个数 k，我们可以在确定性时间 $O(ku)$ 内计算出一个集合$\text{OUT} \subseteq [ku]$，使得$\sim\Sigma_{\le k}(X) \subseteq \text{OUT} \subseteq \Sigma(X)$。

证明。 算法如下：

~1. 通过对一个足够大的常数 c取 $b = \log^c(ku)$，并假设 $k > b$，应用引理 2.5，可在 $O(u)$时间内计算 X的一个$(k, k/\log k)$-二分器(X', X'')。2. 递归地对输入集 X'和数 $k/2 + k/\log k$求解问题；令 $\text{OUT}' \subseteq [ku]$为输出集合。递归地对输入集 X''和数 $k/2 + k/\log k$求解问题；令 $\text{OUT}'' \subseteq [ku]$ 为输出集合。3. 返回截断和集 $\text{OUT} = (\text{OUT}' + \text{OUT}'') \cap [ku]$；该集合可通过对两个长度为 $O(ku)$的布尔向量进行卷积，在 $O(ku)$时间内计算得到。

运行时间满足以下递推式：

$$T(k, u) = 2\,T(k/2 + k/\log k, u) + \widetilde{O}(ku).$$

对于基本情况 $k \le b$（这意味着 $k \le \log^{O(1)} u$），我们可以使用引理 2.1 在 $\widetilde{O}(u)$~ ~ 时间内计算 $\Sigma_{\le k}(X)$，从而得到 $T(k, u) = O(u)$。该递推式可解得 $T(k, u) = O(ku)$。

为证明正确性，考虑一个包含 $|S| \le k$ 且 $\sigma(S) = y$ 的子集 $S \subseteq X$。根据二分器的定义，存在一个子集 $\hat{} S \subseteq X$，使得 $|\hat{}S| \le k$, $\sigma(\hat{}S) = y$，且 $|\hat{}S \cap X'|$，$|\hat{}S \cap X''| \le k/2 + k/\log k$。于是，根据归纳法，$\sigma(\hat{}S \cap X') \in \text{OUT}'$ 且 $\sigma(\hat{}S \cap X'') \in \text{OUT}''$，因此 $y = \sigma(\hat{}S \cap X') + \sigma(\hat{}S \cap X'') \in \text{OUT}$。 □

定理 2.7。*Gi* ~给定一个集合 X，其中包含 n 整数，位于 $[t]$中，我们可以在 $O(t)$ </style}确定性时间内计算 $(X) \cap [t]$.

证明。对于每个 $i \in [\lceil \log t \rceil]$，令 $X_i = X \cap (t/2^i, t/2^{i-1}]$。通过将引理 2.6 应用于 $u = t/2^{i-1}$和 $k = 2^i$，计算 $\Sigma(X_i) \cap [t]$，其耗时为 $\widetilde{O}(2^i \cdot t/2^{i-1}) = \widetilde{O}(t)$ 时间（这是因为当 $X_i \subseteq (t/2^i, \infty)$ 时，$\Sigma(X_i) \cap [t] = \Sigma_{\le 2^i}(X_i) \cap [t]$）。我们返回截断和集 $((\Sigma(X_0) \cap [t]) + (\Sigma(X_1) \cap [t]) + \cdots) \cap [t]$，它可以通过 $O(\log t)$ 次长度为 $O(t)$ 的布尔向量卷积，在 $O(t)$ 时间内计算得到。 □

3 输出敏感算法

本节将介绍我们的技术的另一项应用：对 Bringmann、Fischer 和 Nakos [BFN25], 提出的近期全目标子集和的输出敏感算法进行去随机化~，该算法的运行时间为$O(|\Sigma(X) \cap [t]|\sqrt{n})$。Bringmann 和 Nakos [BN20], 提出的更早的输出敏感算法的运行时间为 $\widetilde{O}(|\Sigma(X) \cap [t]|^{4/3})$，同样也可以用类似的方法进行去随机化，我们将在最后简要提及。

3.1 预备知识

我们从一个关于和集的输出敏感计算的已知子程序开始。Cole 和 Hariharan [CH02] 首次通过随机化解决了这一子问题，随后一系列论文对其进行了去随机化和改进 [CL15、BFN21、BFN22、JX24]:

引理 3.1。给定两个由 $[t]$ 中整数构成的集合 A 和 B，我们可以在 $O(|A+B|)$确定性时间内计算和集 $A+B$。

输出敏感的截断和集计算代价更高。Bringmann 和 Nakos [BN20] 描述了一种简单的方法（使用引理 3.1 中的子程序），可得到以下结果:

引理 3.2。给定两个由 $[t]$ 中整数构成的集合 A 和 B，我们可以在 $\widetilde{O}(\sqrt{|A||B||C|}) \leq \widetilde{O}((|A|+|C|)\sqrt{|B|})$ 确定性时间内计算截断和集 $C=(A+B)\cap[t]$。

下一个引理由 Bringmann、Fischer 和 Nakos [BFN25 的引理 3.1] 中提出（但表述形式不同），它使用 Bringmann 的颜色编码方法 [Bri17] 对较小的 k 情形进行去随机化。为完整起见，我们简要给出证明概要:

引理 3.3. 给定一个由 $[t]$ 中 n 个整数组成的集合 X、一个由 $[t]$ 中的整数构成的集合 A，以及一个数 k，我们可以在 $\widetilde{O}(k^{O(1)}|(A+\Sigma(X))\cap[t]|\sqrt{n})$确定性时间内计算出一个集合 OUT，使得 $(A+\Sigma_{\leq k}(X))\cap[t] \subseteq \text{OUT} \subseteq (A+\Sigma(X))\cap[t]$。

此后，对于任意给定的 $y \in \text{OUT}$，我们可以在 $O(|S|)$ 时间内报告一个数 $a \in A$ 和一个子集 $S \subseteq X$，满足 $a+\sigma(S)=y$。

证明。存在一个由 O 个（$n \log n$）“完美”哈希函数组成的族 $\mathcal{H}$，这些函数从 X 映射到 $\{1,\dots,k^2\}$，并满足以下性质：对于每个满足 $|S| \leq k$的子集 $S \subseteq X$，都存在一个 $h \in \mathcal{H}$，使得 $\{h(x)\colon\ x \in S\}$ 互不相同。该函数族可在 $O(k^{O(1)}\log^{O(1)} n)$ 时间内构造。例如，参见 [AYZ95，第 4] 节。

固定 $h \in \mathcal{H}$。对于每个 $j \in \{1,\dots,k^2\}$，令 $X_{h,\,j}=\{x \in X\colon\ h(x)=j\}$。计算截断后的和

$$\text{OUT}_h = (A+(X_{h,1}\cup\{0\})+\cdots+(X_{h,k^2}\cup\{0\}))\cap[t]$$

通过在 $\widetilde{O}(k^{O(1)}|(A+\Sigma(X))\cap[t]|\sqrt{n})$ 时间内应用引理 3.2 $O(k^2)$ 次。然后 $\text{OUT} = \bigcup_{h\in\mathcal{H}}\text{OUT}_h$满足输出要求。对于任意给定的 $y \in \text{OUT}$，我们都可以再次使用 Alon 和 Naor 的确定性见证查找技术 [AN96]，报告相应的数值 $a \in A$ 和子集 $S \subseteq X$。 □

3.2 二分器

和之前一样，关键思路是构造一个二分器。现在我们需要一个“加权”变体，它将一个子集的和而不是其基数平分:

定义 3.4。给定一个正整数集合 X 以及数值 t 和 Δ，X 的加权 (t,Δ)-二分器 是将 X 划分为两个不相交子集 X' 和 X'' 的一种划分，并具有以下性质:

$\hat{}$ $\hat{}$对于每个满足 $S \subseteq X$ 且满足 $\sigma(S) \leq t$ 的子集，都存在一个满足 $S \subseteq X$ 且满足 $\sigma(S)=\sigma(S)$，并且满足 $\sigma(\hat{}S\cap X') \ \sigma(\hat{}S\cap X'') \leq \sigma(\hat{}S)/2+\Delta$ 的子集。

和之前一样，我们将生成一组较小的规范子集，然后应用引理 2.3 来构造一个二分器。遗憾的是，Koiliaris 和 Xu 的算法不再适用，因为它不是输出敏感的。为此，我们将改用引理 3.3 中适用于较小 k 的算法。

然而，由于该算法并非通过分治或二进区间来运作，因此如何利用它获得规范子集并不明确。我们的新思路是，针对与不同二进区间相关的不同输入，多次调用引理 3.3 中的算法。乍看之下，界定这些多次调用的总成本似乎很棘手，但幸运的是，以下简单的观察可以解决这一问题：

观察 3.5。设 X 为一个数字有序列表。假设 X 被划分为连续子列表 X_1、…、X_m。则 $|\Sigma_{\le k}(X_1) \cap [t]| + \cdots + |\Sigma_{\le k}(X_m) \cap [t]| \le O(k)|\Sigma(X) \cap [t]|$。

证明。固定一个 i，$\Sigma_i(X_1), \ldots, \Sigma_i(X_m)$ 两两不相交，因为对于任意 j，$\Sigma_i(X_j)$ 中的每个元素都小于 $\Sigma_i(X_{j+1})$ 中的每个元素。因此，$|\Sigma_i(X_1) \cap [t]| + \cdots + |\Sigma_i(X_m) \cap [t]| \le |\Sigma_i(X) \cap [t]|$。对所有 $i \in [k]$ 求和即可得出该观察结论。□

引理 3.6（典范子集生成）。给定一个常数 $\delta \in (0, 1]$，X 一个由整数构成的集合，属于 $[u/(1+\delta)\ u]$，以及数字 t 和 b，我们可以在 $\widetilde{O}(b^{O(1)}|\Sigma(X) \cap [t]|\sqrt{n})$ 确定性时间内，生成一个由 $O(b|\Sigma(X) \cap [t]|)$ 个 $\mathcal{C}$ 组成的集合，其中每个“典范”子集 $\hat{X}$ 的大小至多为 $(1+\delta)b$，并满足以下性质：

> 对于每个满足 $\sigma(S) \le t$ 的子集 $S \subseteq X$，都存在一个子集 $\hat{S} \subseteq X$，使得 $\sigma(\hat{S}) = \sigma(S)$，并且 $\hat{S}$ 可以表示为 $\mathcal{C}$ 中 $O((|S|/b+1)\log n)$ 个两两不相交的规范子集之并。

证明。无妨设 n 是 2 的幂。将 X 的元素 x_1、…、x_n 排序。对于每个二进区间 $I = (a, a+\ell] \subseteq (0, n]$，在输入集 $X_I := \{x_j : j \in I\}$ 上运行引理 3.3 中的算法，其中 $A = \emptyset$、$k = b$ 和 t 替换为 $\min\{t, bu\}$，得到输出集 OUT_I。对于每个二进区间 I 以及每个 $y \in \mathsf{OUT}_I \subseteq \Sigma(X_I) \cap [\min\{t, bu\}]$，找出一个满足 $\sigma(S_I[y]) = y$ 的子集 $S_I[y] \subseteq X_I$，并将该子集加入集合 $\mathcal{C}$。注意，$|S_I[y]| \le bu/(u/(1+\delta)) = (1+\delta)b$。由观察 3.5 可知，对于任意固定长度的二进区间，$|\Sigma(X_I) \cap [\min\{t, bu\}]| = |\Sigma_{\le(1+\delta)b}(X_I) \cap [\min\{t, bu\}]|$ 的总和至多为 $O(b|\Sigma(X) \cap [\min\{t, bu\}]|)$；因此，对所有长度的二进区间求和，结果为 $O(b|\Sigma(X) \cap [\min\{t, bu\}]| \log n)$。所以，$\mathcal{C}$ 中子集的总数有界于 $O(b|\Sigma(X) \cap [t]| \log n)$，计算时间为 $\widetilde{O}(b^{O(1)}|\Sigma(X) \cap [t]|\sqrt{n})$。

为证明 $\mathcal{C}$ 满足所述性质，考虑任意子集 $S \subseteq X$。首先将 $(0, n]$ 划分为一组互不相交的二进区间 $\mathcal{I}(S)$，其中每个区间至多包含 b 个 $\{j$ 的指标：$x_j \in S\}$：具体来说，我们从访问根二进制区间开始；每当访问到一个包含超过 b 个 $\{j$ 的指标的二进区间时，就递归地访问它的两个子区间。每一层访问的二进区间数为 $O(|S|/b+1)$。因此，生成的二进区间数为 $|\mathcal{I}(S)| \le O((|S|/b+1)\log n)$。对于每个区间 $I \in \mathcal{I}(S)$，令 $y_I = \sigma(S \cap X_I) \in \Sigma_{\le b}(X_I) \cap [\min\{t, bu\}] \subseteq \mathsf{OUT}_I$。定义 $\hat{S}$ 为对所有 $I \in \mathcal{I}(S)$ 的 $S_I[y_I]$ 取并集；这些集合 $S_I[y_I] \subseteq X_I$ 显然互不相交，因为 $\mathcal{I}(S)$ 中的区间彼此不相交；并且 $\sigma(S_I[y_I]) = \sigma(S \cap X_I)$。因此，$\sigma(\hat{S}) = \sigma(S)$。□

引理 3.7（快速二分器构造）。给定 $[u]$ 中的 X 个 n 整数以及数 t 和 b，我们可以在 $\widetilde{O}$ 中以 $\cap [t]|\sqrt{n})$ 确定性时间计算出 X 的一个带权（t、$\widetilde{O}(t/\sqrt{b} + bu)$）-二分器。

证明。令 $\delta = 1/\sqrt{b}$。对于每个 $\ell \in [\lceil \log_{1+\delta} u \rceil]$，令 $X^{(\ell)} = X \cap [(1+\delta)^\ell, (1+\delta)^{\ell+1})$。将引理 3.6 应用于输入集 $X^{(\ell)}$（将 u 替换为 $(1+\delta)^{\ell+1}$），以在 $\widetilde{O}(b^{O(1)}|\Sigma(X) \cap [t]|\sqrt{n})$ 时间内构造一个由 $\widetilde{O}(b|\Sigma(X) \cap [t]|)$ 个规范子集组成的集合 $\mathcal{C}^{(\ell)}$。令 $\mathcal{C} = \bigcup_\ell \mathcal{C}^{(\ell)}$。然后将引理 2.3 应用于该集合 $\mathcal{C}$，其中为 $k = (1+\delta)b$，以在 $\widetilde{O}(b^{O(1)}|\Sigma(X) \cap [t]|)$ 时间内计算出一个划分 (X', X'')。

为了证明 (X', X'') 满足所需的性质，考虑任意子集 $S \subseteq X$，满足 $\sigma(S) \le t$。令 $\hat{S}^{(\ell)} \subseteq X^{(\ell)}$ 满足 $\sigma(\hat{S}^{(\ell)}) = \sigma(S \cap X^{(\ell)})$，且 $\hat{S}^{(\ell)}$ 可表示为

$O((|S\cap X^{(\ell)}|/b+1)\log\ n)$ 个互不相交的规范子集 $S_j^{(\ell)}\in\mathcal{C}^{(\ell)}$ 的并集。对于每个这样的子集 S_j，我们有 $|S_j^{(\ell)}\cap X'|,|S_j^{(\ell)}\cap X''|\le|S_j^{(\ell)}|/2+\widetilde{O}(\sqrt{b})$。对 $S_j^{(\ell)}$ 求和可得

$$\begin{aligned}\max\{|\hat{S}^{(\ell)}\cap X'|,|\hat{S}^{(\ell)}\cap X''|\} &\le |\hat{S}^{(\ell)}|/2+\widetilde{O}((|S\cap X^{(\ell)}|/b+1)\cdot\sqrt{b})\\ &= |\hat{S}^{(\ell)}|/2+\widetilde{O}(|S\cap X^{(\ell)}|/\sqrt{b}+\sqrt{b}).\end{aligned}$$

令 $\hat{S}=\bigcup_\ell\hat{S}^{(\ell)}$。则 $\sigma(\hat{S})=\sigma(S)$。现在，$\frac{1}{1+\delta}\sigma(\hat{S}\cap X')\le\sum_\ell(1+\delta)^\ell|\hat{S}^{(\ell)}\cap X'|$，且 $\frac{1}{1+\delta}\sigma(\hat{S}\cap X'')\le\sum_\ell(1+\delta)^\ell|\hat{S}^{(\ell)}\cap X''|$。因此，

$$\begin{aligned}\max\{\sigma(\hat{S}\cap X'),\sigma(\hat{S}\cap X'')\} &\le \sum_\ell(1+\delta)^\ell\left(|\hat{S}^{(\ell)}|/2+\widetilde{O}(|S\cap X^{(\ell)}|/\sqrt{b}+\sqrt{b})\right)+O(\delta t)\\ &\le \sigma(\hat{S})/2+\widetilde{O}\left(\sigma(S)/\sqrt{b}+\sum_\ell(1+\delta)^\ell\sqrt{b}\right)+O(\delta t)\\ &= \sigma(\hat{S})/2+\widetilde{O}(t/\sqrt{b}+\sqrt{b}u/\delta+\delta t)\ =\ \sigma(\hat{S})/2+\widetilde{O}(t/\sqrt{b}+bu).\end{aligned}$$

□

3.3 将所有内容整合起来

现在我们可以对 Bringmann、Fischer 和 Nakos 的算法 [BFN25] 进行去随机化了。该算法采用多路分治，因此我们首先扩展二分器，将其划分为 r 个部分：

定义 3.8。给定一个正整数集合 X 以及数值 t 和 Δ，X 的一个加权 r路 (t,Δ)划分器是将 X 划分为 r 个互不相交的子集 $X_1,\dots,X_r$，并满足以下性质：

$\hat{S}$对于每个满足 $\sigma(S)\le t$ 的子集 $S\subseteq X$，都存在一个子集 $S\subseteq X$，满足 $\sigma(S)=\sigma(S)$，并且$\hat{\sigma}(S\cap X_j)\le\sigma(S)/r+\Delta$ 对每个 $j\in\{1,\dots,r\}$ 都成立。

推论 3.9 (快速 r路分割器构造)。给定一个由 n 个整数构成的集合 X，这些整数属于 $[u]$，以及数值 t、b、r，我们可以在确定性时间内计算一个 r 路加权$(t,\ \widetilde{O}((t/\sqrt{b}+bu)\log r))$分割器，其大小为 X，且属于 $\widetilde{O}(rb^{O(1)}|\Sigma(X)\cap[t]|\sqrt{n})$。

证明。只需对 $\log r$ 层递归地应用引理 3.7。□

定理 3.10。给定一个由 n 个整数构成的集合 X，这些整数属于 $[t]$，我们可以在确定性时间内计算 $\Sigma(X)\cap[t]$，其所需时间为 $O(|\Sigma(X)\cap[t]|\sqrt{n})$。

证明。令 $u_S=\lceil t/(r^6\log^c t)\rceil$，其中 c 为足够大的常数。令 $X_S=X\cap[u_S]$ 和 $X_L=X\cap(u_S,\ t]$。以下是对 Bringmann、Fischer 和 Nakos 的算法 [BFN25] 的重新诠释：

1. 通过应用推论 3.9，计算 X_S 的一个带权 r 路（t，t/r^2）分割器（X_1，…，X_r），耗时 $\widetilde{O}(r^{\Theta(1)}|\Sigma(X)\cap[t]|\sqrt{n})$ 确定性时间，其中将 $b=\widetilde{O}(r^4)$ 和 u 替换为 u_S（参数的选取使得 $(t/\sqrt{b}+bu_S)\log r\ll t/r^2$）。

2. 对每个 $j\in\{1,\dots,r\}$，递归计算 $A_j=\Sigma(X_j)\cap[t/r+t/r^2]$。

3. 通过应用引理 3.1 $O(r)$ 次来计算和集 $A=A_1+\cdots+A_r$，所需确定性时间为 $\widetilde{O}(r|A|)$。

4. 返回 $(A + \Sigma(X_L)) \cap [t]$，它可以通过应用引理 3.3 并结合 $k = r^6 \log^c t$ 计算，所需确定性时间为 $\widetilde{O}(r^{O(1)}|\Sigma(X) \cap [t]|\sqrt{n})$（这是因为当 $X_L \subseteq (t/(r^6 \log^c t) \ \infty)$ 时，$\Sigma(X_L) \cap [t] = \Sigma_{\leq r^6 \log^c t}(X_L) \cap [t]$）。

Bringmann、Fischer 和 Nakos [BFN25]证明了

$$|(\Sigma(X_1) \cap [t/r(1+1/r)]) + \cdots + (\Sigma(X_r) \cap [t/r(1+1/r)])| \ \leq \ |\Sigma(X_1 \cup \cdots \cup X_r) \cap [t]|^{1+1/(r-1)}$$

利用 Gyarmati、Matolcsi 和 Ruzsa [GMR10]提出的和集"次乘性"性质。因此，$|A| = |A_1 + \cdots + A_r| \leq |\Sigma(X) \cap [t]|^{1+1/(r-1)}$。Bringmann、Fischer 和 Nakos 进一步指出，$|A_1| + \cdots + |A_r| \leq |A_1 + \cdots + A_r| + r - 1$。

因此，$\mu = |\Sigma(X) \cap [t]|$ 的运行时间满足递推式

$$T(n,t,\mu) \ \leq \max_{\substack{n_1,\ldots,n_r,\mu_1,\ldots,\mu_r: \\ n_1+\cdots+n_r \leq n, \\ \mu_1+\cdots+\mu_r \leq \mu^{1+1/(r-1)}+r-1}} \Big(T(n_1, t/r + t/r^2, \mu_1) + \cdots + T(n_r, t/r + t/r^2, \mu_r) + \widetilde{O}(r^{O(1)} \cdot (\mu^{1+1/(r-1)} + \mu\sqrt{n})) \Big).$$

取 $r = \log^2 t + 1$，我们有 $\mu^{1/(r-1)} \leq 2^{\log \mu / \log^2 t} \leq 2^{1/\log t} = 1 + O(1/\log t)$，且递推式可解得 $T(n, t, \mu) = O(\mu\sqrt{n})$。 □

最后，我们提到，Bringmann 和 Nakos [BN20]提出的另一种输出敏感子集和算法也可以用类似的方法去随机化。他们描述了一种关于截断和集输出敏感构造的替代方案（比引理 3.2 更复杂），其运行时间为 $\widetilde{O}(|C|^{4/3})$（如果使用引理 3.1 的确定性版本，他们的构造就是确定性的）。替换引理 3.2，保持算法其余部分不变，并对上述分析作简单修改，我们立即得到：

定理 3.11. 给定 $[t]$ 中的 n 个整数构成的集合 X，我们可以在 $\widetilde{O}(|\Sigma(X) \cap [t]|^{4/3})$确定性时间内计算 $\Sigma(X) \cap [t]$。

4 备注

在第 2 节中运行时间为 $O(t)$ 的算法里，我们没有尝试优化（因此也没有写出）$\log t$ 因子的数量。它比之前的随机化算法更多（Bringmann 的算法 [Bri17]有 4 个 log 因子，而 Jin 和 Wu 的算法 [JW19]有 2 个），但也不算特别多（大概不到 20 个）。

对于第 3 节中的输出敏感算法，我们再次强调，隐藏的额外因子是关于 n 和 t 的多对数因子。Bringmann、Fischer 和 Nakos [BFN25] 之前提出的随机化算法实际上得到的是关于输出大小 $|\Sigma(X) \cap [t]|$ 的多对数因子，而非关于 t 的因子；从去随机化的角度来看，这似乎更难实现（不过，除了一些极端情况外，影响并不大）。

与先前的工作一样，我们的算法很容易适用于 X 是多重集的情形——子集 X 中某个元素的使用次数不能超过其原始重数。（请注意，此处 n 可能大于 t；为避免运行时间中出现 $O(n)$ 项，我们可能需要假设 X 表示为一个由不同元素及其给定重数组成的列表。）

在 $O(t)$ 时间算法在第 2 节中，我们通过调整 Koiliaris 和 Xu 算法 [KX19]，来生成规范子集；但在第 3 节中，我们意识到 Koiliaris 和 Xu 算法对于我们的技术并非必不可少（尽管它很方便，因为其中已经包含了二进区间）。这一认识可能会对未来的应用有所帮助。

我们希望我们的技术能够在该领域的其他算法去随机化中找到更多应用，因为先前有一些算法使用了类似于二分器的随机划分——仅举两个例子：Chen、Lian、Mao 和 Zhang 的 $\widetilde{O}(n+\sqrt{ut})$ 时间子集和算法 [CLMZ24b]（其中 $u = \max(X)$），以及 Bringmann、Dürr 和 Polak 的背包算法 [BDP24]。我们将这些算法的去随机化问题留待未来研究（这些算法还有其他组件，可能会给去随机化带来额外挑战）。

致谢。感谢 Karl Bringmann 提出该技术的潜在应用，包括对输出敏感型子集和算法进行去随机化。

参考文献

[ABB+21] Kyriakos Axiotis、Arturs Backurs、Karl Bringmann、Ce Jin、Vasileios Nakos、Christos Tzamos 和 Hongxun Wu。快速而简洁的模子集和。发表于第 *4* 届算法简洁性研讨会（*SOSA*）论文集，第 57–67 页，2021 年。doi:10.1137/1.9781611976496.6。[ABHS22] Amir Abboud、Karl Bringmann、Danny Hermelin 和 Dvir Shabtay。子集和与双标准路径基于 SETH 的下界。*ACM* 算法汇刊，18(1):6:1–6:22，2022 年。doi:10.1145/3450524。[ABJ+19] Kyriakos Axiotis、Arturs Backurs、Ce Jin、Christos Tzamos 和 Hongxun Wu。利用线性草图实现快速模子集和。发表于第 *30* 届 *ACM–SIAM* 离散算法研讨会（*SODA*）论文集，第 58–69 页，2019 年。doi:10.1137/1.9781611975482.4。[AN96] Noga Alon 和 Moni Naor。去随机化、布尔矩阵乘法的见证以及完美哈希函数的构造。*Algorithmica*，16(4/5):434–449，1996 年。doi:10.1007/BF01940874。[AYZ95] Noga Alon、Raphael Yuster 和 Uri Zwick。颜色编码。*J.ACM*，42(4):844–856，1995 年。doi: 10.1145/210332.210337。[BCD+14] David Bremner、Timothy M. Chan、Erik D. Demaine、Jeff Erickson、Ferran Hurtado、John Iacono、Stefan Langerman、Mihai P˘atra¸scu 和 Perouz Taslakian。项链、卷积和 $X+Y$。*Algorithmica*，69(2):294–314，2014 年。doi:10.1007/S00453-012-9734-3。[BCKM24] Tatiana Belova、Nikolai Chukhin、Alexander S. Kulikov 和 Ivan Mihajlin。改进的子集和空间界。发表于第 *32* 届欧洲算法研讨会（*ESA*）论文集，第 21:1–21:17 页，2024 年。doi:10.4230/LIPICS.ESA.2024.21。[BDP24] Karl Bringmann、Anita Dürr 和 Adam Polak。通过矩形单调最小值加卷积和平衡，实现更快的背包算法。发表于第 *32* 届欧洲算法研讨会（*ESA*）论文集，第 33:1–33:15 页，2024 年。doi:10.4230/LIPICS.ESA.2024.33。[Bel57] Richard E. Bellman。动态规划。普林斯顿大学出版社，1957 年。[BFN21] Karl Bringmann、Nick Fischer 和 Vasileios Nakos。稀疏非负卷积等价于稠密非负卷积。发表于第 *53* 届 *ACM* 计算理论研讨会（*STOC*）论文集，第 1711–1724 页，2021 年。doi:10.1145/3406325.3451090。[BFN22] Karl Bringmann、Nick Fischer 和 Vasileios Nakos。稀疏非负卷积的确定性算法与拉斯维加斯算法。发表于第 *33* 届 *ACM–SIAM* 离散算法研讨会（*SODA*）论文集，第 3069–3090 页，2022 年。doi:10.1137/1.9781611977073.119。[BFN25] Karl Bringmann、Nick Fischer 和 Vasileios Nakos。超越 Bellman 算法求解子集和。发表于第 *36* 届 *ACM–SIAM* 离散算法研讨会（*SODA*）论文集，第 4596–4612 页，2025 年。doi:10.1137/1.9781611978322.157。

[BN20] Karl Bringmann 和 Vasileios Nakos。Top-k-卷积与近线性输出敏感子集和的探索。发表于第 52 届 ACM 计算理论研讨会（STOC）论文集，第 982–995 页，2020 年。doi:10.1145/3357713.3384308。[Bri17] Karl Bringmann。一个用于子集和的近线性伪多项式时间算法。发表于第 28 届 ACM–SIAM 离散算法研讨会（SODA）论文集，第 1073–1084 页，2017 年。doi:10.1137/ 1.9781611974782.69。[BW21] Karl Bringmann 和 Philip Wellnitz。关于稠密子集和的近线性时间算法。发表于第 32 届 ACM–SIAM 离散算法研讨会（SODA）论文集，第 1777–1796 页，2021 年。doi:10. 1137/1.9781611976465.107。[CDL+16] Marek Cygan、Holger Dell、Daniel Lokshtanov、Dániel Marx、Jesper Nederlof、Yoshio Okamoto、Ramamohan Paturi、Saket Saurabh 和 Magnus Wahlström。关于与合取范式可满足性同样困难的问题。ACM 算法汇刊，12(3):41:1–41:24，2016 年。doi:10.1145/2925416。[CH02] Richard Cole 和 Ramesh Hariharan。稀疏匹配与通配符匹配中的候选匹配验证。发表于第 34 届 ACM 计算理论研讨会（STOC）论文集，第 592–601 页，2002 年。doi: 10.1145/509907.509992。[CH20] Timothy M. Chan 和 Qizheng He。将 3SUM 归约为卷积 3SUM。发表于第 3 届算法简洁性研讨会（SOSA）论文集，第 1–7 页，2020 年。doi:10.1137/1.9781611976014.1。[Cha01] Bernard Chazelle。差异方法：随机性与复杂性。剑桥大学出版社，2001 年。网址：https://www.cs.princeton.edu/~chazelle/book.html。[Cha18] Timothy M. Chan。低维线性规划的改进确定性算法。ACM 算法汇刊，14(3):30:1–30:10，2018 年。doi:10.1145/3155312。[CI21] Jean Cardinal 和 John Iacono。模子集和、动态字符串与零和集。发表于第 4 届算法简洁性研讨会（SOSA）论文集，第 45–56 页，2021 年。doi:10.1137/1. 9781611976496.5。[CL15] Timothy M. Chan 和 Moshe Lewenstein。通过加法组合学求解聚类整数 3SUM。发表于第 47 届 ACM 计算理论研讨会（STOC）论文集，第 31–40 页，2015 年。doi: 10.1145/2746539.2746568。[CLMZ24a] Lin Chen、Jiayi Lian、Yuchen Mao 和 Guochuan Zhang。近线性时间近似划分问题。发表于第 56 届 ACM 计算理论研讨会（STOC）论文集，第 307–318 页，2024 年。doi: 10.1145/3618260.3649727。[CLMZ24b] Lin Chen、Jiayi Lian、Yuchen Mao 和 Guochuan Zhang。改进的子集和伪多项式时间算法。发表于第 65 届 IEEE 计算机科学基础研讨会（FOCS）论文集，第 2202–2216 页，2024 年。doi:10.1109/FOCS61266.2024.00129。[CMWW19] Marek Cygan、Marcin Mucha、Karol W.egrzycki 和 Michał Włodarczyk。关于等价于 (min +)-卷积的问题。ACM 算法汇刊，15(1):14:1–14:25，2019 年。doi:10.1145/3293465。[CW21] Timothy M. Chan 和 R. Ryan Williams。确定性全点对最短路径、正交向量等：快速去随机化 Razborov–Smolensky。ACM 算法汇刊，17(1):2:1–2:14，2021 年。doi:10. 1145/3402926。[CX24] Timothy M. Chan 和 Yinzhan Xu。更简洁的精确三角形归约。发表于第 7 届算法简洁性研讨会（SOSA）论文集，第 28–38 页。SIAM，2024 年。doi:10.1137/1.9781611977936. 4。[FKP24] Nick Fischer、Piotr Kaliciak 和 Adam Polak。确定性 3SUM 困难性。发表于第 15 届理论计算机科学前沿会议（ITCS）论文集，第 49:1–49:24 页，2024 年。doi:10.4230/ LIPICS.ITCS.2024.49。

[GMR10] Katalin Gyarmati、Máté Matolcsi 和 Imre Z. Ruzsa。和集基数的超可加性与次乘性性质。Comb.，30(2):163–174，2010。doi:10.1007/S00493-010-2413-6。[HS74] Ellis Horowitz 和 Sartaj Sahni。计算划分及其在背包问题中的应用。J. ACM，21(2):277–292，1974。doi:10.1145/321812.321823。[JVW21] Ce Jin、Nikhil Vyas 和 Ryan Williams。子集和的快速低空间算法。载于：第 32 届 ACM–SIAM 离散算法研讨会（SODA）论文集，第 1757–1776 页，2021。doi:10.1137/ 1.9781611976465.106。[JW19] Ce Jin 和 Hongxun Wu。一种简单的近线性伪多项式时间子集和随机化算法。载于：第 2 届算法简洁性研讨会（SOSA）论文集，第 17:1–17:6 页，2019。doi:10.4230/OASICS.SOSA.2019.17。[JX24] Ce Jin 和 Yinzhan Xu。借助大筛不等式消除对数因子：更快的稀疏卷积算法及其他应用。载于：第 56 届 ACM 计算理论年度研讨会（STOC）论文集，第 1573–1584 页，2024。doi:10.1145/3618260.3649605。[KMPS03] Hans Kellerer、Renata Mansini、Ulrich Pferschy 和 Maria Grazia Speranza。一种用于子集和问题的高效全多项式近似方案。计算机与系统科学杂志，66(2):349–370，2003。doi:10.1016/S0022-0000(03)00006-0。[KPS17] Marvin Künnemann、Ramamohan Paturi 和 Stefan Schneider。一维动态规划的细粒度复杂性。载于：第 44 届自动机、语言与编程国际研讨会（ICALP）论文集，第 21:1–21:15 页，2017。doi:10.4230/LIPICS.ICALP.2017.21。[KX18] Konstantinos Koiliaris 和 Chao Xu。让子集和变得简单。CoRR，abs/1807.08248，2018。arXiv: 1807.08248。[KX19] Konstantinos Koiliaris 和 Chao Xu。更快的子集和伪多项式时间算法。ACM 算法汇刊，15(3):40:1–40:20，2019。预备版本发表于 SODA 2017。doi:10.1145/ 3329863。[LN10] Daniel Lokshtanov 和 Jesper Nederlof。通过代数化节省空间。载于：第 42 届 ACM 计算理论研讨会（STOC）论文集，第 321–330 页，2010。doi:10.1145/1806689.1806735。[Mat00] Jirí Matoušek。计算几何中的去随机化。载于 Jörg-Rüdiger Sack 和 Jorge Urrutia 编，《计算几何手册》，第 559–595 页。North Holland / Elsevier，2000。doi: 10.1016/B978-044482537-7/50014-0。[NW21] Jesper Nederlof 和 Karol W.egrzycki。利用正交向量改进Schroeppel和Shamir算法，以求解子集和问题。载于：第 53 届 ACM 计算理论年度研讨会（STOC）论文集，第 1670–1683 页，2021。doi:10.1145/3406325.3451024。[PRW21] Adam Polak、Lars Rohwedder 和 Karol W.egrzycki。含小物品的背包问题与子集和问题。载于：第 48 届自动机、语言与编程国际研讨会（ICALP）论文集，第 106:1–106:19 页，2021。doi:10.4230/LIPICS.ICALP.2021.106。

将 0-1 背包问题归约为 Min-Plus 卷积

在所有容量 版本的*0-1* 背包问题中，给定 n 个物品，其中第 j 个物品的重量 w_j 为正整数，收益p_j为正实数，另给定一个数 t。对于每个 $i \in [t]$，我们要计算 $f[i] = \max\left\{\sum_{j\in J} p_j \colon \sum_{j\in J} w_j = i,\ J \subseteq \{1,\dots,n\}\right\}$。

0-1 背包问题可以看作子集和的二维变体。更准确地说，我们将每个物品映射为二维空间中的一个点 (w_j, p_j)。对于一个二维点集合 $A \subseteq [t] \times \mathbb{R}^+$，定义

算子 P-MAX(A) $= \left\{(w,\ p^*) \in A\colon p^* = \max_{(w,p)\in A} p\right\}$。于是，该问题等价于：给定一个 n 二维点集合 $X \subset [t]\times\mathbb{R}^+$，计算 P-MAX($\Sigma(X) \cap ([t]\times\mathbb{R}^+)$)，其中 $\Sigma(\cdot)$ 的定义与之前完全相同（现在的加法在二维空间中进行）。我们可以假设，经过初始的 $O(n)$ 时间预处理（将 X 替换为 P-MAX($X \cap ([t]\times\mathbb{R}^+)$）之后，$n \le t$

Cygan、Mucha、W.egrzycki 和 Włodarczyk [CMWW19]，以及 Künnemann、Paturi 和 Schneider [KPS17]，给出了从最小加卷积到全目标 0-1 背包问题的确定性归约。在反方向上，Cygan 等人 [CMWW19]通过修改 Bringmann 的子集和随机化算法 [Bri17]，给出了从全目标 0-1 背包问题到最小加卷积的<style id='8'>随机化归约。通过修改第 2 节中的算法，我们可以获得从全目标 0-1 背包问题到最小加卷积的确定性归约。

这种修改大多是直接的。下面说明其中一些主要变化：

- 诸如 $X \cap I$ 的表达式应重新解释为 $X \cap (I\times\mathbb{R}^+)$。
- 大多数 $\Sigma_i(X)$ 的出现都应改为 P-MAX($\Sigma_i(X)$)。
- 每个和集 $A+B$ 都应改为 P-MAX($A+B$)。给定 $A,\ B \subseteq [t]\times\mathbb{R}^+$，计算 P-MAX($A+B$) 等价于对两个长度为 $O(t)$ 的实向量进行最大加卷积，其 n h 通过取负即可等价地转化为最小加卷积。
- 在关于二分器的定义 2.2 中，性质现在为：对于每个满足 $|S| \le k$ 且 $\sigma(S) \in$ P-MAX($\Sigma_{\le k}(X)$) 的子集 $S \subseteq X$，都存在一个子集 $\hat{}S \subseteq X$，满足 $|\hat{}S| \le k$、$\sigma(\hat{}S) = \sigma(S)$，并且 $|\hat{}S \cap X'|,\ |\hat{}S \cap X''| \le |\hat{}S|/2 + \Delta$。
- 关于典范子集生成的引理 2.4 中，性质现在为：

  $\hat{}$ 对于每个满足 $\sigma(S) \in$ P-MAX($\Sigma_{|S|}(X)$) 的子集 $S \subseteq X$，都存在一个子集 $S \subseteq X$，使得 $\hat{}\hat{}\hat{}\ |S| = |S|$、$\sigma(S) = \sigma(S)$，并且 S 可表示为 O 个互不相交的规范子集在 $\mathcal{C}$ 中的并，其中 O=(($|S|/b+1$) log u)。注意，如果 $\sigma(S) \in$ P-MAX($\Sigma_{|S|}(X)$)，那么对于任意区间 I，我们也必须有 $\sigma(S \cap (I\times\mathbb{R}^+))$ $\in$ P-MAX$_{|S\cap(I\times\mathbb{R}^+)|}(X \cap (I\times\mathbb{R}^+))$.
- 在引理 2.6 中，我们现在要计算一个集合 OUT $\subseteq [ku]\times\mathbb{R}^+$，使得 P-MAX($\Sigma_{\le k}(X)$) $\subseteq \{(w,\ p')$: $p' \ge p,\ (w,\ p) \in$ OUT$\}$ 和 OUT $\subseteq \Sigma(X)$。

其余修改很容易推导出来。

定理 A.1。 如果长度为 n 的两个实向量的最小加卷积可以在 $T(n)$ 确定性时间内计算出来，那么 *0-1* 背包问题的全容量版本可以在 $O(T(t))$ 确定性时间内求解。