Knapsack with Small Items*

Lian Jiayi [1]
(joint work with Chen Lin [2], Mao Yuchen [1] and Zhang Guochuan [1])

[1]Zhejiang University

[2]Texas Tech University

April 6, 2024

*Faster Algorithms for Bounded Knapsack and Bounded Subset Sum Via Fine-Grained Proximity Results (SODA 2024)

The Knapsack Problem

- A knapsack of capacity T
- Set I: n items with weights $\{w_i\}_{i\in[n]}$ and profit $\{p_i\}_{i\in[n]}$
- T, w_i, and p_i are all integers
- Maximize the total profit s.t. the capacity constraint
- NP-hard, solvable in O(nT) time [Bellman '57]
- Pseudo-polynomial time complexity in other parameters?
  - the number of items n
  - the maximum weight of items $w = \max\{w_i\}_{i\in[n]}$

Current Results

$O(nT)$ $O(n^3 \cdot w^2)$	[Bellman '57] [Tamir '09]
$\widetilde{O}(n + w \cdot T)$	[Kellerer& Pferschy '04] [Bateni et al. '18] [Axiotis & Tzamos '19]
$\widetilde{O}(n \cdot w^2 \cdot \min\{n, w\})$	[Bateni et al. '18]
$\widetilde{O}(n \cdot w^2)$ $\widetilde{O}(n + w^3)$	[Eisenbrand & Weismantel '19] [Axiotis & Tzamos '19] [Polak et al. '21]

Conditional Lower Bound $(n + w)^{2-o(1)}$	[Kunnemann et al. '17] [Tygan et al. '19]

Current Results

$O(nT)$ $O(n^3 \cdot w^2)$	[Bellman '57] [Tamir '09]
$\widetilde{O}(n + w \cdot T)$	[Kellerer& Pferschy '04] [Bateni et al. '18] [Axiotis & Tzamos '19]
$\widetilde{O}(n \cdot w^2 \cdot \min\{n, w\})$	[Bateni et al. '18]
$\widetilde{O}(n \cdot w^2)$ $\widetilde{O}(n + w^3)$	[Eisenbrand & Weismantel '19] [Axiotis & Tzamos '19] [Polak et al. '21]
$\widetilde{O}(n + w^{2.4})$	Our Work
Conditional Lower Bound $(n + w)^{2-o(1)}$	[Kunnemann et al. '17] [Tygan et al. '19]

Current Results

$O(nT)$ $O(n^3 \cdot w^2)$	[Bellman '57] [Tamir '09]
$\widetilde{O}(n + w \cdot T)$	[Kellerer& Pferschy '04] [Bateni et al. '18] [Axiotis & Tzamos '19]
$\widetilde{O}(n \cdot w^2 \cdot \min\{n, w\})$	[Bateni et al. '18]
$\widetilde{O}(n \cdot w^2)$ $\widetilde{O}(n + w^3)$	[Eisenbrand & Weismantel '19] [Axiotis & Tzamos '19] [Polak et al. '21]
$\widetilde{O}(n + w^{2.4})$ $\widetilde{O}(n + w^2)$	Our Work [Jin '23], [Bringmann '23]
Conditional Lower Bound $(n + w)^{2-o(1)}$	[Kunnemann et al. '17] [Tygan et al. '19]

Dynamic Programming in O(nT)

- Let I' be a set of items. For $x =\in \{0, ..., T\}$, let

$$f_{I'}[x] = \max\{\sum_{i \in I'} p_i y_i : \sum_{i \in I'} w_i y_i \leqslant x, y_i \in \{0, 1\}\}.$$

- Goal: $f_I[T]$

Dynamic Programming in O(nT)

- Let I' be a set of items. For $x = \in \{0, ..., T\}$, let

$$f_{I'}[x] = \max\{\sum_{i \in I'} p_i y_i : \sum_{i \in I'} w_i y_i \leqslant x, y_i \in \{0, 1\}\}.$$

- Goal: $f_I[T]$
- Algorithm:
  1. I_i be the first i-th items,
  2. for $i = 1, \ldots, n$: compute

  $$f_{I_i}[x] = \max\{f_{I_{i-1}}[x], f_{I_{i-1}}[x - w_i] + p_i\}$$

     for $x = 0, \ldots, T$. — $O(T)$ time,
  3. return $f_I[T]$.
- If we can add items group by group?

$(\max, +)$-Convolution

- Let I_1 and I_2 be two disjoint sets of items. Then $f_{I_1 \cup I_2} = f_{I_1} \oplus f_{I_2}$ where

$$f_{I_1} \oplus f_{I_2}[x] = \max_{x' \in [0,x]} (f_{I_1}[x'] + f_{I_2}[x - x']).$$

- Let $I = I_1 \cup \ldots \cup I_m$,

$$f_I = f_{I_1} \oplus \ldots \oplus f_{I_m}.$$

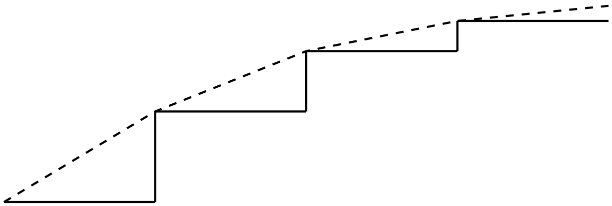
- Computing $f_{I_1} \oplus f_{I_2}$ requires $O(T^2)$ time in general.

Faster Convolution

Lemma [SMAWK algorithm '87]

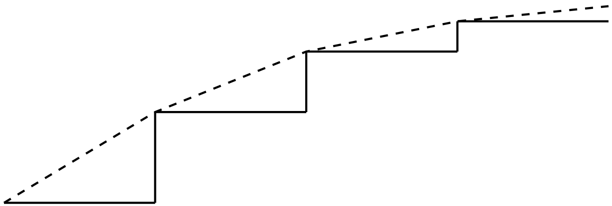
If f_{I_2} is k-step concave for some k, then $f_{I_1} \oplus f_{I_2}$ can be computed in $O(T)$ time.

- $f[ik] - f[(i-1)k] \geq f[(i+1)k] - f[ik]$ for all i,
- for all j such that $j \pmod{k} \neq 0$, $f[j] = f[j-1]$.



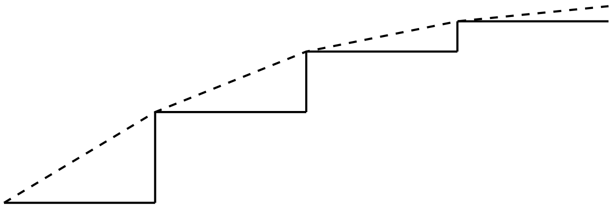
An $\widetilde{O}(n + wT)$-time Algorithm

- Partition the items by their weights
- f for each group is w-step concave for their weight w



An $\widetilde{O}(n + wT)$-time Algorithm

- Partition the items by their weights
- f for each group is w-step concave for their weight w
- Algorithm:
  1. Partition the items by their weights — $O(n)$ time
  2. Compute f for each group — $\widetilde{O}(n)$ time
  3. Merge all f's one by one using convolution — $O(wT)$ time

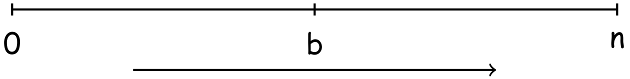


Convolution with Hint

- Let $I_1, \ldots, I_w$ be the partition of I by item weight.
- Our goal is $f_I[T]$ rather than the whole function f_I.
- $f_I[T] = f_{I_1}[t_1] + \cdots + f_{I_w}[t_w]$ for some $t_1 + \ldots + t_w = t$.
- We can not know $\{t_1, \ldots, t_w\}$ exactly.
- Some hint on $\{t_1, \ldots, t_w\}$ may help to accelerate the convolution.

A Proximity Result

- Assume that $p_1/w_1 \geqslant p_2/w_2 \geqslant \cdots \geqslant p_n/w_n$.
- Let $x' = \{1, \ldots, 1, 0, \ldots, 0\}$ be the maximal prefix solution. (greedily select the item with the highest efficiency)



larger index has lower efficiency

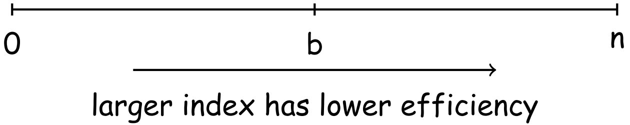
A Proximity Result

- Assume that $p_1/w_1 \geqslant p_2/w_2 \geqslant \cdots \geqslant p_n/w_n$.
- Let $x' = \{1, \ldots, 1, 0, \ldots, 0\}$ be the maximal prefix solution. (greedily select the item with the highest efficiency)

Lemma [Eisenbrand & Weismantel '19][Polak et al. '21]

There is an optimal solution x^* such that

$$\sum_{i \in I} |x_i^* - x_i'| \leqslant 2w.$$



An $\widetilde{O}(n + w^3)$ Algorithm

- There exists $\{g_1, \ldots, g_w\}$ such that

$$\sum_{j \in \{1,\ldots,w\}} |t_j - g_j| \leqslant 2w^2.$$

- When computing $f_{I_1 \cup \cdots \cup I_j} \oplus f_{I_{j+1}}$, it suffices to consider their subsequences of length at most $4w^2$.
  Each convolution needs $O(w^2)$ time. [Polak et al. '21]
- we can compute $f_I[T]$ in $\widetilde{O}(n + w^3)$ time.

A Stronger Hint

- Previous hint:

$$\sum_{j \in \{1,\ldots,w\}} |t_j - g_j| \leqslant 2w^2.$$

- It is impossible that for any $j \in \{1, \ldots, w\}$,

$$|t_j - g_j| = \Theta(w^2).$$

- There exists a subset D of $\{1, \ldots, w\}$ that $|D| = \widetilde{O}(w^{0.5})$ and

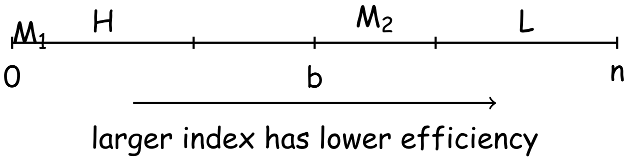
$$\sum_{j \in \overline{D}} |t_j - g_j| \leqslant O(w^{1.5}).$$

An $\widetilde{O}(n + w^{5/2})$-Time Algorithm

- There exists a subset D of $\{1, \ldots, w\}$ that $D = \widetilde{O}(w^{0.5})$ and
$$\sum_{j \in \overline{D}} |t_j - g_j| \leqslant O(w^{1.5}).$$
- Only $\widetilde{O}(w^{0.5})$ out of the w convolutions require $O(w^2)$ time for each.
- The other convolutions require $O(w^{1.5})$ time for each.
- The total running time is
$$\widetilde{O}(n + w^{0.5} \cdot w^2 + w \cdot w^{1.5}) = \widetilde{O}(n + w^{2.5})$$

.

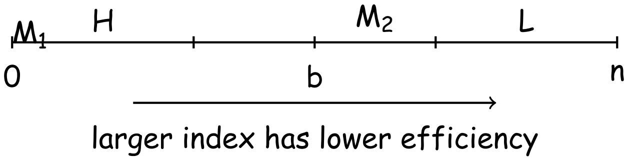
Intuition for the Stronger Hint



OPT tends to select

- a lot of items in H,
- very few item in L.

Intuition for the Stronger Hint



- If items in M_1 and M_2 has $O(w^{0.5} \log w)$ distinct weight, then
- the total weight of items in H not selected by OPT is at most $O(w^{1.5})$,
- the total weight of items in L that are selected by OPT is at most $O(w^{1.5})$.

Additive Combinatorics

A Fundamental Result [Szemeredi & Vu '06]

For any subset S of $\{1, \ldots, w\}$ with $|S| \geq O(w^{0.5} \log w)$, the subset sums of S contains an arithmetic progression of length w.

Additive Combinatorics

A Fundamental Result [Szemeredi & Vu '06]

For any subset S of $\{1, \ldots, w\}$ with $|S| \geq O(w^{0.5} \log w)$, the subset sums of S contains an arithmetic progression of length w.

It was first used to tackle Subset Sum. [Galil & Margalit '91] [Bringmann & Wellnitz '21]

Result in [Bringmann & Wellnitz '21](Informal)

If S is dense and has no $\Theta(1)$-almost divisor, then there exists λ_X such that $[\lambda_X, \Sigma_X - \lambda_X] \subseteq \mathcal{S}_X$.

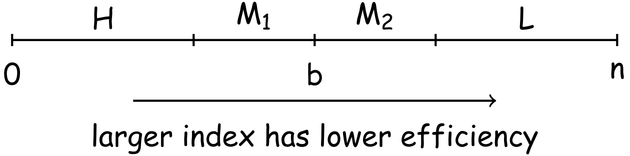
Additive Combinatorics

Using additive combinatorics tools, we can obtain

Lemma

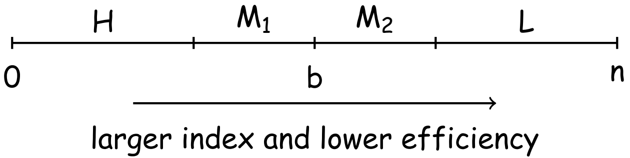
Let A and B be two subset of $\{1, \dots, w\}$. If $|A| \geq O(w^{0.5} \log w)$ and $\Sigma_{x \in B} x \geq O(w^{1.5} \log w)$, then there are non-empty subsets $A' \subseteq A$ and $B' \subseteq B$ that

$$\sum_{x \in A'} x = \sum_{x \in B'} x.$$



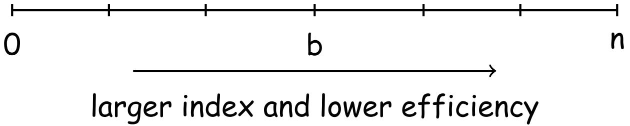
Further Improvement

- Using additive combinatorics result for multi-set.
- Partition into more groups (in terms of efficiency).



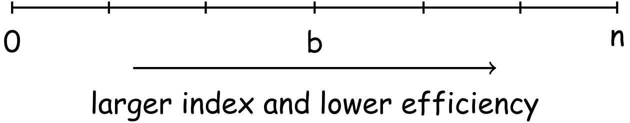
Further Improvement

- Using additive combinatorics result for multi-set.
- Partition into more groups (in terms of efficiency).
- We partition into six groups and obtain $\widetilde{O}(n + w^{2.4})$.



Further Improvement

- Using additive combinatorics result for multi-set.
- Partition into more groups (in terms of efficiency).
- We partition into six groups and obtain $\widetilde{O}(n + w^{2.4})$.
- [Bringmann '23] and [Jin '23] partition into $O(\log n)$ groups and obtain $\widetilde{O}(n + w^2)$.



Other Results

- Bounded Knapsack: an $\tilde{O}(N + w^2)$ algorithm.
- Approximation for Knapsack: an $\widetilde{O}(n + \frac{1}{\varepsilon^2})$-time FPTAS.
- Subset sum: an $\widetilde{O}(n + w^{1.5})$ algorithm.

Thank you!