2-approximating Minimum Knapsack Problem

- Changki Yun, 2023-12-17

Intro

Definition. Given n items labeled by $[n]$, and item i has value v_i and cost c_i, we define minimum knapsack problem as following optimization problem - integer programming in fact.

$$\begin{aligned} \text{minimize} \quad & \sum_i c_i x_i \\ \text{s.t.} \quad & \sum_i v_i x_i \geq D \\ & x_i \in \{0,1\} \end{aligned}$$

where D is given desire on value.

We can devise a naive LP relaxation for this:

$$\begin{aligned} \text{minimize} \quad & \sum_i c_i x_i \\ \text{s.t.} \quad & \sum_i v_i x_i \geq D \\ & x_i \in [0,1] \end{aligned}$$

while this can be as bad as D in terms of approximation ratio.
Where $v_1 = D-1, v_2 = D$ and $c_1 = 0, c_2 = D$, LP optimum is 1 where $x = (1, \frac{1}{D})$ but the IP optimum is D for $x = (0,1)$.

However, adopting additional LP constraints provides a significant improvement on performance.
Assume that for a set $A \subseteq [n]$, we added all items in A for our collection. Then, it should be satisfied in the Integer Program. They are referred to **knapsack covering inequalities**.

$$\sum_{i \notin A} \min(D - v(A), v_i) x_i \geq D - v(A)$$

Where $v(A) := \sum_{i \in A} v_i$ is sum of the value of items in A.
Now we can re-write the LP relaxation as below.

$$\begin{aligned} \text{minimize} \quad & \sum_i c_i x_i \\ \text{s.t.} \quad & \sum_{i \in [n] \setminus A} v_i^{-A} x_i \geq D - v(A) \qquad \forall A \subseteq [n] \\ & x_i \in [0,1] \end{aligned}$$

Where $v_i^{-A} := \min(v_i, D - v(A))$.
We will prove in two ways, that this LP gives **2-approximation** of Minimum Knapsack Problem, and address the way dealing with exponentially many conditions.

Simple Primal-Dual solution

The simplest way is considering the dual program.

$$\begin{aligned} \text{maximize} \quad & \sum_i (D - v(A)) y_A \\ \text{s.t.} \quad & \sum_{i \in [n] \setminus A} v_i^{-A} y_A \leq c_i \qquad \forall i \in [n] \\ & y_A \geq 0 \end{aligned}$$

Note that the Primal Complementary Slackness condition

$$x_i > 0 \implies c_i = \sum_{i \in [n] \setminus A} v_i^{-A} y_A$$

Will guide us to the derivation. We execute the dual algorithm as follows.

- Set $B := \emptyset$.
- While $v(B) < D$:
  - Increase y_B until an inequality becomes tight.
  - If an inequality for item j became tight, add j into B.
- Return B.

The algorithm maintains a valid dual solution, since once j is added into B no dual variables in constraint for item j grow up. It'll terminate in $O(n^2)$ time. Now, let's verify the performance.

Approximation ratio

Note that we set the primal / dual pair to satisfy the complementary slackness condition.
Now using the complementary slackness condition, we expand the cost term.

$$\sum_{i \in B} c_i x_i = \sum_{i \in B} x_i \sum_{i \in [n] \setminus A} v_i^{-A} y_A = \sum_{A \subseteq B} y_A \sum_{i \in B \setminus A} v_i^{-A}$$

Note that $y_A = 0$ for $A \nsubseteq B$. Specifically, y_A will be positive for only (proper) prefix set of $(a_1, \cdots, a_k)$, the trace of elements added into A.
Fix a set A, and let l be the latest element added to B. It will be always in $B \subseteq A$. Since $v(B) - v_l < D$, We can say that $v(B) - v(A) - v_l < D - v(A)$ hence for all $t \in B \setminus A \setminus l$, $v_t = v_t^{-A}$.

Combining this, we obtain
$\sum_{i \in B \setminus A} v_i^{-A} = v_l^{-A} + \sum_{i \in B \setminus A \setminus l} v_i = (v(B) - v_l - v(A)) + v_l^{-A} < 2(D - v(A))$.

Thus we can say that

$$\sum_{i \in B} c_i \leq 2 \sum_{A \subseteq B} y_A (D - v(A)) \leq 2\mathrm{OPT}.$$

Solution from rounding

Primal-dual is a constructive proof of 2-approximation. Via LP rounding, we will achieve stronger result.

Claim. For any feasible solution x of knapsack covering LP, there is a set of feasible integral solutions $x^{(1)}, \cdots, x^{(t)}$, which accepts a convex combination dominated by $2x$. i.e. there is a non-negative coefficient $\lambda^{(i)} \geq 0$ such that $\sum_{i=1}^{t} \lambda^{(i)} x^{(i)} \leq 2x$, and $\sum_i \lambda^{(i)} = 1$.

This is clearly a generalization of 2-approximation result if we set x as the optimum of knapsack covering LP. But how do we solve this LP, having exponentially many constraints?

There are some known methods as ellipsoid methods provided by poly-time separation oracle, but it is not that easy to obtain such oracle as well. Here, we delve into the **Claim** and obtain the poly-time solution using the separation-oracle based on ellipsoid method.

Proof of claim

We assume the coefficients are all integer, and x is a rational vector. Suppose that for an integer $r > 0$, rx is integral. We will set r buckets on a circle, and assign items to each bucket. Assume $A := \{i : x_i \geq 1/2\}$. For items in $[n] \setminus A$, we renumber the items to $1, \cdots, k$ so that $v_1 \geq \cdots \geq v_k$. Starting from the item 1, we distribute $2rx_i$ copies of item i to each bucket, going clockwise through the circle. Note that an item will never be put into the same bucket as $2rx_i < 2r \cdot \frac{1}{2} = r$.
Now, we claim that each bucket generates a feasible integral solution combined with A. It suffices to see the last (r-th) bucket K which will have the least value. If $\sum_{i \in K} v_i^{-A} < D - v(A)$, we will show that $\sum_{i \in F} v_i^{-A} < 2(D - v(A))$ for the first bucket F. And then, it contradicts to the fact that average value of all the bucket is

$$\frac{1}{r} \sum_{i=1}^{k} (2rx_i) v_i^{-A} = 2 \sum_{i=1}^{r} v_i^{-A} x_i \geq 2(D - v(A))$$

from the knapsack feasibility of x.

Note that $v_i^{-A} \geq v_j^{-A}$ if $i < j$.
So $\sum_{i \in F} v_i^{-A} - \sum_{i \in K} v_i^{-A}$ is sum of a non-increasing alternating sequence of which starting term is not greater than $D - v(A)$; thus also bounded by $D - v(A)$. $\square$

Some readers might've noticed that the "bucket distribution" involves only k distinct budgets, so it's able to construct such bucket in polynomial time.

Solving an exponentially sized LP

Actually, we'll restrain from solving the entire LP to obtain the optimum of knapsack covering LP. From the proof of claim, we noticed that the solution x suffices, if it satisfies knapsack-covering inequality for the set

$$A_x := \{i \in [n] : x_i \geq 1/2\}.$$

provided the cost of x is better than the optimal cost of knapsack covering LP. We'll proceed the ellipsoid algorithm as follows:

- Start from a random solution x and a large enough ellipsoid centered at x.
  - If x satisfies the knapsack covering inequality for A_x, cut the ellipsoid by the half-plane $\{y : c^T y \leq c^T x\}$.
  - If x violates, we got the separation oracle.

It's provided that the algorithm terminates in the polynomial time, finding or failed to finding an optimal (basic) point x_0 satisfying knapsack covering inequality respect to A_{x_0}.
We call a solution x a **self-covering solution** if x satisfies A_x knapsack-covering inequality.

- If the algorithm ended up failing to find any single **self-covering solution**, we conclude that the polytope is empty, since the ellipsoid must be shrunken enough.
- Otherwise, the last **self-covering solution** and the following violations guarantees that there is **no feasible solution of full knapsack-covering LP** with better cost compared to it.

Either way, we proved that our algorithm gives the desired result.
As a side note, set of self-covering solution is not necessarily convex, considering the form of constraint.

References

- Williamson, David P., and David B. Shmoys. *The design of approximation algorithms*. Cambridge university press, 2011.
  - For primal-dual approach.
- Carr, Robert D., et al. *Strengthening integrality gaps for capacitated network design and covering problems*. No. SAND99-1972C. Sandia National Lab.(SNL-NM), Albuquerque, NM (United States); Sandia National Lab.(SNL-CA), Livermore, CA (United States), 1999.
  - For rounding approach.

- Grötschel, Martin, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*. Vol. 2. Springer Science & Business Media, 2012.