

Lecture Notes on an *FPRAS* for *#KNAPSACK*.

Eric Vigoda

Georgia Institute of Technology

Last updated for 7530 - Randomized Algorithms, Spring 2010.¹

Approximately Counting KNAPSACK Solutions

In this lecture, we study the counting and sampling versions of the knapsack problem. This is a further example (as with Euclidian TSP in the last lecture) of the power of simple dynamic programming based approaches. It also highlights the intimate connection between approximate counting and random sampling.

Here is the problem statement.

#KNAPSACK: We are given n objects, together with each object $i \in [n]$ (recall $[n] = \{1, 2, \dots, n\}$) we have its *integer* weight w_i , and the total weight W our knapsack can hold. Our objective is to find the number of subsets $T \subseteq [n]$ such that $\sum_{i \in T} w_i \leq W$. We call the sets satisfying this weight constraint *feasible*, and denote them as \mathcal{S} . Thus, we are computing $|\mathcal{S}|$. A geometric interpretation of the problem is the following. Recall, the n -dimensional hypercube has vertex set $\{0, 1\}^n$ and edges connecting vertices which differ in exactly one coordinate. The constraint $\sum_i w_i \leq W$ defines a hyperplane. Thus, we are counting the number of points of the hypercube that intersect the hyperplane.

This problem is sometimes referred to as the zero-one #KNAPSACK. #KNAPSACK is known to be #P-complete, therefore we focus on approximation schemes.

We present an FPRAS for #KNAPSACK due to Dyer [1]. An *FPRAS* (fully polynomial randomized approximation scheme), is a randomized algorithm, which for all $0 < \epsilon, \delta < 1$, computes *OUT* satisfying:

$$OUT(1 - \epsilon) \leq |\mathcal{S}| \leq OUT(1 + \epsilon)$$

with probability $\geq 1 - \delta$, and running in time polynomial in n , $1/\epsilon$, and $\log(1/\delta)$.

The first FPRAS was due to Morris and Sinclair [2]. They used a more standard Markov chain Monte Carlo approach, but the analysis of their algorithm was considerably more complicated than Dyer's approach.

Dyer first presents an easy algorithm to approximate $|\mathcal{S}|$ within a polynomial in n factor. Using this sampler, it is then straightforward to estimate $|\mathcal{S}|$ arbitrarily close.

First we present a pseudo-polynomial solution for #KNAPSACK, i.e., the running time depends on W . Let $F(j, k)$ be the number of sets $T \subseteq [j]$ such that $\sum_{i \in T} w_i \leq k$. Thus, $|\mathcal{S}| = F(n, W)$.

¹Based on scribe notes first prepared by Ivona Bezáková at the University of Chicago in the winter quarter, 2003.

Now we have an easy recurrence for $F(j, k)$. Either: we include the j -th item in the knapsack and our available weight goes from k to $k - w_j$; or we don't include the j -th item and the available weight stays the same. Hence, we have:

$$\begin{aligned} F(j, k) &= F(j-1, k) + F(j-1, k-w_j) && \text{if } j \geq 1 \\ F(0, k) &= \begin{cases} 1 & \text{if } k \geq 0 \\ 0 & \text{if } k < 0 \end{cases} \end{aligned}$$

Using dynamic programming we find $F(n, W)$ in time $O(nW)$.

We don't want the running time to depend on W . Hence, we'll rescale the weights so that W scales down to something which is polynomial in n . The scaling will round the weights w_i down to integers, thereby altering the set of feasible solutions. In fact, the rounding will increase the number of solutions, but by at most a factor of $(n+1)$.

Scaling. Without loss of generality, assume the weights are sorted: $0 \leq w_1 \leq w_2 \leq \dots \leq w_n \leq W$. Let the scaled-down weights be $w'_i := \lfloor \frac{n^2}{W} w_i \rfloor$, and the new total weight $W' := n^2$. Let \mathcal{S} be the solution set of this new instance of #KNAPSACK. Using the above dynamic programming we can compute $|\mathcal{S}'|$ in time $O(n^3)$.

Note, if $T \in \mathcal{S}$, i.e., T is a feasible set of the original problem, then

$$\sum_{i \in T} w'_i \leq \frac{n^2}{W} \sum_{i \in T} w_i \leq \frac{n^2}{W} W \leq n^2,$$

and thus T is also solution to the new instance. Therefore, $\mathcal{S} \subseteq \mathcal{S}'$.

Comparing $|\mathcal{S}'$ with $|\mathcal{S}|$. We have just shown $|\mathcal{S}| \leq |\mathcal{S}'|$. We will now define a map $f: \mathcal{S}' \rightarrow \mathcal{S}$ such that for every $T \in \mathcal{S}$ there will be at most $(n+1)$ sets in \mathcal{S}' mapping to it. This implies $|\mathcal{S}'| \leq (n+1)|\mathcal{S}|$. Therefore in $O(n^3)$ time we can compute $|\mathcal{S}'|$ which satisfies:

$$\frac{|\mathcal{S}'|}{n+1} \leq |\mathcal{S}| \leq |\mathcal{S}'| \tag{1}$$

It remains to define f . For $T \in \mathcal{S}$, let $f(T) := T$.

Consider $T \in \mathcal{S}' \setminus \mathcal{S}$. Let K be the largest integer k such that $w_k \leq W/n$. Note, any $T' \subseteq [K]$ is a solution to the original problem (i.e. $T' \in \mathcal{S}$). To see why, observe that for such a $T' \subseteq [K]$ we have

$$\sum_{i \in T'} w_i \leq \sum_i \frac{W}{n} \leq W.$$

Hence, $T' \in \mathcal{S}$. Therefore, for $T \in \mathcal{S}' \setminus \mathcal{S}$, we know $T \not\subseteq [K]$. Let ℓ be the index of the heaviest element in T . Note, $\ell > K$. Then we define $f(T) := T \setminus \{\ell\}$. We need to show that $f(T) \in \mathcal{S}$. To simplify notation, let

$$\delta_i := w_i \frac{n^2}{W} - w'_i,$$

denote the rounding error. Thus,

$$w_i = \frac{W}{n^2} (w'_i + \delta_i),$$

and $0 \leq \delta_i \leq 1$. Finally, we need to show that $f(T) \in \mathcal{S}$. We have:

$$\begin{aligned} \sum_{i \in f(T)} w_i &= \frac{W}{n^2} \sum_{i \in f(T)} (w'_i + \delta_i) \\ &= \frac{W}{n^2} \left(\left(\sum_{i \in T} w_i \right) - w'_\ell + \left(\sum_{i \in T} \delta_i \right) - \delta_\ell \right) \\ &\leq \frac{W}{n^2} \left(\left(\sum_{i \in T} w'_i \right) + n \right) - w_\ell \\ &\leq \frac{W}{n^2} \sum_{i \in T} w'_i \quad \text{since } w_\ell > W/n \\ &\leq \frac{W}{n^2} (n^2) \quad \text{since } T \in \mathcal{S}' \\ &= W, \end{aligned}$$

Therefore, $f(T) \in \mathcal{S}$ and f is properly defined from \mathcal{S}' to \mathcal{S} .

How many sets map to a given set $T \in \mathcal{S}$? Let ℓ be the index of the heaviest element in T . Then, by the definition of f , the set T maps to itself via the first rule defining f , and at most $n - \ell$ sets map to T via the second rule. This sums to at most $(n + 1)$ sets mapping to any $T \in \mathcal{S}$. This proves (1).

To improve the approximation factor, we will use the above approach to sample uniformly from \mathcal{S}' , and then we will use these samples to closely estimate $|\mathcal{S}|$ by looking at the fraction of the samples that lie in \mathcal{S} .

Sampling. We now use the above dynamic programming approach to sample uniformly at random from \mathcal{S}' . We will trace back a solution using the table F . We start with the empty set and we will add elements according to the probability distribution defined by F .

1. Let $T := \emptyset$, $j := n$, and $k := n^2$.
2. While $j > 0$ do:
 - (a) With probability $F(j - 1, k - w'_j)/F(j, k)$:

$$\text{Set } T = T \cup \{j\} \text{ and set } k = k - w'_j.$$

- (b) Decrease j by one.

It is easy to see that this algorithm samples sets uniformly from \mathcal{S}' . Given access to F , the above algorithm takes $O(n)$ time to generate a sample. Thus, for t samples it takes $O(n^3)$ time to compute F and then $O(tn)$ time to generate t random samples from \mathcal{S}' , therefore it takes $O(n^3 + tn)$ time in total.

Counting. Let $p := |\mathcal{S}|/|\mathcal{S}'|$. Note, we know $1 \geq p \geq 1/n$. We will use sampling to estimate p . Generate $N = \frac{4n}{\epsilon^2} \log(2/\delta)$ random sets from \mathcal{S}' . For $i = 1, \dots, N$, let

$$X_i = \begin{cases} 1 & \text{if the } i\text{-th sample is in } \mathcal{S} \\ 0 & \text{otherwise} \end{cases}$$

Let $X = \sum_{i=1}^N X_i$ and let

$$Z = |\mathcal{S}'| \frac{X}{N}.$$

Note,

$$\mu = \mathbf{E}[Z] = \frac{|\mathcal{S}'|}{N} \sum_i \mathbf{E}[X_i] = |\mathcal{S}'|p = |\mathcal{S}|.$$

By Chernoff bounds,

$$\Pr[|X - \mu| \geq \epsilon\mu] \leq 2 \exp(-\mu\epsilon^2/4) \leq \exp(-pN\epsilon^2/4) \leq 1/\delta.$$

If $X \in [\mu(1 - \epsilon), \mu(1 + \epsilon)] = pN[1 - \epsilon, 1 + \epsilon]$, then $Z \in |\mathcal{S}||[1 - \epsilon, 1 + \epsilon]$. Hence, we have an FPRAS for estimating $|\mathcal{S}|$. Note, the total running time is $O(n^3 + \epsilon^{-2}n^2 \log(1/\delta))$.

Improvements and generalizations. Using randomized rounding, Dyer improved the running time to $O\left(\left(n^{5/2}\sqrt{\log(1/\epsilon)} + n^2\epsilon^{-2}\right)\log(1/\delta)\right)$. The m -constraint #KNAPSACK has m weights for each object and m total weights. The objective is to find the number of sets satisfying all m weight constraints. Geometrically, it corresponds to finding the number of vertices in the hypercube contained in the intersection of m hyperplanes. Dyer's approach carries through, resulting in $O(n^{2m+1} + \epsilon^{-2}n^{m+1})$ running time. In the general (non 0-1) #KNAPSACK problem one wants to find the number of all feasible *multisets*. Again, Dyer's approach gives $O(n^{2m+3} + \epsilon^{-2}n^{m+3})$ time bound.

Open Problems: Is there a deterministic algorithm to estimate $|\mathcal{S}|$ arbitrarily close? The above approach computes $|\mathcal{S}'|$ deterministically, and this yields a \sqrt{n} approximation to $|\mathcal{S}|$. We only use randomness to boost the approximation factor.

References

- [1] M. Dyer. Approximate Counting by Dynamic Programming, *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, 693-699, 2003.
- [2] B. Morris and A.J. Sinclair. Random walks on truncated cubes and sampling 0-1 knapsack solutions. *SIAM J. Comput.*, 34(1):195-226, 2004.