CS264: Beyond Worst-Case Analysis
Lecture #14: Smoothed Analysis of Pareto Curves*

Tim Roughgarden†

November 5, 2014

1 Pareto Curves and a Knapsack Algorithm

Our next application of smoothed analysis is to Pareto curves. We begin by explaining one of several reasons why you should care about the size of Pareto curves: they govern the running time of some interesting algorithms.

In the *Knapsack* problem, the input comprises n items, each with a positive value v_i and a positive weight (or size) w_i. Also given is a knapsack capacity W. The goal is choose a subset $S \subseteq \{1, 2, \ldots, n\}$ of the items that maximizes the total value $\sum_{i \in S} v_i$, subject to the chosen items fitting in the knapsack (i.e., $\sum_{i \in S} w_i \leq W$). When you were first introduced to this problem, you may have been told some hokey story about a burglar trying to escape with the best loot — but it really is a fundamental problem, relevant whenever you're trying to make optimal use of a limited resource.

We say that a subset $S \subseteq \{1, 2, \ldots, n\}$ *dominates* $T \subseteq \{1, 2, \ldots, n\}$ if: (i) the total value of S is at least that of T ($\sum_{i \in S} v_i \geq \sum_{i \in T} v_i$); (ii) the total weight of S is at most that of T ($\sum_{i \in S} w_i \leq \sum_{i \in T} w_i$); and (iii) at least one of these two inequalities is strict. If S dominates T then it renders T moot — the solution T can be safely pruned without regret.

The *Pareto curve* of a Knapsack instance is the set of all undominated solutions; see Figure 1. Geometrically, a point is dominated if and only if there is another point "to the northwest" of it. You've seen this concept before, in Lecture #2, when we studied the 2D Maxima problem. The Pareto curve corresponds, after a reflection about the y-axis, to the set of maxima of the point set with one point for each subset of $\{1, 2, \ldots, n\}$, the two coordinates of a point being the total weight and total value of S.

Here is a Knapsack algorithm that you may not have seen before (due to [5]).

1. Generate the Pareto curve. If multiple solutions have identical total value and total weight, an arbitrary one of them is retained.

†Department of Computer Science, Stanford University, 474 Gates Building, 353 Serra Mall, Stanford, CA 94305. Email: tim@cs.stanford.edu.

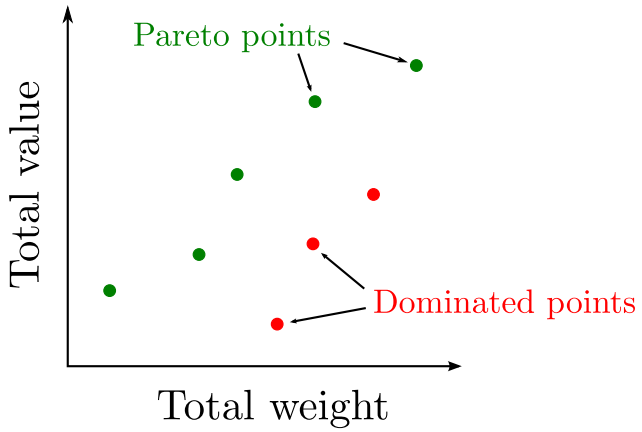

Figure 1: The Pareto curve. The red points are dominated, the green points form the Pareto curve.

2. Among all solutions in the Pareto curve with total weight at most the knapsack capacity W, return the one with the largest total value.

The first thing to notice about this algorithm is that it does not involve dynamic programming (at least not explicitly). It is clearly correct, since there is no loss of generality in restricting the search for an optimal solution to the Pareto curve (by the definition of domination).

A good way to implement the algorithm above is to order the items arbitrarily and generate a sequence of Pareto curves $PC_0, PC_1, \ldots, PC_n$, where PC_i denotes the Pareto curve for the Knapsack instance induced by the first i items. Provided each Pareto curve is represented in increasing order of total weight, PC_{i+1} can be obtained from PC_i by merging two sorted lists (PC_i and a "shifted version" of PC_i). A suitable implementation of this idea yields an algorithm running in time $O(\sum_{i=0}^{n} |PC_i|)$; see Homework #7 for details.

The good news is that the bound $O(\sum_{i=0}^{n} |PC_i|)$ can be bounded above, even in the worst case, in three different ways. Certainly $|PC_i| \leq 2^i$ for every i, so $O(\sum_{i=0}^{n} |PC_i|) = O(2^n)$. Next, suppose that all items' weights are integers between 1 and $w_{\max}$. No curve PC_i contains more than one solution with the same total weight (all but the highest-value one are dominated), so in this case $|PC_i| \leq nw_{\max}$ and hence $O(\sum_{i=0}^{n} |PC_i|) = O(n^2 w_{\max})$. Similarly, if all of the items' values are integers between 1 and $v_{\max}$, then $O(\sum_{i=0}^{n} |PC_i|) = O(n^2 v_{\max})$. Thus, this algorithm not only solves the knapsack problem without explicit dynamic programming, its running time bound of $O(\min\{2^n, nw_{\max}, nv_{\max}\})$ is at least as good as *both* of the dynamic programming algorithms that you may have seen previously for the problem. One of these classical dynamic programs has a 2-D table indexed by a number i of items and a budget $W' \leq W$ on weight; the other is indexed by i and a target value $V \leq nv_{\max}$.

The bad news is that the worst-case size of the Pareto curve of a Knapsack instance with n items can indeed be as large as 2^n (see Homework #7).

2 Smoothed Analysis of Pareto Curves

One reason to cover the above Knapsack algorithm is that it's simple, practically useful in some cases, and underrepresented in undergraduate courses and textbooks. But the main reason here is because it's particularly amenable to a smoothed analysis. The main result in this lecture is the following.

Theorem 2.1 ([1]) *In smoothed Knapsack instances (see details below), the expected size of the Pareto curve is $O(n^2/\sigma)$, where σ is a measure of "perturbation size."*

The bound in Theorem 2.1 is tight; see Homework #7.

Theorem 2.1 and linearity of expectation imply that the Knapsack algorithm in the previous section has expected running time $O(n^3/\sigma)$, which is polynomial provided σ is bounded below by an inverse polynomial function of n. This is particularly interesting in light of the fact that Knapsack is an NP-hard problem! Thus, Theorem 2.1 tells us something interesting not just about a specific Knapsack *algorithm*, but more generally about the (smoothed) complexity of the Knapsack *problem*. (Next lecture elaborates on this difference further.) That the Knapsack problem has polynomial smoothed complexity corroborates well with empirical evidence that it is among the easiest of NP-hard problems.

The perturbation model we use in Theorem 2.1 is essentially the same as last lecture. We allow the item values v_i to be arbitrarily chosen; only the item weights w_i are random.[1] We assume, without loss of generality, that the values lie in $[0, 1]$. We assume that each weight w_i is drawn independently according to a density function $f_i : [0, 1] \to [0, \frac{1}{\sigma}]$ that is supported on $[0, 1]$ and is "not too spiky," meaning that the density is never higher than $1/\sigma$. In particular, the support of the distribution has area at least σ, with equality if and only if f_i is uniform over its support. This perturbation model is even simpler than last lecture, since we've moved from two-dimensional to one-dimensional objects (from points in the plane to item weights). It is very satisfying that Theorem 2.1 makes only the minimal assumption that the distributions "have enough randomness," and makes no assumptions about their specific form. It is easy to believe that "real data" contains "a little randomness," even though any specific distributional assumption would be implausible.

3 Proof of Theorem 2.1

Our goal is to upper bound the expected value of N, the number of Pareto optimal solutions of a smoothed Knapsack instance. Since N can be exponential in the worst case (see Homework #7), we need to use the assumption that the item weights are random. It seems difficult to relate directly this hypothesis to N. For example, consider some subset S of items, say $\{4, 7, 10\}$. We know the total value of this subset (recall values are fixed), say 127. Depending on the random item weights, S might or might not be Pareto optimal. This event depends on the random weights of items 4, 7, and 10 — the smaller these are, the more likely S

[1] Alternatively, one can assume that the weights are arbitrary and that the item values are random.

is to be Pareto optimal. It also depends on other item weights, however — the larger the weights of other items (and hence other solutions), the more likely S is to be Pareto optimal. To make matters worse, there might be an exponential number of other solutions that are eligible to dominate S, and it is hard to argue about all of them in a coherent way. Even if we condition on all item weights except one, it is not clear how to argue directly about whether or not a given solution S contributes to N.

So our plan is to decompose N into simpler random variables that are easier to relate to our hypothesis that the item weights are random. To identify a role model, we recall the high-level idea of last lecture's analysis, of the 2OPT local search algorithm for TSP in the plane. We first zoomed in on a specific swap — happily, there were only $O(n^4)$ fundamentally distinct swaps to consider. Such a swap involved 8 numbers, two coordinates for each of the 4 relevant points. We then zoomed in further on one of the $(4!)^2$ relevant integer linear combinations of these 8 numbers, which corresponds to a particular relative ordering of the 4 x-coordinates and of the 4 y-coordinates. After conditioning on all but one of these numbers, the bad event boiled down simply to whether or not the remaining number fell into a particular interval. This probability was trivial to upper bound because of the upper bound on every point's density function. Returning to the present setting, we would like to identify analogously simple bad events, ideally corresponding simply to one of the smoothed random variables (i.e., item weights) falling into a particular interval.

As a first step, imagine plotting the total weight and total value of every subset $S \subseteq \{1, 2, \ldots, n\}$, in the box $[0, n] \times [0, n]$. For the sake of analysis, we imagine chopping up this box into n/ϵ vertical "slices," each of width ϵ (see Figure 2). Here $\epsilon > 0$ is a parameter internal to our analysis, chosen for convenience (it will not appear in our final bound). We choose ϵ so small (e.g., inverse doubly exponential in n) that there is essentially zero probability that more than one subset S appears in a single slice.[2] We are then justified in redefining N as the number of slices that contain a Pareto optimal solution, rather than as the number of Pareto optimal solutions.

Here's an idea that is in the spirit of what we want, but too simple to work. We could decompose $N = \sum_{\text{slices } t} N_t$, where N_t is an indicator random variable for whether or not there is a Pareto optimal solution in the slice with left boundary t (and right boundary $t + \epsilon$). Since $N \approx \sum_t \mathcal{N}_t$, linearity of expectation reduces our task from upper bounding the expectation of N to upper bounding the probability of an event $\{N_t = 1\}$. This event is still too complicated to analyze directly, though: there are potentially an exponential number of item subsets that might land in this slice, even after conditioning on all item weights except for one. Intuitively, only the subsets with the highest total values are relevant. To tease out this point, we need to refine our approach and decompose the random variable N further.

We now proceed to the proof of Theorem 2.1, which is extremely clever. There are two parts, and both are somewhat delicate. The first part shows that a particular decomposition

[2] In more detail: because all density functions f_i are bounded above by $1/\sigma$, the probability that a given subset S winds up in a given slice is at most ϵ/σ. (To see this, fix a slice, and condition on all item weights except for one item in S.) Thus, the probability that some slice contains two or more points can be bounded above by $\frac{n}{\epsilon} \cdot (\frac{\epsilon}{\sigma})^2$, which goes to 0 as $\epsilon \to 0$. We can also assume that no solutions lands squarely on a slice boundary, since this is a probability 0 event.

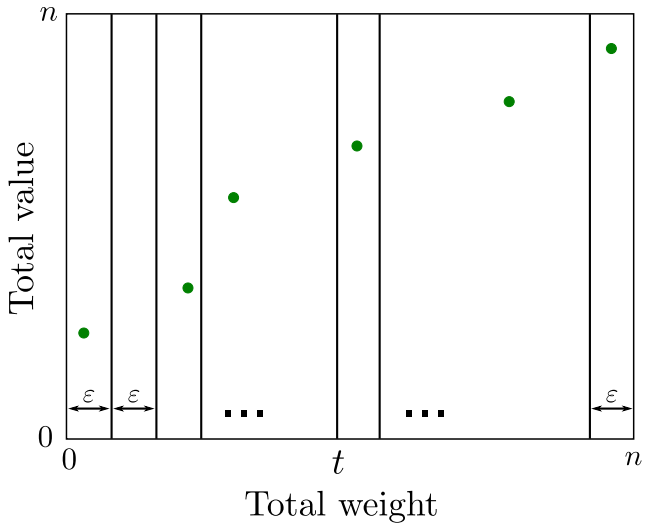

Figure 2: Slices, Pareto optimal solutions, and the slice boundary t.

of N as the sum of indicator random variables is valid, in that the latter only overestimates N. The second part uses the smoothed instance hypothesis to upper bound the probabilities of the indicator variables. The definitions for the first part will seem obscure, but as we'll see, they are carefully crafted to facilitate the second part.

To define the indicator random variables, fix a slice boundary t and an item $i \in \{1, 2, \ldots, n\}$. Ultimately, we want to "charge" each Pareto optimal solution S to the slice boundary immediately to its left and to an item i in S that is responsible for its high value. Formally, we define four random variables.

> $P_{it} \subseteq \{1, 2, \ldots, n\}$ is the solution that maximizes total value subject to two constraints: (i) the total weight of P_{it} is at most t; and (ii) P_{it} does *not* include item i.

Note that P_{it} is a random variable, since a given subset might or might not have weight at most t, depending on the item weights' coin flips. Also note that P_{it} might be the same as the maximum-value solution with weight at most t (if the latter happens to exclude item i), or not (otherwise). Similarly, P_{it} might or might not be a Pareto optimal point.

> V_{it} is the total value of P_{it}.

> S_{it} is the solution that minimizes the total weight subject to two constraints: (i) the total value of S_{it} is more than V_{it}; and (ii) S_{it} includes item i.

Note that the total weight of S_{it} could be more or less than that of P_{it}.

> N_{it} is 1 if the total weight of S_{it} lies in the interval $(t, t + \epsilon)$, and 0 if S_{it} lies in some other slice.

The N_{it}'s are, finally, random variables that we can analyze directly. Before we do this, however, we need to argue that the N_{it}'s cover all the Pareto optimal solutions.

Lemma 3.1 *With probability 1,*

$$\sum_{\text{slices } t} \sum_{\text{items } i} N_{it} \geq N.$$

Proof: We need to show that for every contribution to the right-hand side — every Pareto optimal solution $S \subseteq \{1, 2, \ldots, n\}$ — there is a corresponding contribution to the left-hand side. So, for each Pareto optimal solution we exhibit a distinct item-boundary pair (i, t) with $N_{it} = 1$ — this is the "charging argument" alluded to above.

Condition on all of the items' weights and fix a Pareto optimal solution S. Define t as the slice boundary immediately to the left of S. Thus, the total weight of S lies in the interval $(t, t+\epsilon)$. Let P denote the maximum-value solution with weight at most t, and call its total value V. Since S is Pareto optimal and P has strictly less weight than S, it must be that S has total value strictly larger than V (otherwise P would dominate it). This can only occur when there is at least one item in S that is not also in P; let i denote an arbitrary item of $S \setminus P$.

Now let's trace through the definition of all the random variables — eventually, we will conclude that $N_{it} = 1$. We begin with P_{it}, defined as the maximum-value solution with weight at most t that excludes item i. Since the maximum-value solution with weight at most t, namely P, happens to exclude item i, it must be that $P = P_{it}$. Thus, by the definitions, $V = V_{it}$. Next, recall that S_{it} is defined as the minimum weight solution with value strictly larger than V_{it} (equivalently, strictly larger than V) that excludes item i. By the definition of P and V, every solution with total value strictly greater than V has total weight strictly greater than t. Since S happens to include item i and also happens to have total weight in the earliest possible slice (total weight in $(t, t+\epsilon)$), it must be that $S_{it} = S$ and $N_{it} = 1$.

Finally, since we chose ϵ small enough that distinct Pareto optimal solutions have distinct left boundaries t, every Pareto optimal solution is "charged" (i.e., implies that $N_{it} = 1$) to distinct (i, t) pairs. This completes the proof. ■

Lemma 3.1 reduces the task of bounding N to that of bounding the probability that a given N_{it} equals 1.

Lemma 3.2 *For every i and t, $\mathbf{E}[N_{it}] \leq \frac{\epsilon}{\sigma}$.*

Proof: The lemma is equivalent to proving that, for every i and t, the probability that the set S_{it} has total weight in $(t, t+\epsilon)$ is at most $\frac{\epsilon}{\sigma}$. Recall that S_{it} is the minimum-weight solution with value greater than V_{it} that includes item i, where V_{it} is the maximum value of a solution that excludes item i and has total weight at most t.

So, fix i and t. The dominoes start falling as soon as we condition on all of the random item weights except for that of item i.

1. The total weight and total value of every set that excludes item i is now fixed. (We've conditioned on all weights other than that of item i, and values have been fixed all along.) This means that P_{it} and hence V_{it} are now fixed.

2. The total weight of every subset S that includes item i is a fixed amount W_S (namely, $\sum_{j \in S \setminus \{i\}} w_j$) plus the random weight w_i of item i. This implies that the relative ordering by total weight of all subsets including i is now fixed — it is the same as the relative ordering by the W_S's. The total values of all of these sets are also fixed (since the v_i's were never random).

3. Since S_{it} is the minimum-weight subset that includes item i and has value at least V_{it}, V_{it} is fixed, the subsets with value at least V_{it} are fixed, and the relative order by total weight of the sets including i are fixed, the set S_{it} is fixed.

Summarizing, all that remains random is the actual weight of the (fixed) set S_{it}, which has the form $W + w_i$ for a fixed number W. Thus, the bad event that S_{it} has total weight in the interval $(t, t+\epsilon)$ translates to the still-random weight w_i taking on a value in an interval that has length less than ϵ. Since the density function of w_i is bounded above by $1/\sigma$, this probability is at most ϵ/σ. Since this inequality holds conditioned on arbitrary weights for the items other than i, the unconditional inequality of the lemma follows. ■

Using that there are $n \cdot \frac{n}{\epsilon}$ choices for i and t, combining Lemmas 3.1 and 3.2 with linearity of expectation implies that

$$\mathbf{E}[N] \leq \mathbf{E}\left[\sum_{i,t} N_{it}\right] \leq \frac{n^2}{\epsilon} \cdot \max_{i,t} \mathbf{Pr}[N_{it} = 1] \leq \frac{n^2}{\sigma}.$$

This completes the proof of Theorem 2.1.

Remark 3.3 Over the past few years there has been much progress in extending Theorem 2.1 to a smoothed analysis of higher-dimensional Pareto curves. (In d dimensions, there are d linear objective functions, one point dominates another if and only if it is at least as good in every coordinate, and the Pareto curve contains all undominated points.) The right answer is somewhere between n^d/σ^d and n^{2d}/σ^d; see [4, 2, 3] for more details.

References

[1] R. Beier and B. Vöcking. Typical properties of winners and losers in discrete optimization. *SIAM Journal on Computing*, 35(4):855–881, 2006.

[2] T. Brunsch, N. Goyal, L. Rademacher, and H. Röglin. Lower bounds for the average and smoothed number of Pareto-optima. *Theory of Computing*, 10(10):237–256, 2014.

[3] T. Brunsch and H. Röglin. Improved smoothed analysis of multiobjective optimization. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, pages 407–426, 2012.

[4] A. Moitra and R. O'Donnell. Pareto optimal solutions for smoothed analysts. *SIAM Journal on Computing*, 41(5):1266–1284, 2012.

[5] G. Nemhauser and Z. Ullmann. Discrete dynamic programming and capital allocation. *Management Science*, 15(9):494–505, 1969.