Counting Linear Extensions of a Partial Order

Seth Harris

March 16, 2011

1 Introduction

A *partially ordered set* $(P, <)$ is a set P together with an irreflexive, transitive relation. A *linear extension* of $(P, <)$ is a relation $(P, \prec)$ such that (1) for all $a, b \in P$ either $a \prec b$ or $a = b$ or $b \prec a$, and (2) if $a < b$ then $a \prec b$; in other words, a total order that preserves the original partial order. We define $\Lambda(P)$ as the set of all linear extensions of P, and define $N(P) = |\Lambda(P)|$. Throughout this paper, we will only be considering finite partial orders on n vertices.

The problem of counting linear extensions is not only important in its home field of order theory, but is intimately connected to the theory of sorting. After all, if one is performing a comparison sort, one often starts with incomplete data (a *partially* sorted set) and must decide what alternatives exist for a total sorting, and whether there is an optimal pair of elements to compare next. Clearly one needs to know the number of possible extensions in order to proceed. It is also common to have partial data but to be unable to compare every pair of data points, in which case one must estimate a total ranking of the set, for instance by ranking the "average height" of an element, its mean rank over all linear extensions. This has widespread applications in the social sciences, such as in ranking lists of consumer products or athletic competitors. But again, to have any hope of calculating this, one must know the (approximate) number of extensions.

Graham Brightwell and Peter Winkler have shown in [3] that the problem of exactly counting $N(P)$ is #P-complete. Our goal is to present a randomized algorithm which approximates $N(P)$ within a factor of $(1 + \epsilon)$ with probability $> 1 - \beta$, such that the running time is polynomial in n, $1/\epsilon$, and $1/\beta$.

2 Generating a random linear extension

The first step to achieving our goal comes from an algorithm of Karzanov and Khachiyan [6] which outputs a random linear extension of P in polynomial time:

Theorem 2.1 (Karzanov and Khachiyan)**.** *There is an algorithm with the following properties:*

Input: An n-element partial order $(P, <)$, a number $\epsilon > 0$.

Output: A linear extension of P, such that for any $\lambda \in \Lambda(P)$

$$\left| \mathbb{P}(\text{output is } \lambda) - \frac{1}{N(P)} \right| < \frac{\epsilon}{N(P)}$$

Running Time: $O(n^6 \log(n) \log(1/\epsilon))$

We now define a graph associated with $\Lambda(P)$ and consider a Markov chain for a random walk on this graph.

Definition 2.2. The *linear extension graph* of P, $G(P)$, is the graph whose vertices are linear extensions in $\Lambda(P)$, and such that λ and μ are adjacent in the graph if and only if λ can be obtained from μ by an adjacent transposition.

For a very simple example, suppose that we have a partial order on 4 vertices $\{v_1, v_2, v_3, v_4\}$ with relations $v_1 < v_2$, $v_1 < v_3$. One linear extension of this is $\lambda = v_1 < v_2 < v_3 < v_4$. What other extensions are adjacent to λ in $G(P)$? There are only three possible adjacent transpostions, and the one that swaps v_1 and v_2 violates the original order relation $v_1 < v_2$. So that leaves two legal adjacent transpositions: $v_1 < v_3 < v_2 < v_4$ and $v_1 < v_2 < v_4 < v_3$, and thus λ has degree 2 in $G(P)$. Note that the highest possible degree of a vertex is $n - 1$.

The *order Markov chain* for P is the chain for a random walk on $G(P)$ with the following transition matrix:

$$P_{\lambda,\mu} = \begin{cases} 1/(2n-2) & \text{if } \lambda \text{ and } \mu \text{ are adjacent} \\ 1 - d(\lambda)/(2n-2) & \text{if } \lambda = \mu \\ 0 & \text{otherwise} \end{cases}$$

This Markov chain is clearly aperiodic, since it loops with probability at least 1/2. It is also irreducible. For suppose that $\lambda = v_1 < v_2 < \ldots v_n$ and $\mu = v_{\sigma(1)} < v_{\sigma(2)} < \ldots v_{\sigma(n)}$ are two valid linear extensions. How do we "travel" from λ to μ on the graph? Simple: Find $v_{\sigma(1)}$ in λ, move it to the bottom by transposing it one place at a time; then find $v_{\sigma(2)}$ in λ and move it to second from the bottom, etc. All such moves will be legal, since all relations in P hold in both λ and μ, and therefore will hold in all intermediate steps in the algorithm.

So we have an ergodic Markov chain, and thus if we can find a stationary distribution, the chain will converge to this distribution. And indeed the uniform distribution, $\pi_\mu = \frac{1}{N(P)} \; \forall \, \mu$, is stationary, since

$$\pi_\mu = \frac{1}{N(P)} = (1)\left(\frac{1}{N(P)}\right) = \left[\sum_{\lambda,\mu \text{ adj}} \frac{1}{2n-2} + \left(1 - \frac{d(\mu)}{2n-2}\right)\right]\left(\frac{1}{N(P)}\right) = \sum \pi_\lambda P_{\lambda,\mu}.$$

Therefore, we will eventually converge to the uniform distribution. It remains to show that we have rapid mixing, i.e., that the Markov chain will converge in polynomial time.

Definition 2.3. The *conductance*, α, of a reversible Markov chain with underlying graph G, is defined as

$$\alpha = \frac{1}{2n-1} \min_X \frac{|E(X, \bar{X})|}{|X|}$$

.

where the minimum is taken over all $X \subseteq V(G)$ such that $1 \leq |X| \leq |G|/2$. $E(X, \bar{X})$ is the total number of edges between a vertex in X and a vertex in $\bar{X}$.

The conductance of a reversible Markov chain is frequently used to prove rapid mixing, largely thanks to the following result of Sinclair and Jerrum [7]:

Theorem 2.4 (Sinclair, Jerrum)**.**

$$\left| \pi(\lambda, t) - \frac{1}{N(P)} \right| \leq \left(1 - \frac{\alpha^2}{2}\right)^t$$

Hence it suffices to show that the conductance α is sufficiently large to guarantee rapid mixing, and in some way this is the key step and the true content of Karzanov and Khachiyan's proof. And indeed, they show in [6]:

Theorem 2.5 (Karzanov and Khachiyan)**.** *The conductance of the random walk on $G(P)$ satisfies $\alpha > 2^{-3/2} n^{-5/2}$.*

I will not delve into the details of their proof, but it involves translating our problem into the problem of computing the volume of certain convex bodies. The *order polytope* $Q(P)$ of a partial order $P = \{p_1, \ldots, p_n\}$ is the subset of $[0,1]^n$ such that $x^i < x^j$ if and only if $p_i < p_j$ in P. One can subdivide the n-cube $[0,1]^n$ into $n!$ regions of equal volume, one corresponding to each linear extension $p_{i_1} < \ldots < p_{i_n}$, and so computing the volume of $Q(P)$ would easily tell you how many linear extensions of P exist.

Now that we know that $\alpha > 2^{-3/2} n^{-5/2}$, we can solve for t in the equation in (2.4), and we have $|\pi(\lambda, t) - \frac{1}{N(P)}| < \frac{\epsilon}{N(P)}$ provided $t > 16n^6 \log n \log 1/\epsilon$. Thus we have found a random polynomial-time algorithm for generating a linear extension, as desired.

3 Approximating the number of linear extensions

Now that we can generate a single linear extension in polynomial time, how does this help us count the total number of extensions? Let us simulate the process of slowly building a linear order out of a partial order. Given a partial order P, define a sequence of partial orders $P = P_0, P_1, \ldots, P_k$, where P_k is a linear extension of P_0 and P_{j+1} is just P_j with one

additional relation $a_j < b_j$ (plus the transitive closure, i.e., if $a < a_j$ in P_j, then $a < b_j$ is also a new relation in P_{j+1}). If we are at stage P_j, we can ask ourselves what the probability is that $a_j \prec b_j$ in the final linear order P_k; we will call this probablility $\mathbb{P}(a_j \prec b_j \mid P_j)$. Now we observe that

$$\mathbb{P}(a_j \prec b_j \mid P_j) = \frac{|\text{extensions of } P_j \text{ with } a_j \prec b_j|}{|\text{total extensions of } P_j|} = \frac{N(P_{j+1})}{N(P_j)}$$

.

Therefore,

$$N(P) \;=\; \prod_{j=0}^{k-1} \frac{N(P_j)}{N(P_{j+1})} \;=\; \prod_{j=0}^{k-1} \frac{1}{\mathbb{P}(a_j \prec b_j \mid P_j)}$$

(Note that $N(P_k) = 1$.) So we now have a bridge between Karzanov and Khachiyan's algorithm (2.1) and our final desired result: First we run the algorithm on P a large number of times, giving us a large random sample of linear extensions of P. Next we look at our sample and calculate what proportion of the extensions satisfy $a_j \prec b_j$. Then, we take that proportion as an estimate for $\mathbb{P}(\, a_j \prec b_j \,|\, P_j)$. We then add (a_j, b_j) to our ordering to obtain P_{j+1}, and proceed to estimate $\mathbb{P}(\, a_{j+1} \prec b_{j+1} \,|\, P_{j+1})$. Our next algorithm will be the "proportion-calculating" one.

Theorem 3.1 (Brightwell and Winkler [3])**.** *There is an algorithm with the following properties:*

Input: An n-element partial order $(P, <)$; two numbers η, δ, $\quad 0 < \eta, \delta < 1/3$;
an ordered pair (x, y) incomparable in P such that $\mathbb{P}(x \prec y \mid P) = \gamma \;>\; 2/5$.

Output: An estimate U for γ such that

$$\mathbb{P}\left(\left|\frac{U}{\gamma} - 1\right| > \eta\right) < \delta$$

Running Time: $O(n^6 \log(n) \log(1/\eta)\, \eta^{-2} \log(1/\delta))$

Let us sketch a few details of the proof of this algorithm. The probability p that one trial of the K-K algorithm outputs an order with $x \prec y$ satisfies $|p/\gamma - 1| \leq \epsilon$, where we take $\epsilon = \eta/3$. (This follows from multiplying equation (2) through by $N(P)$ and simplifying). Since we will be running the algorithm a large N number of times, the number $S_{N,p}$ of orderings with $x \prec y$ will be a binomial random variable in N and p. Standard calculations involving the binomial distribution show that if $p > 1/3$,

$$\mathbb{P}(|S_{N,p} - Np| > \epsilon Np) < \exp(-\epsilon^2 N/100)$$

(Note: The assumption that $p > 1/3$ is very mild, for if it is not, just switch the roles of x and y). A few quick inequality calculations will deduce that

$$\mathbb{P}(|U - \gamma| > 3\epsilon\gamma) < \exp(-\epsilon^2 N/100)$$

So if we choose $N > 100\eta^{-2}log(1/\delta)$, then we have, as desired,

$$\mathbb{P}\left(\left|\frac{U}{\gamma} - 1\right| > \eta\right) < \delta$$

.

Note that the running time is nearly the same as the one in (2.1), except multiplied by a factor of $\eta^{-2}\log(1/\delta)$, the approximate size of N. This makes sense, as we are performing N trials of the random linear extension generator.

While for us the calculation of $\mathbb{P}(a_j \prec b_j \mid P_j)$ is essentially a stepping stone to our final result, there is a significant unsolved problem related to it — the so-called "1/3–2/3 conjecture." The conjecture states: For any partial order P, there exist a pair of incomparable elements (a, b) such that $\mathbb{P}(a \prec b \mid P) \in [1/3, 2/3]$. This result would be valuable for comparison sorting, as it implies that there exist (a, b) such that comparing them will reduce the number of possible sortings by a factor of 2/3. Brightwell, Felsner and Trotter [2] show that the conjecture is true if one replaces $[1/3, 2/3]$ with $[(5 - \sqrt{5})/10, (5 + \sqrt{5})/10] \sim [0.2764, 0.7236]$.

We are now in a position to estimate $N(P)$, using the identity obtained in (3):

Theorem 3.2 (Brightwell and Winkler)**.** *There is an algorithm with the following properties:*

Input: An n-element partial order $(P, <)$; two numbers $\epsilon > 0$, $\beta > 0$

Output: A number L such that

$$\mathbb{P}\left(\left|\frac{L}{N(P)} - 1\right| > \epsilon\right) < \beta.$$

Running Time: $O(n^9 \log^6(n) \log(1/\epsilon)\, \epsilon^{-2} \log(1/\beta))$

Starting with $P_0 = P$, we follow a sorting algorithm with no more than $2n \log n$ steps. When we find a pair (a, b) which are not related in P_j, run the previous algorithm (3.1) to estimate $\mathbb{P}(a \prec b \mid P_j)$, using $\eta = \epsilon/(4n \log n)$ and $\delta = \beta/(2n \log n)$. Without loss of generality, we have $\mathbb{P}(a \prec b \mid P_j) > 2/5$ (again, if not, just reverse the roles of a and b). So with probability $> 1 - \delta$, our estimate E_j for $\mathbb{P}(a \prec b \mid P_j)$ will satisfy

$$\left|\frac{E_j}{\mathbb{P}(a \prec b \mid P_j)} - 1\right| < \eta.$$

Add the relation $a \prec b$ (plus transitive closure) to obtain P_{j+1}. We continue following the sorting algorithm until every pair has been compared, at which point we have reached

a linear order P_k. By the nature of the sorting algorithm, $k < 2n \log n$. Now we make an estimate L for $N(P)$, given by $L = \prod_{j=0}^{k-1} E_j^{-1}$, again following the identity in (3). Thus we have (this is an exact equality):

$$\frac{L}{N(P)} = \prod_{j=0}^{k-1} \frac{\mathbb{P}(a_j \prec b_j \mid P_j)}{E_j}$$

With probability $\geq 1 - k\delta \geq 1 - \beta$, each term in the product is between $(1 + \epsilon/2k)^{-1}$ and $(1 - \epsilon/2k)^{-1}$. Thus with probability $\geq 1 - \beta$,

$$1 - \epsilon < \left(\frac{1}{1 + \epsilon/2k}\right)^k < \frac{L}{N(P)} < \left(\frac{1}{1 - \epsilon/2k}\right)^k < 1 + \epsilon$$

exactly as desired.

If we compare the running time of the extension count algorithm (3.2) with the running time of the proportion estimator (3.1), we note that in (3.2) each proportion estimate takes $O(n^6 \log(n) \log(1/\eta)\, \eta^{-2} \log(1/\delta))$ steps, which by the way we defined η and δ, is bounded above by $O(n^8 \log^5(n) \log(1/\epsilon)\, \epsilon^{-2} \log(1/\beta))$. Since we perform this estimate at most $2n \log n$ times, that gives us our running time bound stated above.

Given that our main results were proved two decades ago, it is no surprise that there are more efficient algorithms for counting $N(P)$ today. Indeed, Banks et al. [1] detail a method called the Tootsie Pop Algorithm (TPA) that counts $N(P)$ in $O(n^3 \log n\, (\log N(P))^2 \epsilon^{-2} \log(1/\beta))$ steps.

References

[1] J. Banks, S. Garrabrant, M. Huber, A. Perizzolo (2010). "Using TPA to count linear extensions." arXiv:1010.4981v1

[2] G. Brightwell, S. Felsner, W. Trotter, *Balancing Pairs and the Cross-Product Conjecture*. Order 12 (1995), 327-349.

[3] G. Brightwell and P. Winkler, *Counting Linear Extensions*. Order 8 (1991), 225-242.

[4] M. Dyer, A. Frieze, R. Kannan, *A Random Polynomial-Time Algorithm for Approximating the Volume of Convex Bodies*. Proc 21st ACM Symposium on the Theory of Computing, 375-381.

[5] O. Häggström, *Finite Markov Chains and Algorithmic Applications*. Cambridge University Press, Cambridge, 2008.

[6] A. Karzanov and L. Khachiyan, *On the Conductance of Order Markov Chains*. Order 8 (1991), 7-15.

[7] A. Sinclair and M. Jerrum, *Approximate counting, generation, and rapidly mixing Markov chains*. Information and Computation 82 (1989), 93-133.