Math 206B Lecture 15 Notes

Daniel Raban

February 11, 2019

1 The NPS Algorithm

1.1 Motivation

Here is the hook length formula.

Theorem 1.1 (FRT, 1958)**.**

$$f^\lambda = \#\,\mathrm{SYT}(\lambda) = \frac{n!}{\prod_{(i,j)\in\lambda} h_{i,j}}.$$

The following is a general result from representation theory.

Theorem 1.2. *Let χ be a character of a finite group G, and let $d = \chi(1)$ be the dimension of the representation. Then $d \mid |G|$.*

In this case, we get the following.

Corollary 1.1. $\#\,\mathrm{SYT}(\lambda) \mid n!$.

Why should this be true?

Example 1.1. Let $\lambda = (n-k, 1^k)$. Then

$$f^\lambda = \binom{n-1}{k} = \frac{(n-1)!}{k!(n-k)!} \mid (n-1)!,$$

so it divides $n!$.

Example 1.2. Let $\lambda = (m, m)$, where $n = 2m$. Then

$$f^\lambda = \frac{1}{m+1}\binom{2m}{m} = \frac{(2m)!}{m!(m+1)!} = \frac{n!}{m!(m+1)!}.$$

Why should this be an integer?

Today, we will construct $\Phi_\lambda : S_n \to \mathrm{SYT}(\lambda)$, such that $\Phi_\lambda^{-1}(A)$ is a constant depending on λ. In reality, this constant will be $\prod h_{i,j}$.

Example 1.3. Let $\lambda = (n)$. Then $f^\lambda = 1$. In this case, Φ_λ will be some sorting algorithm.

Example 1.4. Let $\lambda = (1^n)$. Then $f^\lambda = 1$. In this case, Φ_λ will sort vertically.

1.2 Construction of Φ_λ

This will sort of be like 2-dimensional bubble sort.

Take $\lambda = (5, 5, 3, 2)$, and choose a permutation:

9	8	11	12	4
15	3	14	2	5
10	7	1		
13	6			

Look at the last column. The column is sorted, so we are ok. Now include the next column. 12 is bigger than 2, so we must switch them. Same with 12 and 5.

9	8	11	2	4
15	3	14	5	12
10	7	1		
13	6			

Now look at the 1 in the next column. This is sorted. Now look at the 14. This is bigger than the 1, so we need to switch it.

9	8	11	2	4
15	3	1	5	12
10	7	14		
13	6			

Going up the columns, we need to switch the 11 with something. It must be switched with the 11. Then we need to switch the 11 with the 5.

9	8	1	2	4
15	3	5	11	12
10	7	14		
13	6			

Move on to the next column. The 6 is fine, but when we move up to the 7, we see that we have to switch it with the 6.

9	8	1	2	4
15	3	5	11	12
10	6	14		
13	7			

The 3 is fine where it is. But the 8 above needs to be switched. We switch it with the 1, then the 2, and then the 4 until this part is sorted.

9	1	2	4	8
15	3	5	11	12
10	6	14		
13	7			

The 13 is okay, but the 10 needs to be switched. Switch it with the 6.

9	1	2	4	8
15	3	5	11	12
6	10	14		
13	7			

The 15 has a long way to go. See if you can figure out where it needs to go:

9	1	2	4	8
3	5	11	12	15
6	10	14		
13	7			

Finally, we move the 9 where it needs to go:

1	2	4	8	9
3	5	11	12	15
6	10	14		
13	7			

We now have a standard Young tableau. We had no choice at each step, so we can see that this algorithm is well-defined. Notice that this is similar to jeu-de-taquin.

1.3 Construction of Ψ_λ

Now we will construct $\Psi_\lambda : S_n \to \prod_{(i,j)\in\lambda}[-\lambda'_j + 1, \cdots, \lambda_i - i]$. This range has size $h_{i,j}$.

Lemma 1.1. $(\Phi_\lambda, \Psi_\lambda)$ *gives a bijection between* S_n *and the Cartesian product of* $\mathrm{SYT}(\lambda)$ *with the above product.*

Example 1.5. When we are just doing bubblesort, Ψ_λ will give us $(a_1, \ldots, a_2, a_1)$, where a_i is the number of steps the i-th number made.

Take our example from earlier:

9	8	11	12	4
15	3	14	2	5
10	7	1		
13	6			

Let's record the sorting procedure and keep track of what numbers move.

9	8	11	12	4
15	3	14	2	5
10	7	1		
13	6			

0	0	0	0	0
0	0	0	0	0
0	0	0		
0	0			

12 is moved down and then to the right. Our rule is that when we move to the right, we add 1. When we move a number down, we switch the two numbers and subtract 1 from the top.

9	8	11	2	4
15	3	14	12	5
10	7	1		
13	6			

0	0	0	−1	0
0	0	0	0	0
0	0	0		
0	0			

9	8	11	2	4
15	3	14	5	12
10	7	1		
13	6			

0	0	0	−1	0
0	0	0	1	0
0	0	0		
0	0			

When we move the 14 down, we get:

9	8	11	2	4
15	3	1	5	12
10	7	14		
13	6			

0	0	0	−1	0
0	0	−1	1	0
0	0	0		
0	0			

When 11 goes down, we switch the two blocks and then subtract 1 from the new top:

9	8	1	2	4
15	3	11	5	12
10	7	14		
13	6			

0	0	−2	−1	0
0	0	0	1	0
0	0	0		
0	0			

Now the 11 has to move to the right:

9	8	1	2	4
15	3	5	11	12
10	7	14		
13	6			

0	0	−2	−1	0
0	0	1	1	0
0	0	0		
0	0			

The 7 in the next column has to be moved down:

9	8	1	2	4
15	3	5	11	12
10	6	14		
13	7			

0	0	−2	−1	0
0	0	1	1	0
0	−1	0		
0	0			

Now we have to move the 8. We only change the numbers in the "column and row of action."

9	1	2	4	8
15	3	5	11	12
10	6	14		
13	7			

0	3	−2	−1	0
0	0	1	1	0
0	−1	0		
0	0			

Now move the 13:

9	1	2	4	8
15	3	5	11	12
10	6	14		
13	7			

0	3	−2	−1	0
0	0	1	1	0
0	−1	0		
1	0			

Continuing like this, we get

1	2	4	8	9
3	5	11	12	15
6	10	14		
13	7			

4	3	-2	-1	0
4	0	1	1	0
1	-1	0		
1	0			

In summary, right moves change the numbers on the right as

x	$\mapsto$	$x+1$	
y		y	

and down moves change the numbers on the right as

x	$\mapsto$	$y-1$	
y		x	

Check that we can invert this algorithm. Next time, we will hear the story of this algorithm and learn about the GNW algorithm.