The Lindström-Gessel-Viennot Lemma

Jasper Ty

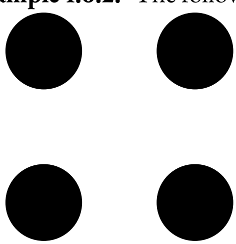
I needed to re-understand this lemma to understand the proof of equivalence of the two definitions of Schur polynomials, one as a quotient of determinants and the other as a certain weight generating function.

1 Preliminaries

Let's get the graph theory out of the way.

Definition 1.0.1 (Digraphs). A *directed graph* D is a pair (V, A) consisting of a set V of vertices and a set A of arcs (normally called *edges* for undirected graphs) consisting of pairs of the form (v_1, v_2) where $v_1, v_2 \in V$.

Example 1.0.2. The following is a digraph



Definition 1.0.3 (Arcs). Let $a = (v_1, v_2)$ be an arc of a digraph. v_1 will be called the *source* of the arc and v_2 will be called the *destination* of the arc. For any such arc a, define $\overset{\uparrow}{\bullet}\, a = v_1$ and $\underset{\uparrow}{\bullet}\, a = v_2$.

Definition 1.0.4 (Paths)**.** A *path* p in a digraph D is a finite sequence of arcs $(a_1, \ldots, a_n)$ connected end-to-end, i.e

$$\overset{\bullet}{\uparrow}\, a_i = \overset{\uparrow}{\bullet}\, a_{i+1}$$

for all $i < n$.

We define the *source* of a p and *destination* of p to be

$$\overset{\uparrow}{\bullet}\, p := \overset{\uparrow}{\bullet}\, a_1$$

$$\overset{\bullet}{\uparrow}\, p := \overset{\bullet}{\uparrow}\, a_n$$

We say that a digraph D is *path-finite* whenever there are only finitely many paths between any two vertices u, v in V. We say D is *acyclic* when it does not contain any cycles. A digraph D is *simple* whenever there are no self-arcs.

Definition 1.0.5 (Lattice Paths)**.** Consider the digraph whose vertex set is $\mathbb{Z}^2$, and whose edge set consists of the arcs

$$(i, j) \to (i + 1, j) \qquad \text{for all } (i, j) \in \mathbb{Z}^2,$$

which will be called *right-steps*, and

$$(i, j) \to (i, j + 1) \qquad \text{for all } (i, j) \in \mathbb{Z}^2,$$

which will be called *up-steps*.

This digraph will be called the *lattice* and paths in the lattice will be called *lattice paths*.

The lattice is acyclic, simple, and path-finite.

Remark 1.0.6. Lattice paths may be identified with a starting point A and a *step sequence* consisting of up-steps and right-steps. The step sequence can be identified further with strings consisting of the letters U and R.

2 Counting lattice paths

Apply the previous remark to counting lattice paths. If we knew ahead of time how many up-steps (say $\#U$) and right-steps (say $\#R$) we needed to get from a point on the lattice to the other, we can biject all paths between two points with strings consisting of $\#U$ up-steps and $\#R$ right-steps. This string has length $\#U + \#R$. With some quick reasoning we get the formula

$$(\#\text{ of paths}) = \binom{\#U + \#R}{\#U} = \binom{\#U + \#R}{\#R}.$$

Fortunately, we do know ahead of time how many up-steps and right-steps we need to get from one point to another on the lattice. Let $(a, b), (c, d) \in \mathbb{Z}^2$ and consider counting paths $(a, b) \to (c, d)$. Then $\#R = c - a$ and $\#U = d - b$.

But, $\#U$ and $\#R$ are apparently meaningless if it's impossible to get from one point to the other on the lattice (i.e no path exists). Intuitively, this happens if and only if the destination further left than the source ($c < a$) or further down than the source ($d < b$) or both.

Put another way, a path exists if and only if $c \geq a$ and $d \geq b$. A necessary *but not sufficient* condition of this is that (c, d) lies within the half-space in $\mathbb{R}^2$ defined by $x + y \geq a + b$. This is not sufficient because the cone defined by the intersections of $\{(x, y) : x \geq a\}$ and $\{(x, y) : y \geq b\}$ is strictly smaller than the aforementioned half-space.

But that's not a problem! Because the trick here is that we use a fairly clever definition of the binomial coefficient which makes a whole slew of identities much easier to write:

Definition 2.0.1 (Binomial coefficients)**.**

$$\binom{n}{k} := \begin{cases} \dfrac{n(n-1)(n-2)\cdots(n-k+1)}{k!}, & \text{if } k \in \mathbb{N} \\ 0, & \text{otherwise.} \end{cases}$$

This totally solves our problem, since negative k automatically zero the binomial coefficient. The requirement that the destination live in a certain half-space will guarantee that the top entry of our binomial coefficient is ultimately positive (and we don't have to deal with any upper negation business). These two facts lend us the following conclusion:

Proposition 2.0.2. Let $(a, b), (c, d) \in \mathbb{Z}^2$. Then

$$(\#\text{ of paths from } (a, b) \text{ to } (c, d)) = \begin{cases} \binom{c+d-a-b}{c-a}, & \text{if } c + d \geq a + b \\ 0, & \text{otherwise.} \end{cases}$$

Proof. Put $\#R = c - a$ and $\#U = d - b$. □

3 Tuples of lattice paths

Now that we have a foothold in lattice path counting, let's consider the problem of counting *tuples* of them.

Definition 3.0.1. Let $k \in \mathbb{N}$

1. A *k-vertex* is a k-tuple of lattice points $(A_1, A_2, \ldots, A_k) \in (\mathbb{Z}^2)^k$.
2. Permutations will act on k-vertices by permuting indices, i.e if

$$\mathbf{A} = (A_1, A_2, \ldots A_k)$$

then $\sigma(\mathbf{A}) = (A_{\sigma(1)}, A_{\sigma(2)}, \ldots, A_{\sigma(k)})$.

3. If $\mathbf{A} = (A_1, A_2, \ldots A_k)$ and $\mathbf{B} = (B_1, B_2, \ldots B_k)$, then a *path tuple* from $\mathbf{A}$ to $\mathbf{B}$ is a k-tuple $(p_1, p_2, \ldots, p_k)$, where each p_i is a lattice path from A_i to B_i.
4. A path tuple is *non-intersecting* if p_i and p_j share no vertex in common for all $1 \leq i < j \leq k$.
5. A path tuple is *intersecting* if... it is not non-intersecting.

We will call intersecting path tuples *ipats* and non-intersecting path tuples *nipats*. Now to get some notation in:

Definition 3.0.2. Let A, B be lattice points. The number of paths from A to B will be denoted $\#(A \rightarrow B)$.

Simple enough, let's upgrade to k-vertices.

Definition 3.0.3. Let $\mathbf{A}$ and $\mathbf{B}$ be k-vertices.

- The number of path tuples from $\mathbf{A}$ to $\mathbf{B}$ will be denoted $\#(\mathbf{A} \rightarrow \mathbf{B})$.
- The number of nipats from $\mathbf{A}$ to $\mathbf{B}$ will be denoted $\#(\mathbf{A} \rightrightarrows \mathbf{B})$.
- The number of ipats from $\mathbf{A}$ to $\mathbf{B}$ will be denoted $\#(\mathbf{A} \rightleftarrows \mathbf{B})$.

Hopefully it's clear why this notation in particular. We have a particularly simple result about counting path tuples:

Remark 3.0.4. Let $\mathbf{A} = (A_1, A_2, \ldots, A_k)$ and $\mathbf{B} = (B_1, B_2, \ldots B_k)$. Then we have that.

$$\#(\mathbf{A} \to \mathbf{B}) = \prod_{i=1}^{k} \#(A_i \to B_i).$$

What about counting nipats?

4 The LGV lemma for two paths

Consider the following operation on two intersecting paths.

Definition 4.0.1. Let A_1, A_2, B_1, B_2 be lattice points and let p and q be intersecting paths from A_1 to B_1 and A_2 to B_2 respectively. The path $p \looparrowright q$ will be a path from A_1 to B_2 defined as follows:

Let v be the first point of intersection of p and q.

Let $\mathrm{head}_v\, p$ be all arcs in p between A and v, and $\mathrm{tail}_v\, p$ be all arcs in p between v and B. Define them similarly for q.

Then let $p \looparrowright q := \mathrm{head}_v\, p \cup \mathrm{tail}_v\, q$.

Then we have the following consequence:

Remark 4.0.2. For any two intersecting paths p, q,

$$(p \looparrowright q) \looparrowright (q \looparrowright p) = p$$

Proof.

$$\begin{aligned}
(p \looparrowright q) \looparrowright (q \looparrowright p) &= (\mathrm{head}_v\, p \cup \mathrm{tail}_v\, q) \looparrowright (\mathrm{head}_v\, p \cup \mathrm{tail}_v\, q) \\
&= \mathrm{head}_v(\mathrm{head}_v\, p \cup \mathrm{tail}_v\, q) \cup \mathrm{tail}_v(\mathrm{head}_v\, q \cup \mathrm{tail}_v\, p) \\
&= \mathrm{head}_v\, p \cup \mathrm{tail}_v\, p \\
&= p
\end{aligned}$$

□

And this is the key to proving our first theorem about counting nipats.

Theorem 4.0.3 (Lindström-Gessel-Viennot for lattice paths, $k = 2$)**.** Let $\mathbf{A} = (A_1, A_2)$ and $\mathbf{B} = (B_1, B_2)$. Let $\mathbf{B}' = (B_2, B_1)$. Then

$$\det \begin{pmatrix} \#(A_1 \to B_1) & \#(A_1 \to B_2) \\ \#(A_2 \to B_1) & \#(A_2 \to B_2) \end{pmatrix} = \#(\mathbf{A} \rightrightarrows \mathbf{B}) - \#(\mathbf{A} \rightrightarrows \mathbf{B}')$$

Proof. First, we evaluate the determinant

$$\begin{aligned} &\det \begin{pmatrix} \#(A_1 \to B_1) & \#(A_1 \to B_2) \\ \#(A_2 \to B_1) & \#(A_2 \to B_2) \end{pmatrix} \\ &= \Big[\#(A_1 \to B_1) \cdot \#(A_2 \to B_2)\Big] - \Big[\#(A_1 \to B_2) \cdot \#(A_2 \to B_1)\Big] \\ &= \#(\mathbf{A} \to \mathbf{B}) - \#(\mathbf{A} \to \mathbf{B}'). \end{aligned}$$

and so we have to show that

$$\#(\mathbf{A} \to \mathbf{B}) - \#(\mathbf{A} \to \mathbf{B}') = \#(\mathbf{A} \rightrightarrows \mathbf{B}) - \#(\mathbf{A} \rightrightarrows \mathbf{B}').$$

Since a path tuple is either an ipat or a nipat, we have that

$$\#(\mathbf{A} \to \mathbf{B}) = \#(\mathbf{A} \rightrightarrows \mathbf{B}) + \#(\mathbf{A} \mathbin{\nearrow\!\!\!\!\!\searrow} \mathbf{B}).$$

Then

$$\begin{aligned} \#(\mathbf{A} \to \mathbf{B}) - \#(\mathbf{A} \to \mathbf{B}') &= \Big[\#(\mathbf{A} \rightrightarrows \mathbf{B}) + \#(\mathbf{A} \mathbin{\nearrow\!\!\!\!\!\searrow} \mathbf{B})\Big] - \Big[\#(\mathbf{A} \rightrightarrows \mathbf{B}') + \#(\mathbf{A} \mathbin{\nearrow\!\!\!\!\!\searrow} \mathbf{B}')\Big] \\ &= \Big[\#(\mathbf{A} \rightrightarrows \mathbf{B}) - \#(\mathbf{A} \rightrightarrows \mathbf{B}')\Big] + \Big[\#(\mathbf{A} \mathbin{\nearrow\!\!\!\!\!\searrow} \mathbf{B}) - \#(\mathbf{A} \mathbin{\nearrow\!\!\!\!\!\searrow} \mathbf{B}')\Big]. \end{aligned}$$

Now our goal is further modified to showing that

$$\Big[\#(\mathbf{A} \rightrightarrows \mathbf{B}) - \#(\mathbf{A} \rightrightarrows \mathbf{B}')\Big] + \Big[\#(\mathbf{A} \mathbin{\nearrow\!\!\!\!\!\searrow} \mathbf{B}) - \#(\mathbf{A} \mathbin{\nearrow\!\!\!\!\!\searrow} \mathbf{B}')\Big] = \#(\mathbf{A} \rightrightarrows \mathbf{B}) - \#(\mathbf{A} \rightrightarrows \mathbf{B}').$$

Which simplifies to

$$\#(\mathbf{A} \mathbin{\nearrow\!\!\!\!\!\searrow} \mathbf{B}) = \#(\mathbf{A} \mathbin{\nearrow\!\!\!\!\!\searrow} \mathbf{B}').$$

Nice! If this is true, then the proof follows. It is in fact true, and we can construct an explicit bijection using $\looparrowright$. Define f as a function on pairs of intersecting paths that returns another pair of intersecting paths to be

$$f(p, q) := (p \looparrowright q, q \looparrowright p).$$

we have that

$$\begin{aligned} f(f(p,q)) &= f(p \looparrowright q, q \looparrowright p) \\ &= \Big((p \looparrowright q) \looparrowright (q \looparrowright q), (q \looparrowright p) \looparrowright (p \looparrowright q)\Big) \\ &= (p,q). \end{aligned}$$

Hence f is an involution. □

That wasn't too bad, since it's quite literally the smallest nontrivial case of the theorem. To do better, we will need a more general tool, *sign-reversing involutions*.

5 Sign-reversing involutions

I think *sign-reversing involution* is a good name. Hopefully after seeing the definitions you might agree.

Definition 5.0.1. Let $\mathcal{A}$ be a finite set, and consider a *sign* function

$$\operatorname{sign} : \mathcal{A} \to \mathbb{Z}.$$

A *sign-reversing involution* is an involution f defined on a subset $\mathcal{X}$ of $\mathcal{A}$ such that

$$\operatorname{sign} f(a) = -\operatorname{sign} a$$

for all $a \in \mathcal{A}$. That f is an involution means that

$$f(f(x)) = x$$

for all $x \in \mathcal{X}$.

Then we have a principle that is entirely obvious to me but I never tried proving and have no idea if it is difficult or not to prove:

Theorem 5.0.2 (Cancellation principle)**.** Again, let $\mathcal{A}$ be a finite set with a sign function sign.

Let a sign-reversing involution f be defined for a subset $\mathcal{X}$ of $\mathcal{A}$. Then

$$\sum_{a \in \mathcal{A}} \operatorname{sign} a = \sum_{a \in \mathcal{A} \setminus \mathcal{X}} \operatorname{sign} a,$$

The idea behind the cancellation principle is that, well, the signs of the elements of $\mathcal{X}$ cancel each other out, and f records exactly how they cancel out.

6 Weighted digraphs

Definition 6.0.1. Let D be a digraph.

7 Statement and proof of the LGV lemma

Theorem (Lindström-Gessel-Viennot)**.** Let $k \in \mathbb{N}$.

Let $\mathbf{A} = (A_1, A_2, \ldots, A_k)$ and let $\mathbf{B} = (B_1, B_2, \ldots, B_k)$. Then

$$\det\left[W(A_i \to B_j)\right]_{1 \le i \le k,\ 1 \le j \le k} = \sum_{\sigma \in \mathfrak{S}_k} \operatorname{sgn} \sigma \cdot W\big(\mathbf{A} \rightrightarrows \sigma(\mathbf{B})\big),$$

where $\mathfrak{S}_k$ is the symmetric group on k letters, and sgn denotes the sign of a permutation.

Proof. We're going to set up a sign-reversing involution and use the previous remark.

Let our background set $\mathcal{A}$ be

$$\mathcal{A} := \{(\sigma, \mathbf{p}) : \sigma \in \mathfrak{S}_k \text{ and } \mathbf{p} \in \mathbf{A} \to \sigma(\mathbf{B})\}.$$

And let our subset $\mathcal{X}$ be the set of all ipats in $\mathcal{A}$, i.e

$$\mathcal{X} := \{(\sigma, \mathbf{p}) \in \mathcal{A} : \mathbf{p} \in \mathbf{A} \bigtimes \mathbf{B}\}.$$

Define our sign function on $\mathcal{A}$ to be

$$\operatorname{sign}(\sigma, \mathbf{p}) := \operatorname{sgn}(\sigma).$$

Now let's do the same thing we did before– expand the determinant. We have

$$\begin{aligned}
\det\left[W(A_i \to B_j)\right] &= \sum_{\sigma \in \mathfrak{S}_k} \operatorname{sgn}(\sigma) \cdot \prod_{1 \le i \le k} W\big(A_i \to B_{\sigma(i)}\big) \\
&= \sum_{\sigma \in \mathfrak{S}_k} \operatorname{sgn}(\sigma) \cdot W\big(\mathbf{A} \to \sigma(\mathbf{B})\big).
\end{aligned}$$

It turns out by grouping together $\mathbf{p}$'s we have that

$$\sum_{(\sigma, \mathbf{p}) \in \mathcal{A}} \operatorname{sign}(\sigma, \mathbf{p}) = \sum_{\sigma \in \mathfrak{S}_k} \operatorname{sgn}(\sigma) \cdot W\big(\mathbf{A} \to \sigma(\mathbf{B})\big).$$

And, by considering that $\mathcal{X}$ consists of *precisely all* the ipats, we have

$$\sum_{(\sigma,\mathbf{p})\in\mathcal{X}} \operatorname{sign}(\sigma,\mathbf{p}) = \sum_{\sigma\in\mathfrak{S}_k} \operatorname{sgn}(\sigma)\cdot W\big(\mathbf{A}\rightarrow\sigma(\mathbf{B})\big)$$

and

$$\sum_{(\sigma,\mathbf{p})\in\mathcal{A}\setminus\mathcal{X}} \operatorname{sign}(\sigma,\mathbf{p}) = \sum_{\sigma\in\mathfrak{S}_k} \operatorname{sgn}(\sigma)\cdot W\big(\mathbf{A}\rightrightarrows\sigma(\mathbf{B})\big).$$

Then, since $W(\mathbf{A}\rightarrow\mathbf{B}) = W(\mathbf{A}\rightrightarrows\mathbf{B}) + W(\mathbf{A}\rightarrow\mathbf{B})$,

$$\begin{aligned}
&\sum_{\sigma\in\mathfrak{S}_k} \operatorname{sgn}(\sigma)\cdot W\big(\mathbf{A}\rightarrow\sigma(\mathbf{B})\big)\\
&= \sum_{\sigma\in\mathfrak{S}_k} \operatorname{sgn}(\sigma)\cdot W\big(\mathbf{A}\rightrightarrows\sigma(\mathbf{B})\big) + \sum_{\sigma\in\mathfrak{S}_k} \operatorname{sgn}(\sigma)\cdot W\big(\mathbf{A}\rightarrow\sigma(\mathbf{B})\big)\\
&= \sum_{(\sigma,\mathbf{p})\in\mathcal{A}\setminus\mathcal{X}} \operatorname{sign}(\sigma,\mathbf{p}) + \sum_{(\sigma,\mathbf{p})\in\mathcal{X}} \operatorname{sign}(\sigma,\mathbf{p})\\
&= \sum_{(\sigma,\mathbf{p})\in\mathcal{A}} \operatorname{sign}(\sigma,\mathbf{p}).
\end{aligned}$$

If we find a sign-reversing involution on $\mathcal{X}$, this will prove the desired statement.

"Swapping paths" will suffice again, but we have to take care about our *specific choices* now. Here, the superhero of choosing-from-sets appears to save the day once again: *we fixed an order on our vertices*!

So, given a path tuple $\mathbf{p} = (p_1, \ldots, p_k)$, we may have multiple paths that intersect, i.e we have multiple choices of $i < j$ such that p_i and p_j share a vertex in common. We know how to swap two paths already– meaning, if we had $i < j$ already such that p_i intersects with p_j, we could construct a path tuple

$$\mathbf{p}' = (p_1, \ldots, p_i \looparrowright p_j, \ldots, p_j \looparrowright p_i, \ldots, p_k)$$

from $\mathbf{A}$ to $(ij)(\mathbf{B})$ [1].

Now we just need to make sure we have a well defined procedure that *reliably* picks out two paths from our path tuple. Here's how I would do it:

Assign a total order (say, lexicographic) on all vertices in $\mathbb{Z}^2$.

For any ipat, pick the smallest such vertex that is an intersection of any two paths in the ipat. Then, from the set of paths that contain that vertex, pick the ones with the smallest two indices. This suffices as our pairing rule.

$\square$

[1] where (ij) refers to the transposition that swaps i and j