Exploring Rimhook Rules and Quantum Schubert Calculus

Elizabeth Beazley

Haverford College

Sage Days 45 at ICERM
February 13, 2013

A First Example

Consider projective space $\mathbb{P}^3$. Intersection theory is encoded by the cup product in cohomology. The cohomology of $\mathbb{P}^3$ has a basis indexed by the following Young diagrams:

$$\text{whole space } = \emptyset, \quad \text{plane } = \ydiagram{1}, \quad \text{line } = \ydiagram{2}, \quad \text{point } = \ydiagram{3}$$

These simple representations allow us to compute products in a nice way – as “box addition”. Intuitively, think about 3D space.

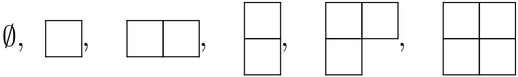
$$\ydiagram{1} \cdot \ydiagram{1} = \ydiagram{2}$$

$$\ydiagram{1} \cdot \ydiagram{2} = \ydiagram{3}$$

$$\ydiagram{1} \cdot \ydiagram{3} = 0$$

A Second Example

This idea can be used on more complicated spaces, too! Now think about the *Grassmannian* of 2-dimensional subspaces of complex 4-dimensional space $Gr(2,4)$. The subvarieties we're interested in are indexed by

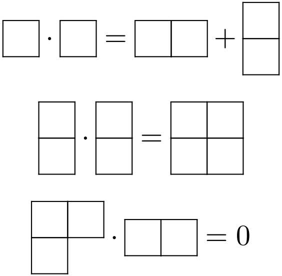


whole space, 3D space, plane, other plane, line, point

In general, cohomology classes of the Grassmannian $Gr(k,n)$ are indexed by Young diagrams fitting inside a $k \times (n-k)$ rectangle, and # boxes corresponds to codimension.

A Second Example

Once again, we can compute intersections/cup products in $H^*(Gr(2,4))$:



$$\ydiagram{1} \cdot \ydiagram{1} = \ydiagram{2} + \ydiagram{1,1}$$

$$\ydiagram{1,1} \cdot \ydiagram{1,1} = \ydiagram{2,2}$$

$$\ydiagram{2,1} \cdot \ydiagram{2} = 0$$

Too many boxes ⟷ sum of the codimensions of intersecting classes is too large

These classes have no *classical* intersection – but they do have *quantum* intersection!

Littlewood-Richardson Rule

The intersections we did previously are fairly simple, but in general they get very complicated. For example, in $Gr(4, 8)$,

$$\ydiagram{2,1} \cdot \ydiagram{2,1} = \ydiagram{4,2} + \ydiagram{4,1,1} + \ydiagram{3,3} + 2\,\ydiagram{3,2,1} + \ydiagram{3,1,1,1} + \ydiagram{2,2,2} + \ydiagram{2,2,1,1}$$

In $H^*(Gr(k, n))$, we wish to expand products of the form

$$\sigma_\lambda \cdot \sigma_\mu = \sum c^{\nu}_{\lambda,\mu} \sigma_\nu, \quad \text{where}$$

- λ, μ, and ν fit inside a $k \times (n - k)$ box and
- $(\#\text{ boxes in }\nu) = (\#\text{ boxes in }\lambda) + (\#\text{ boxes in }\mu)$.

The numbers $c^{\nu}_{\lambda,\mu}$ are called *Littlewood-Richardson coefficients*.

Littlewood-Richardson Rule

Littlewood-Richardson coefficients are known to encode:

- intersection cohomology for the Grassmannian
- expansions of products of Schubert polynomials
- expansions of products of Schur functions
- multiplicities of irreducible representations of products of symmetric groups
- decompositions of tensor products of Schur modules into irreducibles

Anders Buch has developed a *Littlewood-Richardson calculator* which is now running in Sage!

Brief History of Quantum Schubert Calculus

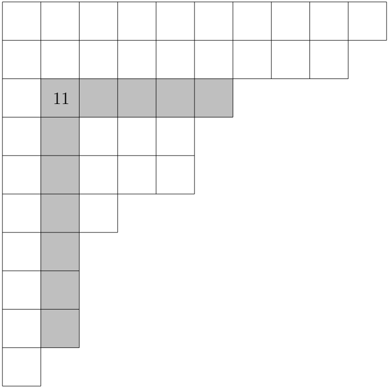
- Physicists wanted to count curves ("worldsheets") in a particular way.
- String theorists in the 1990s invented quantum cohomology.
- Mathematicians seek positive, non-recursive formulas for

$$\sigma_\lambda \star \sigma_\mu = \sum_{\nu,d} c_{\lambda,\mu}^{d,\nu} q^d \sigma_\nu,$$

  where the numbers $c_{\lambda,\mu}^{d,\nu}$ are *quantum Littlewood-Richardson coefficients.*
- There are several methods for computing quantum LR coefficients. ***None of these methods exist in Sage yet!***

The Rimhook Rule

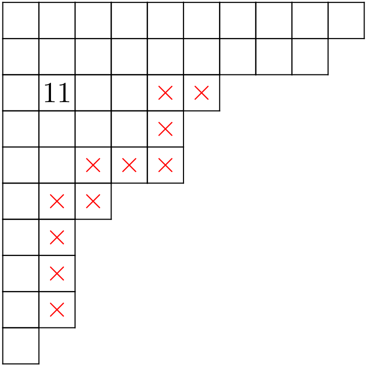
In the late 1990s, Bertram, Ciocan-Fontanine, and Fulton came up with an algorithm called the *rimhook rule* for quantum multiplication. The algorithm involves removing n-rimhooks.



An 11-hook in a Young diagram.

The Rimhook Rule

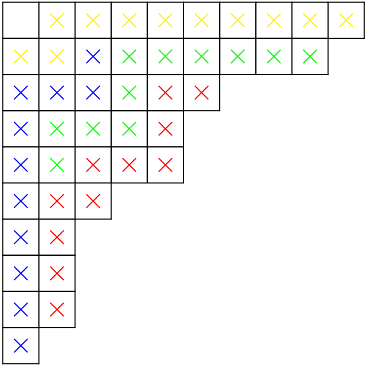
In the late 1990s, Bertram, Ciocan-Fontanine, and Fulton came up with an algorithm called the *rimhook rule* for quantum multiplication. The algorithm involves removing n-rimhooks.



The corresponding removable 11-rimhook.

The Rimhook Rule

Removing all possible n-rimhooks from a partition ν results in the n-core for ν, which we will denote by $c(\nu)$.



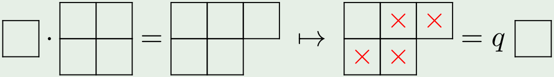
The 11-core for (10, 9, 6, 5, 5, 3, 2, 2, 2, 1) is $\square$

The Rimhook Rule

The Idea: Compute $QH^*(Gr(k,n))$ from $H^*(Gr(k, 2n-k))$, where all products of $k \times (n-k)$ boxes "fit".

Example

To compute $\sigma_{\square} \star \sigma_{\boxplus}$ in $QH^*(Gr(2,4))$, first compute the classical product in $H^*(Gr(2,6))$:



Then remove all possible 4-rimhooks, picking up a (signed) power of q for each rimhook removed. This gives

$$\sigma_{\square} \star \sigma_{\boxplus} = q\sigma_{\square}$$

The Rimhook Rule

Theorem (Bertram, Ciocan-Fontanine, Fulton)

To compute $\sigma_\lambda \star \sigma_\mu \in QH^*(Gr(k,n))$, *first compute* $\sigma_\lambda \cdot \sigma_\mu = \sum c^{\nu}_{\lambda,\mu}\sigma_\nu \in H^*(Gr(k,2n-k))$. *Apply the following rimhook rule to each term in the expression:*

$$\sigma_\nu \mapsto \begin{cases} (-1)^{\sum_i (n-k-ht(R_i))} q^d \sigma_{c(\nu)} & \text{if } c(\nu) \subseteq k \times (n-k) \\ 0 & \text{otherwise.} \end{cases}$$

Here d *equals the total number of rimhooks* R_i *removed to get* $c(\nu)$ *from* ν, *and* $ht(R_i)$ *is the height of the rimhook, which equals the number of rows in* R_i. *Collecting terms gives the quantum product* $\sigma_\lambda \star \sigma_\mu$.

The Rimhook Rule

$$\sigma_\nu \mapsto \begin{cases} (-1)^{\sum_i (n-k-ht(R_i))} q^d \sigma_{c(\nu)} & \text{if } c(\nu) \subseteq k \times (n-k) \\ 0 & \text{otherwise.} \end{cases}$$

Example

$$\sigma_{\square\square} \cdot \sigma_{\square\square} =$$

The Rimhook Rule

$$\sigma_\nu \mapsto \begin{cases} (-1)^{\sum_i (n-k-ht(R_i))} q^d \sigma_{c(\nu)} & \text{if } c(\nu) \subseteq k \times (n-k) \\ 0 & \text{otherwise.} \end{cases}$$

Example

$$\sigma_{\ydiagram{2}} \cdot \sigma_{\ydiagram{2}} = \sigma_{\ydiagram{2,2}} + \sigma_{\ydiagram{3,1}} + \sigma_{\ydiagram{4}}$$

The Rimhook Rule

$$\sigma_\nu \mapsto \begin{cases} (-1)^{\sum_i (n-k-ht(R_i))} q^d \sigma_{c(\nu)} & \text{if } c(\nu) \subseteq k \times (n-k) \\ 0 & \text{otherwise.} \end{cases}$$

Example

$$\begin{aligned} \sigma_{\ydiagram{2}} \cdot \sigma_{\ydiagram{2}} &= \sigma_{\ydiagram{2,2}} + \sigma_{\ydiagram{3,1}} + \sigma_{\ydiagram{4}} \\ &= \sigma_{\ydiagram{2,2}} + \sigma_{\cancel{\ydiagram{3,1}}} + \sigma_{\cancel{\ydiagram{4}}} \end{aligned}$$

The Rimhook Rule

$$\sigma_\nu \mapsto \begin{cases} (-1)^{\sum_i (n-k-ht(R_i))} q^d \sigma_{c(\nu)} & \text{if } c(\nu) \subseteq k \times (n-k) \\ 0 & \text{otherwise.} \end{cases}$$

Example

$$\begin{aligned} \sigma_{\ydiagram{2}} \cdot \sigma_{\ydiagram{2}} &= \sigma_{\ydiagram{2,2}} + \sigma_{\ydiagram{3,1}} + \sigma_{\ydiagram{4}} \\ &= \sigma_{\ydiagram{2,2}} + \sigma_{\begin{ytableau} \times & \times & \times \\ \times \end{ytableau}} + \sigma_{\begin{ytableau} \times & \times & \times & \times \end{ytableau}} \\ &\mapsto \sigma_{\ydiagram{2,2}} + (-1)^{(2-2)} q\sigma_{.} + (-1)^{(2-1)} q\sigma_{.} \end{aligned}$$

The Rimhook Rule

$$\sigma_\nu \mapsto \begin{cases} (-1)^{\sum_i (n-k-ht(R_i))} q^d \sigma_{c(\nu)} & \text{if } c(\nu) \subseteq k \times (n-k) \\ 0 & \text{otherwise.} \end{cases}$$

Example

$$\begin{aligned} \sigma_{\ydiagram{2}} \cdot \sigma_{\ydiagram{2}} &= \sigma_{\ydiagram{2,2}} + \sigma_{\ydiagram{3,1}} + \sigma_{\ydiagram{4}} \\ &= \sigma_{\ydiagram{2,2}} + \sigma_{\cancel{\ydiagram{3,1}}} + \sigma_{\cancel{\ydiagram{4}}} \\ &\mapsto \sigma_{\ydiagram{2,2}} + (-1)^{(2-2)} q\sigma_{.} + (-1)^{(2-1)} q\sigma_{.} \\ &= \sigma_{\ydiagram{2,2}} + q\sigma_{.} - q\sigma_{.} \end{aligned}$$

The Rimhook Rule

$$\sigma_\nu \mapsto \begin{cases} (-1)^{\sum_i (n-k-ht(R_i))} q^d \sigma_{c(\nu)} & \text{if } c(\nu) \subseteq k \times (n-k) \\ 0 & \text{otherwise.} \end{cases}$$

Example

$$\begin{aligned}
\sigma_{\ydiagram{2}} \cdot \sigma_{\ydiagram{2}} &= \sigma_{\ydiagram{2,2}} + \sigma_{\ydiagram{3,1}} + \sigma_{\ydiagram{4}} \\
&= \sigma_{\ydiagram{2,2}} + \sigma_{\ytableaushort{\times\times\times,\times}} + \sigma_{\ytableaushort{\times\times\times\times}} \\
&\mapsto \sigma_{\ydiagram{2,2}} + (-1)^{(2-2)} q\sigma_{.} + (-1)^{(2-1)} q\sigma_{.} \\
&= \sigma_{\ydiagram{2,2}} + q\sigma_{.} - q\sigma_{.} \\
&= \sigma_{\ydiagram{2,2}}
\end{aligned}$$

The Rimhook Rule

$$\sigma_\nu \mapsto \begin{cases} (-1)^{\sum_i (n-k-ht(R_i))} q^d \sigma_{c(\nu)} & \text{if } c(\nu) \subseteq k \times (n-k) \\ 0 & \text{otherwise.} \end{cases}$$

Example

$$\begin{aligned} \sigma_{\ydiagram{2}} \cdot \sigma_{\ydiagram{2}} &= \sigma_{\ydiagram{2,2}} + \sigma_{\ydiagram{3,1}} + \sigma_{\ydiagram{4}} \\ &= \sigma_{\ydiagram{2,2}} + \sigma_{\ytableaushort{\times\times\times,\times}} + \sigma_{\ytableaushort{\times\times\times\times}} \\ &\mapsto \sigma_{\ydiagram{2,2}} + (-1)^{(2-2)} q\sigma. + (-1)^{(2-1)} q\sigma. \\ &= \sigma_{\ydiagram{2,2}} + q\sigma. - q\sigma. \\ &= \sigma_{\ydiagram{2,2}} \\ &= \sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{2}} \end{aligned}$$

Possible Sage Projects

(1) Put quantum Littlewood-Richardson coefficients into Sage. There are several possible methods for implementing the quantum Littlewood-Richardson coefficients:

- Apply the rimhook rule to the results from Buch's Littlewood-Richardson commands.
- Solve for them recursively using the *quantum Pieri rule*.

The Quantum Pieri Rule

The *Pieri rule* says how to multiply by a special Schubert class

$$\sigma_{\square} \cdot \sigma_{\lambda} = \sum_{\mu \to \lambda} \sigma_{\mu},$$

where $\mu \to \lambda$ means that $\mu = \lambda \cup \square$.

The *quantum Pieri rule* similarly tells us that in $QH^*(Gr(k,n))$,

$$\sigma_{\square} \star \sigma_{\lambda} = \sum_{\mu \to \lambda} \sigma_{\mu} + q\sigma_{\lambda^-},$$

where $\lambda^- = \lambda -$ an $(n-1)$-rimhook.

The Quantum Pieri Rule

$$\sigma_{\square} \star \sigma_{\lambda} = \sum_{\mu \to \lambda} \sigma_{\mu} + q\sigma_{\lambda^-},$$

Example

In $Gr(2,4)$, we can use the quantum Pieri rule to compute:

$$\begin{aligned}
\sigma_{\square} \star \sigma_{\square} &= \sigma_{\ydiagram{2}} + \sigma_{\ydiagram{1,1}} \\
\sigma_{\square} \star \sigma_{\ydiagram{1,1}} &= \sigma_{\ydiagram{2,1}} \\
\sigma_{\square} \star \sigma_{\ydiagram{2}} &= \sigma_{\ydiagram{2,1}} \\
\sigma_{\square} \star \sigma_{\ydiagram{2,1}} &= \sigma_{\ydiagram{2,2}} + q\sigma. \\
\sigma_{\square} \star \sigma_{\ydiagram{2,2}} &= q\sigma_{\square}
\end{aligned}$$

The Quantum Pieri Rule

To compute another product, we can solve recursively, using the fact that the product is commutative and associative.

Example

$$\sigma_{\tiny\begin{array}{|c|c|}\hline \, & \, \\ \hline\end{array}} \star \sigma_{\tiny\begin{array}{|c|}\hline \, \\ \hline \, \\ \hline\end{array}} =$$

The Quantum Pieri Rule

To compute another product, we can solve recursively, using the fact that the product is commutative and associative.

Example

$$\sigma_{\square\square} \star \sigma_{\substack{\square\\\square}} = \left(\sigma_{\square} \star \sigma_{\square} - \sigma_{\substack{\square\\\square}}\right) \star \sigma_{\substack{\square\\\square}}$$

The Quantum Pieri Rule

To compute another product, we can solve recursively, using the fact that the product is commutative and associative.

Example

$$\sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} = (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1}} - \sigma_{\ydiagram{1,1}}) \star \sigma_{\ydiagram{1,1}} = \sigma_{\ydiagram{1}} \star (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} =$$

The Quantum Pieri Rule

To compute another product, we can solve recursively, using the fact that the product is commutative and associative.

Example

$$\begin{aligned}\sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} &= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1}} - \sigma_{\ydiagram{1,1}}) \star \sigma_{\ydiagram{1,1}} = \sigma_{\ydiagram{1}} \star (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} = \\ &= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}}\end{aligned}$$

The Quantum Pieri Rule

To compute another product, we can solve recursively, using the fact that the product is commutative and associative.

Example

$$\begin{aligned}\sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} &= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1}} - \sigma_{\ydiagram{1,1}}) \star \sigma_{\ydiagram{1,1}} = \sigma_{\ydiagram{1}} \star (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} = \\ &= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} \\ &= \sigma_{\ydiagram{2,2}} + q - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} \qquad \text{and so by rearranging,}\end{aligned}$$

The Quantum Pieri Rule

To compute another product, we can solve recursively, using the fact that the product is commutative and associative.

Example

$$
\begin{aligned}
\sigma_{\yng(2)} \star \sigma_{\yng(1,1)} &= (\sigma_{\yng(1)} \star \sigma_{\yng(1)} - \sigma_{\yng(1,1)}) \star \sigma_{\yng(1,1)} = \sigma_{\yng(1)} \star (\sigma_{\yng(1)} \star \sigma_{\yng(1,1)}) - \sigma_{\yng(1,1)} \star \sigma_{\yng(1,1)} = \\
&= (\sigma_{\yng(1)} \star \sigma_{\yng(2,1)}) - \sigma_{\yng(1,1)} \star \sigma_{\yng(1,1)} \\
&= \sigma_{\yng(2,2)} + q - \sigma_{\yng(1,1)} \star \sigma_{\yng(1,1)} \qquad \text{and so by rearranging,} \\
\sigma_{\yng(2,2)} &= \sigma_{\yng(2)} \star \sigma_{\yng(1,1)} + \sigma_{\yng(1,1)} \star \sigma_{\yng(1,1)} - q. \quad \text{Left multiplying by } \sigma_{\yng(1)},
\end{aligned}
$$

The Quantum Pieri Rule

To compute another product, we can solve recursively, using the fact that the product is commutative and associative.

Example

$$\begin{aligned}
\sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} &= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1}} - \sigma_{\ydiagram{1,1}}) \star \sigma_{\ydiagram{1,1}} = \sigma_{\ydiagram{1}} \star (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} = \\
&= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} \\
&= \sigma_{\ydiagram{2,2}} + q - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} \qquad \text{and so by rearranging,} \\
\sigma_{\ydiagram{2,2}} &= \sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} + \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} - q. \quad \text{Left multiplying by } \sigma_{\ydiagram{1}},
\end{aligned}$$

$$q\sigma_{\ydiagram{1}} = \sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2,2}} = \sigma_{\ydiagram{1}} \star (\sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} + \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} - q)$$

The Quantum Pieri Rule

To compute another product, we can solve recursively, using the fact that the product is commutative and associative.

Example

$$\begin{aligned}
\sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} &= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1}} - \sigma_{\ydiagram{1,1}}) \star \sigma_{\ydiagram{1,1}} = \sigma_{\ydiagram{1}} \star (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} = \\
&= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} \\
&= \sigma_{\ydiagram{2,2}} + q - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} \qquad\qquad \text{and so by rearranging,} \\
\sigma_{\ydiagram{2,2}} &= \sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} + \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} - q. \quad \text{Left multiplying by } \sigma_{\ydiagram{1}}, \\
q\sigma_{\ydiagram{1}} = \sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2,2}} &= \sigma_{\ydiagram{1}} \star (\sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} + \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} - q) \\
&= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2}}) \star \sigma_{\ydiagram{1,1}} + (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1,1}}) \star \sigma_{\ydiagram{1,1}} - q\sigma_{\ydiagram{1}}
\end{aligned}$$

The Quantum Pieri Rule

To compute another product, we can solve recursively, using the fact that the product is commutative and associative.

Example

$$\begin{aligned}
\sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} &= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1}} - \sigma_{\ydiagram{1,1}}) \star \sigma_{\ydiagram{1,1}} = \sigma_{\ydiagram{1}} \star (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} = \\
&= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} \\
&= \sigma_{\ydiagram{2,2}} + q - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} \qquad \text{and so by rearranging,} \\
\sigma_{\ydiagram{2,2}} &= \sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} + \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} - q. \ \ \text{Left multiplying by } \sigma_{\ydiagram{1}}, \\
q\sigma_{\ydiagram{1}} = \sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2,2}} &= \sigma_{\ydiagram{1}} \star (\sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} + \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} - q) \\
&= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2}}) \star \sigma_{\ydiagram{1,1}} + (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1,1}}) \star \sigma_{\ydiagram{1,1}} - q\sigma_{\ydiagram{1}} \\
&= \sigma_{\ydiagram{2,1}} \star \sigma_{\ydiagram{1,1}} + \sigma_{\ydiagram{2,1}} \star \sigma_{\ydiagram{1,1}} - q\sigma_{\ydiagram{1}}
\end{aligned}$$

The Quantum Pieri Rule

To compute another product, we can solve recursively, using the fact that the product is commutative and associative.

Example

$$
\begin{aligned}
\sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} &= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1}} - \sigma_{\ydiagram{1,1}}) \star \sigma_{\ydiagram{1,1}} = \sigma_{\ydiagram{1}} \star (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} = \\
&= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} \\
&= \sigma_{\ydiagram{2,2}} + q - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} \qquad \text{and so by rearranging,} \\
\sigma_{\ydiagram{2,2}} &= \sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} + \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} - q. \ \ \text{Left multiplying by } \sigma_{\ydiagram{1}}, \\
q\sigma_{\ydiagram{1}} = \sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2,2}} &= \sigma_{\ydiagram{1}} \star (\sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} + \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} - q) \\
&= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2}}) \star \sigma_{\ydiagram{1,1}} + (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1,1}}) \star \sigma_{\ydiagram{1,1}} - q\sigma_{\ydiagram{1}} \\
&= \sigma_{\ydiagram{2,1}} \star \sigma_{\ydiagram{1,1}} + \sigma_{\ydiagram{2,1}} \star \sigma_{\ydiagram{1,1}} - q\sigma_{\ydiagram{1}} \\
&= 2\sigma_{\ydiagram{2,1}} \star \sigma_{\ydiagram{1,1}} - q\sigma_{\ydiagram{1}}. \qquad \text{Therefore,}
\end{aligned}
$$

The Quantum Pieri Rule

To compute another product, we can solve recursively, using the fact that the product is commutative and associative.

Example

$$\begin{aligned}
\sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} &= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1}} - \sigma_{\ydiagram{1,1}}) \star \sigma_{\ydiagram{1,1}} = \sigma_{\ydiagram{1}} \star (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} = \\
&= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2,1}}) - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} \\
&= \sigma_{\ydiagram{2,2}} + q - \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} \qquad \text{and so by rearranging,} \\
\sigma_{\ydiagram{2,2}} &= \sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} + \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} - q. \quad \text{Left multiplying by } \sigma_{\ydiagram{1}}, \\
q\sigma_{\ydiagram{1}} = \sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2,2}} &= \sigma_{\ydiagram{1}} \star (\sigma_{\ydiagram{2}} \star \sigma_{\ydiagram{1,1}} + \sigma_{\ydiagram{1,1}} \star \sigma_{\ydiagram{1,1}} - q) \\
&= (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{2}}) \star \sigma_{\ydiagram{1,1}} + (\sigma_{\ydiagram{1}} \star \sigma_{\ydiagram{1,1}}) \star \sigma_{\ydiagram{1,1}} - q\sigma_{\ydiagram{1}} \\
&= \sigma_{\ydiagram{2,1}} \star \sigma_{\ydiagram{1,1}} + \sigma_{\ydiagram{2,1}} \star \sigma_{\ydiagram{1,1}} - q\sigma_{\ydiagram{1}} \\
&= 2\sigma_{\ydiagram{2,1}} \star \sigma_{\ydiagram{1,1}} - q\sigma_{\ydiagram{1}}. \qquad \text{Therefore,} \\
2q\sigma_{\ydiagram{1}} &= 2\sigma_{\ydiagram{2,1}} \star \sigma_{\ydiagram{1,1}}, \qquad \text{and so} \qquad \sigma_{\ydiagram{2,1}} \star \sigma_{\ydiagram{1,1}} = q\sigma_{\ydiagram{1}}.
\end{aligned}$$

The Quantum Pieri Rule: Generalizations

The Grassmannian is a special example of a *partial flag variety*. There are quantum Pieri (or at least *Chevalley-Monk*) rules for:

- the complete flag variety in $\mathbb{C}^n$
- the complete flag variety G/B in other Lie types
- partial flag varieties G/P in other Lie types

In all of these cases, the quantum Pieri/Chevalley-Monk rule completely determines the full multiplication table in $QH^*(G/P)$, whereas other tools for carrying out quantum multiplication are not as fully developed.

So although this method is not efficient for $Gr(k, n)$, it may be the only way to experiment with many of these generalizations.

Possible Sage Projects

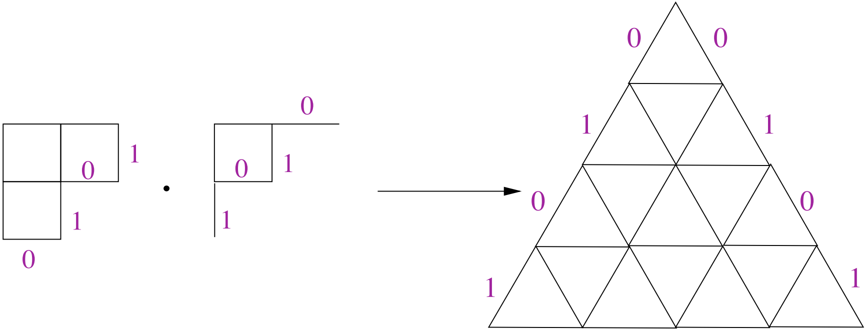
(1) Put quantum Littlewood-Richardson coefficients into Sage. There are several possible methods for implementing the quantum Littlewood-Richardson coefficients:

- Apply the rimhook rule to the results from Buch's Littlewood-Richardson commands.
- Solve for them recursively using the quantum Pieri rule.

(2) Encode quantum Pieri/Chevalley-Monk rules in other Lie types and recursively program the full multiplication table.

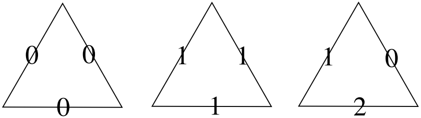
The Puzzle Rule

In 2001, Knutson and Tao invented *puzzles* for computing the Littlewood-Richardson coefficients. Read off the southeast edge of your diagrams to get the puzzle to fill:



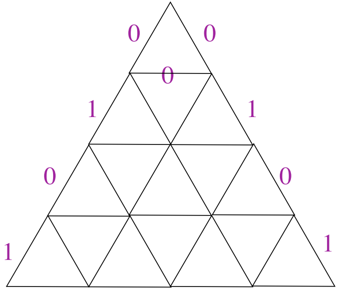
The Puzzle Rule

Use these puzzle pieces to fill the puzzle:

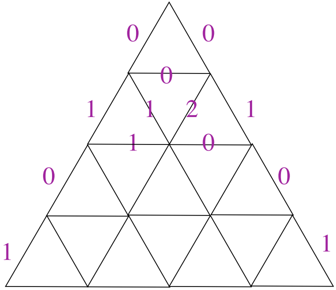


You cannot flip over the pieces; only rotations are allowed.

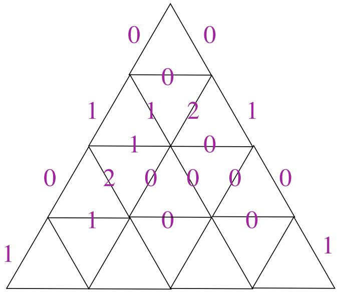
The Puzzle Rule



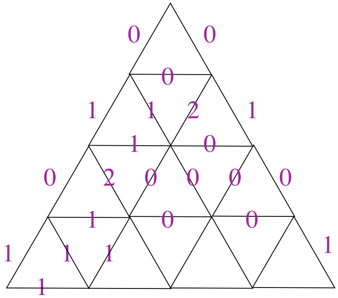
The Puzzle Rule



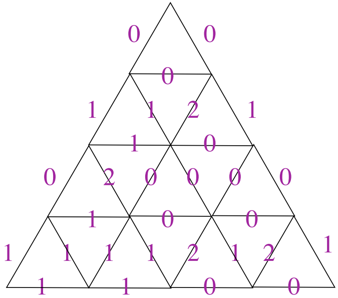
The Puzzle Rule



The Puzzle Rule



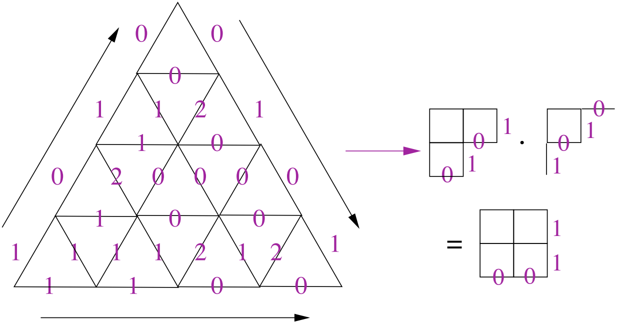
The Puzzle Rule



A completed puzzle!

The Puzzle Rule

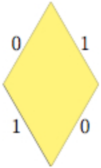
In this case, there was a unique filling that worked.



In general, there may be either none or several. Each valid puzzle contributes a term to the product in $H^*(Gr(k,n))$.

The Puzzle Rule: Generalizations

Knutson and Tao also developed a puzzle rule for computing products in *equivariant cohomology*. There is one additional puzzle piece:



The *equivariant Littlewood-Richardson coefficients* are polynomials in $\mathbb{Z}_{\geq 0}[T_1 - T_2, T_2 - T_3, \ldots, T_{n-1} - T_n]$.

Possible Sage Projects

(1) Put quantum Littlewood-Richardson coefficients into Sage. There are several possible methods for implementing the quantum Littlewood-Richardson coefficients:

- Apply the rimhook rule to the results from Buch's Littlewood-Richardson commands.
- Solve for them recursively using the quantum Pieri rule.
- Apply the rimhook rule to the outputs of a puzzle algorithm.

(2) Encode quantum Pieri/Chevalley-Monk rules in other Lie types and recursively program the full multiplication table.

(3) Implement a puzzle algorithm for both classical and equivariant cohomology of $Gr(k, n)$.

(4) Quantum Schubert polynomials?