Combinatorics of Cohomology Rings of Peterson Varieties

Dr. Brent Gorbutt, Martha Hartt, Ziqi Zhan, Ryan Simmons, Aidan Donahue

George Mason University, MEGL

May 6, 2022

Vandermonde Identity

Theorem (Vandermonde's Identity)

Given a group of m objects of one type, and n of another, there are

$$\binom{m+n}{r} = \sum_{k=0}^{r} \binom{m}{k}\binom{n}{r-k}$$

ways to choose r objects from the group.

Question: Are there any identities which are similar Vandermonde, but with more binomial coefficients?

Dr. Goldin and Dr. Gorbutt's Identity

Theorem 9 [1]

Let $m, n, w, x, y, z \in \mathbb{Z}$ such that $w + x = y + z$ and $m, n \geq 0$. Then

$$\binom{x+m}{w}\binom{y+m}{x}\binom{w+n}{y}\binom{z+n}{z}$$
$$= \sum_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} \binom{x+i+n}{w+i+j}\binom{w+m+j}{i, j, m-i, x-i-j, z-x+j, y-x+i}$$

- Identities of this type are used in the study of cohomology rings.
- Idea: Construct $\mathcal{V}$ and $\mathcal{S}$, such that $|\mathcal{S}| = \text{RHS}$ and $|\mathcal{V}| = \text{LHS}$.
- Use the method of bike lock moves to find bijection between $\mathcal{V}$ and $\mathcal{S}$.
- We wish to apply the same process to find additional identities.

Table of Values

	1's	0's	$*$'s	sum
V_1	x_1	y_1	y_6	n
V_2	x_2	y_2	y_1	n
V_3	x_3	y_3	y_2	n
V_4	x_4	y_4	y_3	n
V_5	x_5	y_5	y_4	n
V_6	x_6	y_6	y_5	n

Working Toward Another Combinatorial Identity

Six Binomial Coefficients

While we do not believe a similar identity can be proven using bike lock moves for three or five binomial coefficients, we made progress in identifying the right hand side of an identity for

$$\binom{x_1+y_1}{x_1}\binom{x_1+y_6}{x_2}\binom{y_1+x_2}{x_3}\binom{y_6+x_1-x_2+x_3}{x_4}\times \binom{y_1+x_2-x_3+x_4}{x_5}\binom{x_6+y_6}{x_6}$$

where $x_1 + x_3 + x_5 = x_2 + x_4 + x_6$.

Bike lock moves will help us construct a set $\mathcal{S}$ which has a bijective correspondence with $\mathcal{V}$ to find the right hand side of an identity.

Bike Lock Moves

Definition (Bike Lock Move)

Let a matrix have r rows and c columns. For each column k such that with $1 \leq k \leq c$, a bike lock move BL_k on a set of matrices M_c with $c < 0$ columns is a map $M_c \to M_c$ such that, for all $M \in M_c$,

1. $BL_k(M)$ is identical to M except in a specified subset of rows $R_{BL_k}(M)$.
2. $BL_k(M)$ cyclically permutes the entries in row $l \in R_{BL_k}(M)$ as follows:
   - An entry in column $m < k$ is fixed.
   - An entry in column m with $k \leq m < c$ of M sent to column $m + 1$ in the same row.
   - If $m = c$, the entry is sent to the kth column of the same row.

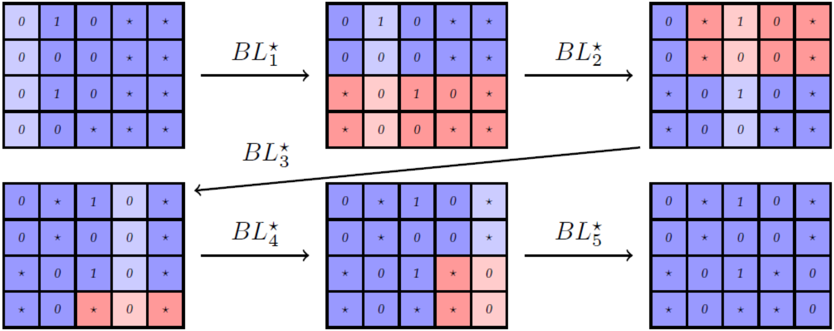
Bike Lock Moves

A bike lock move BL_3 on a 4 x 5 matrix M with $R_{BL_3}(M) = \{1, 3\}$; can be seen as follows. Impacted entries are highlighted in red.

$$\begin{bmatrix} a_{11} & a_{12} & a & b & c \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & d & e & f \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \end{bmatrix} \longrightarrow \begin{bmatrix} a_{11} & a_{12} & c & a & b \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & f & d & e \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \end{bmatrix}.$$

Bike Lock Moves

We use a series of bike lock moves on a matrix such that each 0 in a column has a * to go with it.



Bike Lock Moves

1	1	1	1	0	*	*	*	*	*	*
1	1	1	1	1	1	1	1	0	0	*
1	1	1	1	1	1	0	0	0	*	*
1	1	1	1	0	0	0	0	*	*	*
1	1	0	0	0	0	0	*	*	*	*
0	0	0	0	0	0	*	*	*	*	*

Bike Lock Moves

*	*	1	1	1	1	0	*	*	*	*
1	1	1	1	1	1	1	1	0	0	*
1	1	1	1	1	1	0	0	0	*	*
1	1	1	1	0	0	0	0	*	*	*
1	1	0	0	0	0	0	*	*	*	*
0	0	0	0	0	0	*	*	*	*	*

Bike Lock Moves

*	*	1	1	1	1	0	*	*	*	*
1	1	1	1	1	1	1	1	0	0	*
1	1	1	1	1	1	0	0	0	*	*
1	1	1	1	0	0	0	0	*	*	*
1	1	0	0	0	0	0	*	*	*	*
0	0	*	*	0	0	0	0	*	*	*

Bike Lock Moves

*	*	1	1	*	*	1	1	0	*	*
1	1	1	1	1	1	1	1	0	0	*
1	1	1	1	1	1	0	0	0	*	*
1	1	1	1	0	0	0	0	*	*	*
1	1	0	0	*	*	0	0	0	*	*
0	0	*	*	0	0	0	0	*	*	*

Bike Lock Moves

*	*	1	1	*	*	1	1	0	*	*
1	1	1	1	1	1	1	1	0	0	*
1	1	1	1	1	1	0	0	0	*	*
1	1	1	1	0	0	*	*	0	0	*
1	1	0	0	*	*	0	0	0	*	*
0	0	*	*	0	0	*	*	0	0	*

Bike Lock Moves

*	*	1	1	*	*	1	1	*	0	*
1	1	1	1	1	1	1	1	0	0	*
1	1	1	1	1	1	0	0	*	0	*
1	1	1	1	0	0	*	*	0	0	*
1	1	0	0	*	*	0	0	*	0	*
0	0	*	*	0	0	*	*	0	0	*

Bike Lock Moves

*	*	1	1	*	*	1	1	*	*	0
1	1	1	1	1	1	1	1	0	0	*
1	1	1	1	1	1	0	0	*	*	0
1	1	1	1	0	0	*	*	0	0	*
1	1	0	0	*	*	0	0	*	*	0
0	0	*	*	0	0	*	*	0	0	*

Code Screenshots

```
def pickRows(col):
    i = 0
    rows = []
    if len(np.where(np.array(col) == 1)[0]) == 0:
        return [1,3,5]
    startIndex = np.where(np.array(col) == 1)[0][0]
    modIndex = startIndex
    while i < len(col):
        modIndex = (startIndex + i) % len(col)
        if col[modIndex] == 0 and col[((modIndex + 1) % len(col))] != 0:
            rows.append((modIndex + 1) % len(col))
            i += 1
        elif col[modIndex] == 0 and col[((modIndex + 1) % len(col))] == 0:
            rows.append((modIndex + 1) % len(col))
            i += 2
        else:
            i += 1
    return rows
```

Given any unaligned column, the function above will tell us the rows we need to perform bike lock moves on. We conjecture that this function will allow us to align any matrix when the number of rows is even.

Code Screenshots

```
def align(matrix):
    m_ = len(matrix)
    n_ = len(matrix[0])
    for i in range(0,n_):
        column = []
        rowList = []
        for j in range(0,m_):
            column.append(matrix[j][i])

        rowList = pickRows(column)

        if len(rowList) != 0:
            rowList = [x + 1 for x in rowList]
            blMove(matrix, i + 1, rowList)
    return matrix


def count(matrix):
    n_ = len(matrix[0])
    colDict = {'Sequence': str(matrix.tolist())}
    for num in range(1,len(df.columns)):
        count = 0
        for col in range(n_):
            if np.array_equal(matrix[:,col],np.array(eval(df.columns[num])))  == True:
                count += 1
        colDict[df.columns[num]] = count
    return colDict
```

Output

[1,1,1,1,1,1]	[0,3,1,1,1,1]	[1,0,3,1,1,1]	[1,1,0,3,1,1]	[1,1,1,0,3,1]	[1,1,1,1,0,3]	[3,1,1,1,1,0]	[3,1,1,0,3,0]	[3,0,3,1,1,0]	[1,1,0,3,0,3]	[1,0,3,0,3,1]	[0,3,0,3,1,1]	[3,1,0,3,1,0]	[0,3,1,0,3,1]	[0,3,1,1,0,3]	[0,3,0,3,0,3]	[3,0,3,0,3,0]	[1,0,3,1,0,3]
2	0	0	0	0	2	2	2	0	2	0	0	0	0	0	2	3	0
0	0	0	0	1	1	3	2	0	2	1	0	1	0	1	1	1	1
0	0	0	0	1	1	2	3	1	3	0	1	0	0	1	0	1	1
0	0	0	0	2	1	2	2	0	1	0	0	2	0	1	1	1	2
0	0	0	0	2	1	2	1	0	2	0	0	2	0	2	0	2	1
0	0	0	1	1	2	2	2	0	1	0	0	1	0	1	1	2	1
1	0	0	0	0	2	1	3	0	1	1	0	2	0	1	1	1	1
1	0	0	0	0	2	1	3	0	1	1	0	2	0	1	1	1	1
1	0	0	0	0	2	2	3	0	1	1	0	1	0	0	2	1	1
1	0	0	0	0	2	2	3	0	2	0	1	0	0	0	1	2	1
1	0	0	0	0	1	2	3	0	2	1	0	1	0	1	1	1	1
1	0	0	0	1	1	2	2	0	2	0	0	1	0	1	1	2	1
2	0	0	0	0	1	1	3	0	2	0	0	1	0	1	1	2	1
1	0	0	0	0	1	3	3	0	2	1	0	0	0	0	2	1	1
1	0	0	0	0	3	2	2	0	1	1	0	1	0	0	2	2	0
1	0	0	0	0	2	2	2	0	2	0	0	1	1	0	1	2	1
1	0	0	0	0	2	2	3	0	1	1	0	1	0	0	2	1	1
1	0	0	0	1	2	2	2	0	1	0	0	1	0	0	2	2	1
1	0	0	0	1	2	2	1	0	2	0	0	1	0	1	1	3	0
2	0	0	0	0	2	2	2	0	2	0	0	0	0	0	2	3	0
2	0	0	0	0	2	2	2	0	2	0	0	0	0	0	2	3	0
2	0	0	0	0	2	2	2	0	2	0	0	0	0	0	2	3	0
2	0	0	0	0	2	2	2	0	2	0	0	0	0	0	2	3	0
1	0	0	2	2	3	2	1	0	0	0	1	1	0	1	0	1	0
3	0	0	1	0	2	1	3	0	1	1	1	1	0	1	0	0	0
7	0	0	0	0	2	2	2	0	2	0	0	0	0	0	2	3	0
5	0	0	0	1	1	3	2	0	2	1	0	1	0	1	1	1	1
5	0	0	0	1	1	2	3	1	3	0	1	0	0	1	0	1	1
5	0	0	0	2	1	2	2	0	1	0	0	2	0	1	1	1	2
5	0	0	0	2	1	2	1	0	2	0	0	2	0	2	0	2	1
5	0	0	1	1	2	2	2	0	1	0	0	1	0	1	1	2	1
6	0	0	0	0	2	1	3	0	1	1	0	2	0	1	1	1	1

Possible Columns

$$
\begin{matrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 \\
\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} &
\begin{bmatrix} 0 \\ * \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} &
\begin{bmatrix} 1 \\ 0 \\ * \\ 1 \\ 1 \\ 1 \end{bmatrix} &
\begin{bmatrix} 1 \\ 1 \\ 0 \\ * \\ 1 \\ 1 \end{bmatrix} &
\begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ * \\ 1 \end{bmatrix} &
\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ * \end{bmatrix} &
\begin{bmatrix} * \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} &
\begin{bmatrix} 0 \\ * \\ 0 \\ * \\ 1 \\ 1 \end{bmatrix} &
\begin{bmatrix} 0 \\ * \\ 1 \\ 0 \\ * \\ 1 \end{bmatrix}
\end{matrix}
$$

Possible Columns

$$
\begin{array}{ccccccccc}
c_{10} & c_{11} & c_{12} & c_{13} & c_{14} & c_{15} & c_{16} & c_{17} & c_{18} \\
\begin{bmatrix} 0 \\ * \\ 1 \\ 1 \\ 0 \\ * \end{bmatrix} &
\begin{bmatrix} 1 \\ 0 \\ * \\ 0 \\ * \\ 1 \end{bmatrix} &
\begin{bmatrix} 1 \\ 0 \\ * \\ 1 \\ 0 \\ * \end{bmatrix} &
\begin{bmatrix} * \\ 0 \\ * \\ 1 \\ 1 \\ 0 \end{bmatrix} &
\begin{bmatrix} 1 \\ 1 \\ 0 \\ * \\ 0 \\ * \end{bmatrix} &
\begin{bmatrix} * \\ 1 \\ 0 \\ * \\ 1 \\ 0 \end{bmatrix} &
\begin{bmatrix} * \\ 1 \\ 1 \\ 0 \\ * \\ 0 \end{bmatrix} &
\begin{bmatrix} 0 \\ * \\ 0 \\ * \\ 0 \\ * \end{bmatrix} &
\begin{bmatrix} * \\ 0 \\ * \\ 0 \\ * \\ 0 \end{bmatrix}
\end{array}
$$

System of Equations

Now that we know all possible columns, we can create the following system of equations where C_i is the number of columns of type c_i:

$$
\begin{aligned}
C_1 - C_8 - C_9 - C_{10} - C_{11} - C_{12} - C_{13} - C_{14} - C_{15} - C_{16} - 2C_{17} - 2C_{18} &= -x_1 + x_3 + x_5 - 2y_1 - 2y_6 \\
C_2 + C_8 + C_9 + C_{10} + C_{12} &= y_1 \\
C_3 + C_{11} + C_{12} + C_{13} + C_{18} &= x_1 - x_2 + y_6 \\
C_4 + C_8 + C_{14} + C_{15} + C_{17} &= x_2 - x_3 + y_1 \\
C_5 + C_9 + C_{11} + C_{16} + C_{18} &= x_1 - x_2 + x_3 - x_4 + y_6 \\
C_6 + C_{10} + C_{12} + C_{14} + C_{17} &= x_2 - x_3 + x_4 - x_5 + y_1 \\
C_7 + C_{13} + C_{15} + C_{16} + C_{18} &= y_6
\end{aligned}
$$

Progress

- Finished automation of bike lock moves to find the right hand side of an identity for six binomial coefficients.
- Found 18 possible aligned columns for the six case which we will use to find $\mathcal{S}$.
- Conjecture that more identities exist when you have an even number of binomial coefficients.
- Suspect no identity can be proven using bike lock moves when you have an odd number of binomial coefficients.

Future Explorations

- Identify the right hand side of an identity for six binomial coefficients.
- Prove our conjectures about cases when you have other numbers of binomial coefficients.
- Prove the code we used to automate the bike lock moves for six binomial coefficients works more generally.

References

1 Goldin, R., Gorbutt, B. (2022). A positive formula for type A Peterson Schubert calculus. ArXiv:2004.05959 [Math]. https://arxiv.org/pdf/2004.05959

2 Székely, L. (1985). Common Origin of Cubic Binomial Identities; A Generalization of Surányi's Proof on Le Jen Shoo's Formula. J. Comb. Theory Ser. A. **40**(1), 171-174