A calculus for monomials in Chow group $A^{n-3}(n)$

Jiayue Qi

Joint work with Josef Schicho

December, 2021 — LARD

Doctoral Program Computational Mathematics Numerical Analysis and Symbolic Computation

JOHANNES KEPLER UNIVERSITY LINZ

Der Wissenschaftsfonds.

basic setting

- Let $n \in \mathbb{N}$, $n \geq 3$, set $N := \{1, \ldots, n\}$, we call it the **labeling set**.
- A bipartition $\{I, J\}$ of N where the cardinalities of both I and J are at least 2 is called a **cut**. And I, J are called two **parts** of the cut $\{I, J\}$.
- This talk focus on the Chow ring of M_n, where M_n is the moduli space of stable n-pointed curves of genus zero.
- Denote $\delta_{I,J}$ as the class of a cut subvariety $D_{I,J}$ of M_n.
- However, we will not focus on the details of M_n, what is important here is the properties of this Chow ring.
- We denote the Chow ring of M_n as $A^*(n)$.

ambient ring

- It is a graded ring, we have $A^*(n) = \bigoplus_{k=0}^{n-3} A^k(n)$; and these homogeneous components are the Chow groups (of M_n). Here, for instance, we say $A^r(n)$ is a **Chow group of rank** r.
- Fact 1: $A^r(n) = \{0\}$ for $r > n-3$.
- Fact 2: $A^{n-3}(n) \cong \mathbb{Z}$, we denote this isomorphism as $\int : A^{n-3}(n) \longrightarrow \mathbb{Z}$.
- $\{\delta_{I,J} \mid \{I, J\} \text{ is a cut}\}$ is a set of generators for $A^1(n)$; they are also generators for $A^*(n)$, when viewed as ring generators.
- $\prod_{i=1}^{n-3} \delta_{I_i,J_i}$ can be viewed as an element in $A^{n-3}(n)$.
- Goal: calculate the integral value of this monomial, i.e., $\int(\prod_{i=1}^{n-3} \delta_{I_i,J_i})$.

motivation

- This calculus shows up as a subproblem when we wants to improve an algorithm for the realization-counting of Laman graphs on a sphere.
- With the help of this integral value calculus, we invent another algorithm for the same goal.
- However, by efficiency it does not seem faster or better than the existing one.
- But we see that this problem is fundamental, may be helpful for other similar problems, or even further-away problems.
- Then we focus on it, and try to formalize it as a result on its own.

Keel's quadratic relation

Among the generators of $A^*(n)$, we say the two generators $\delta_{I_1,J_1}, \delta_{I_2,J_2}$ fulfill **Keel's quadratic relation** if the following conditions hold:

- $I_1 \cap I_2 \neq \emptyset$;
- $I_1 \cap J_2 \neq \emptyset$;
- $J_1 \cap I_2 \neq \emptyset$;
- $J_1 \cap J_2 \neq \emptyset$.

And when they are fulfilled, we have $\delta_{I_1,J_1} \cdot \delta_{I_2,J_2} = 0$.

- An easy example: When $n = 5$, $\delta_{12,345} \cdot \delta_{13,245} = 0$ but $\delta_{12,345}$ and $\delta_{123,45}$ does not fulfill this relation.

Keel's quadratic relation

- Inspired by this property, we know that if any two factors of the monomial fulfills this relation, the whole integral will be zero.
- Now we only need to focus on those monomials where no two factors fulfill this quadratic relation, we call those monomials **tree monomial**.
- Since there is a one-to-one correspondence between these monomials and a type of tree, which we define as **loaded tree**.

loaded tree

A **loaded tree with** n **labels and** k **edges** is a tree (V, E, h, m), where h denotes the labeling function from V to the power set of N and m denotes the multiplicity function for edges. The following conditions must hold:

- Non-empty labels $\{h(v)\}_{v \in V}$ form a partition of N;
- Number of edges is k, edges are counted with multiplicity, i.e., $\sum_{e \in E} m(e) = k$;
- $\deg(v) + |h(v)| \geq 3$ holds for every $v \in V$.

(Hint: This loaded tree would correspond to a monomial in $A^k(n)$.)

weighted tree

- Given a loaded tree $LT = (V, E, h, m)$.
- We define its corresponding **weighted tree** $WT = (V, E, w)$ by attaching a weight function to each vertex and edge.
- $w(e) := m(e) - 1$ and $w(v) := \deg(v) + |h(v)| - 3$.
- Assume $WT = (V, E, w)$ is **a weighted tree of some loaded tree with n labels and $n-3$ edges**, then we can verify the following identity about the weight function w.
- $\sum_{v \in V} w(v) = \sum_{e \in E} w(e)$.

loaded tree: examples

Figure: This is a loaded tree with 5 labels and 2 edges.

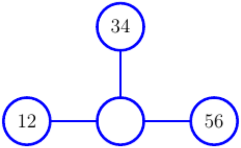


Figure: This is a loaded tree with 6 labels and 3 edges.

monomial of a given tree

- We define the **monomial of a given loaded tree** as the following:
- Remove an edge e, we collect the labels in the two connected components respectively to form I and J. And we say $\{I, J\}$ is the corresponding cut for the edge e.
- The monomial of this given loaded tree is $\prod_{i=1}^{m} \delta_{I_i, J_i}$, where m is the number of edges.
- Each edge of the tree contributes to the monomial a factor $\delta_{I,J}$ if $\{I, J\}$ is the corresponding cut for this edge.
- We can see that it is well-defined and each loaded tree has a unique monomial representation.

monomial of a given tree

Figure: This is a loaded tree with 5 labels and 2 edges. Its corresponding monomial: $\delta_{12,345} \cdot \delta_{123,45}$.

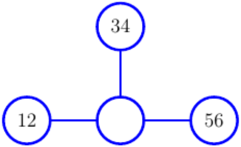


Figure: This is a loaded tree with 6 labels and 3 edges. Its corresponding monomial: $\delta_{34,1256} \cdot \delta_{12,3456} \cdot \delta_{56,1234}$.

one-to-one correspondence

Theorem

There is a one to one correspondence between tree monomials $T = \prod_{i=1}^{m} \delta_{I_i,J_i} (1 \leq m \leq n-3)$ *and loaded trees with* n *labels and* m *edges.*

- We also have an algorithm converting the monomial to tree, we call it **tree algorithm**.
- We will not go into details of this algorithm in today's talk.

the calculus (first half)

- Input: $M := \prod_{i=1}^{n-3} \delta_{I_i,J_i}$. (any monomial in $A^{n-3}(n)$)
- Output: the integral value of the given monomial, $\int(\prod_{i=1}^{n-3} \delta_{I_i,J_i})$, which is an integer.
- Step 1: Check if any two factors of M fulfills Keel's quadratic relation. If yes, return 0, terminate the process. Otherwise, continue.
- Step 2: Apply tree algorithm to the monomial, transfer it to a loaded tree (with n labels and $n-3$ edges).

the calculus – second half

- Input: a loaded tree LT with n labels and $n - 3$ edges.
- Output: the integral value of its corresponding monomial, which is an integer.
- This half mainly contains two parts, one for the absolute value and one for the sign.
- We will show it with a running example.

the calculus – second half: running example

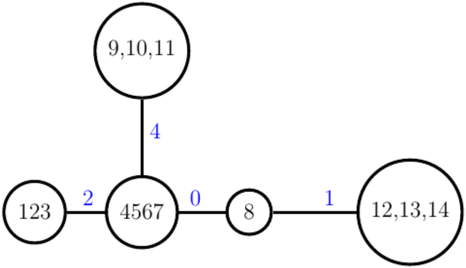


Figure: This is a loaded tree LT with 14 labels and 11 edges.

- **Step 1: Transfer it to a weighted tree.**
- Recall: $w(e) := m(e) - 1$ and $w(v) := \deg(v) + |h(v)| - 3$.

running example: weighted tree

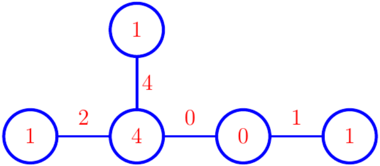


Figure: This is the weighted tree WT of the loaded tree LT, where the weight of vertices and edges are tagged in red.

Step 2: Compute the sign, which is $(-1)^S$. Here S denotes the weight sum of vertices (or equivalently, of edges) of WT.

running example: sign

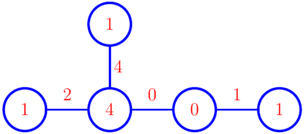


Figure: This is the weighted tree of the loaded tree LT, where the weight of vertices and edges are tagged in red.

Sum of vertex weight $S = 1 + 4 + 1 + 0 + 1 = 7$, so the sign of the monomial value is $(-1)^7 = -1$.

redundancy tree

- Step 3: Replace each edge by a length-two edge with a new vertex connecting them which has the same weight as the replaced edge.
- Then we obtain the redundancy tree RT (of loaded tree LT).

running example: redundancy tree

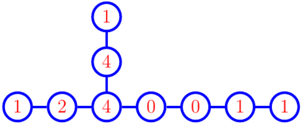


Figure: This is the redundancy tree RT of loaded tree LT, the weight of vertices are tagged in red.

Step 4: Omit those vertices with weight zero and their adjacent edges, we obtain the redundancy forest of LT**.**

running example: redundancy forest

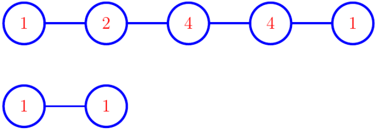


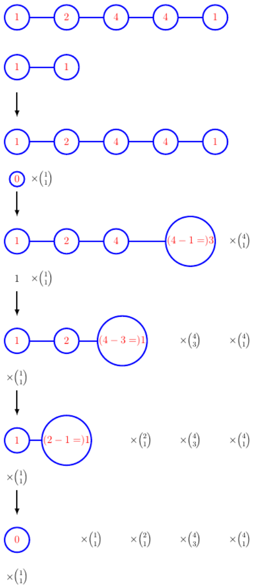
Figure: This is the redundancy forest *RF* of loaded tree *LT*, which contains two trees and the weight of vertices of are tagged in red.

Step 5: Apply a recursive algorithm to the redundancy forest, obtaining the absolute value (of the integral value).

recursive algorithm?

- Let $RF = (V, E, w)$ be the redundancy forest of a loaded tree LT.
- We define the value of RF as the following:
- Pick any leaf of this forest, say $l \in V$, denote the unique parent of l as l_1.
- If $w(l) > w(l_1)$, return 0 and terminate the process; otherwise, remove l from RF and assign weight $(w(l_1) - w(l))$ to l_1, replacing its previous weight. Denote the new forest as RF_1.
- Value of RF is the product of binomial coefficient $\binom{w(l_1)}{w(l)}$ and the value of RF_1.
- Base cases: whenever we reach a degree-zero vertex, if it has non-zero weight, return 0 and terminate the process; otherwise, return 1.
- Value of RF is then the absolute value of LT.

running example: absolute value



running example: integral value

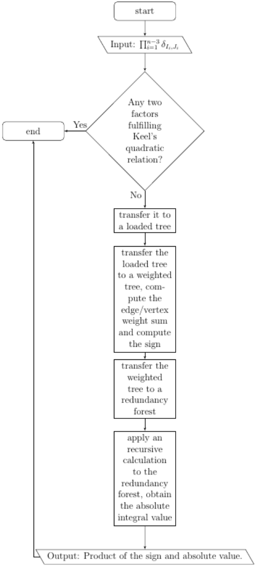
- Finally we get the absolute value as $1 \times \binom{1}{1} \times \binom{2}{1} \times \binom{4}{3} \times \binom{4}{1} \times \binom{1}{1} = 32$.
- Combining with the sign -1, we obtain the value of LT as -32.

Step 6: Product of the sign and absolute value gives us tree value.

the calculus – second half

- Input: a loaded tree with n labels and $n-3$ edges.
- Output: the integral value of the given loaded tree.
- Transfer the loaded tree to a weighted tree.
- Calculate the sign of the integral value.
- Transfer the weighted tree to a redundancy forest.
- Apply the recursive algorithm to this redundancy forest, obtaining the absolute integral value.
- Product of the sign and absolute value gives us the integral value.

the calculus – flow chart



Reference

Jiayue Qi.
A graphical algorithm for the integration of monomials in the Chow ring of the moduli space of stable marked curves of genus zero. preprint arXiv:2102.03575

Thank You