A calculus for monomials in Chow group of zero cycles in the moduli space of stable curves

Doctoral Program Computational Mathematics
Numerical Analysis and Symbolic Computation

FWF Der Wissenschaftsfonds.

Jiayue Qi

The research was funded by the Austrian Science Fund (FWF): W1214-N15, project DK9.

JOHANNES KEPLER UNIVERSITY LINZ

LAND OBERÖSTERREICH

Motivation

The problem originally showed up as a sub-problem for counting the realization of Laman graphs (minimally-rigid graphs) on a sphere [1], when we wanted to improve the algorithm given in [1]. With the help of this algorithm, we invented another algorithm for the same goal. Besides, we see this problem fundamental, standing on its own and we find the algorithm elegant and concise, also may be helpful for other similar or even further-away problems. Therefore, we formulate it on its own. The algorithm was published as an extended abstract, see [2].

Problem

Let $n \in \mathbb{N}$, $n \geq 3$, set $N := \{1, \ldots, n\}$. $\mathcal{M}_n$ denotes the moduli space of stable n-pointed curves of genus zero. A bipartition $\{I, J\}$ of N where both cardinalities of I and J are at least 2 is called a **cut**; I and J are **parts** of this cut. For every cut $\{I, J\}$, there is a variety $D_{I,J}$ in $\mathcal{M}_n$; denote by $\delta_{I,J}$ its corresponding element in the Chow ring. It is a graded ring — denote it as $A^*(n)$ — we have $A^*(n) = \bigoplus_{r=0}^{n-3} A^r(n)$. These homogeneous components are defined as Chow groups (of M_n); $A^r(n)$ is the **Chow group of rank** r. It is known that $A^r(n) = \{0\}$ for $r > n-3$ and $A^{n-3}(n) \cong \mathbb{Z}$. We denote this isomorphism by $\int : A^{n-3}(n) \to \mathbb{Z}$.

The set $\{\delta_{I,J} \mid \{I, J\}$ is a cut $\}$ generates group $A^1(n)$, and also the whole ring $A^*(n)$ (when they are used as ring generators). Then, $\prod_{i=1}^{n-3} \delta_{I_i,J_i}$ can be viewed as an element in $A^{n-3}(n)$. Let $M := \prod_{i=1}^{n-3} \delta_{I_i,J_i}$, we define the **value of** M to be $\int(\prod_{i=1}^{n-3} \delta_{I_i,J_i})$. **In this paper we calculate the value of a given monomial $M = \prod_{i=1}^{n-3} \delta_{I_i,J_i}$.**

Preperation

We say that the two generators $\delta_{I_1,J_1}, \delta_{I_2,J_2}$ fulfill **Keel's quadratic relation** [3] if the following four conditions hold: $I_1 \cap I_2 \neq \emptyset$; $I_1 \cap J_2 \neq \emptyset$; $J_1 \cap I_2 \neq \emptyset$; $J_1 \cap J_2 \neq \emptyset$. In this case, $\delta_{I_1,J_1} \cdot \delta_{I_2,J_2} = 0$.

Hence we only need to consider the monomials where no two factors fulfill Keel's quadratic relation; we call this type of monomials **tree monomials** since there is a one-to-one correspondence between these monomials and *loaded trees*.

Definition

A **loaded tree with n labels and k edges** is a tree (V, E) together with a labeling function $h : V \to 2^N$ and an edge multiplicity function $m : E \to \mathbb{N}^+$ such that the following three conditions hold:

1. $\{h(v)\}_{v \in V, h(v) \neq \emptyset}$ form a partition of N;
2. $\sum_{e \in E} m(e) = k$;
3. For every $v \in V$, $\deg(v) + |h(v)| \geq 3$, note that here multiple edges are only counted once for the degree of its incident vertices.

Theorem

There is a one-to-one correspondence between tree monomials $M = \prod_{i=1}^{k} \delta_{I_i,J_i}$ and loaded trees with n labels and k edges, where $I_i \cup J_i = N$ for all $1 \leq i \leq k$.

We have an algorithm tranferring in between these two representations, but we cannot go into details in this poster.

Theorem

If all factors are distinct in the tree monomial $M = \Pi_{i=1}^{n-3} \delta_{I_i,J_i}$, then $\int(M) = 1$. We call this type of tree monomials **clever monomials**.

With the help of the above theorems, we can illustrate our algorithm, which we will call the "forest algorithm", since it applies a recursive formula to a forest.

Algorithm

Given a loaded tree $LT = (V, E, h, m)$. Define a weight function $w : V \cup E \to \mathbb{N}$ as follows: For any $v \in V$, $w(v) := \deg(v) + |h(v)| - 3$. For any $e \in E$, $w(e) := m(e) - 1$. Then the **semi-redundancy tree** (of LT) is $SRT := (V, E, w)$. Start from this semi-redundancy tree, let S be the sum of vertex weight (or edge weight) of LT. Then the **sign of the tree value** of loaded tree LT is $(-1)^S$.

In the SRT, replace each edge by a length-two edge with a new vertex connecting them carrying the same weight as the replaced edge. Then we obtain the redundancy tree (of loaded tree LT) $RT := (V \cup E, E_1, w_1)$. Deleting the weight-zero vertices and their adjacent edges from RT gives us the **redundancy forest**.

Let $RF = (V, E, w)$ be the redundancy forest of a loaded tree LT. We define the **value of** RF as follows: Pick any leaf $l \in V$ of this forest, 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 the weight $(w(l_1) - w(l))$ to l_1, replacing its previous weight. Denote this new forest as RF_1. The value of RF is the product of binomial coefficient $\binom{w(l_1)}{w(l)}$ and the value of RF_1. Whenever we reach a degree-zero vertex, if it has non-zero weight, return 0 and terminate the process; otherwise return 1. Product of the value of the corresponding redundancy forest of LT and sign of its tree value gives us the value of LT.

the forest algorithm

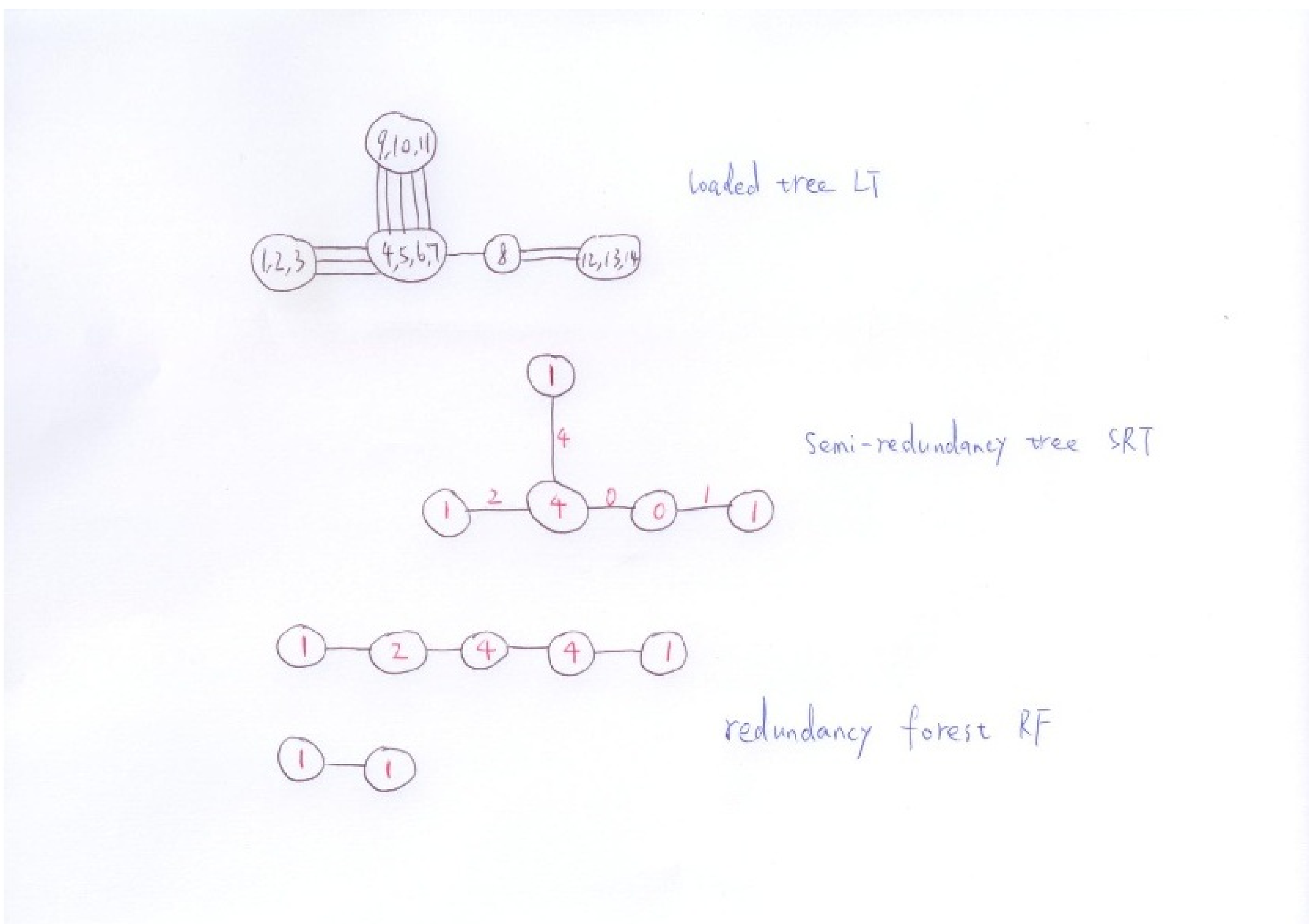
- **Given:** a monomial M in $A^{n-3}(n)$.
- **Output:** an integer which is the value of M.

1. check if M is a tree monomial. If no, return 0; if yes, continue.
2. check if M is a clever monomial. If yes, return 1; if no, continue.
3. transfer M to its corresponding loaded tree T_M.
4. transfer the loaded tree to a *semi-redundancy tree*.
5. calculate the *sign of the tree value*.
6. construct a *redundancy forest* from the semi-redundancy tree.
7. apply a recursive algorithm to this redundancy forest, obtaining the absolute tree value.
8. product of the sign and absolute value gives us the value of M.

Example

Example

Given a loaded tree LT. Follow the definition (construction) of semi-redundancy tree, we obtain the semi-redundancy tree SRT. Sum of vertex weight $S = 1+4+1+0+1 = 7$, so the sign of LT value is $(-1)^7 = -1$. From SRT, follow the construction for the redundancy forest, we obtain the redundancy forest RF. Finally we get the absolute value of RF as $[\binom{1}{1} \times 1] \times [\binom{2}{1} \times \binom{4}{1} \times \binom{4}{3} \times \binom{1}{1} \times 1] = 32$. Combining with the sign -1, we obtain the value of LT as -32.



References

[1] M. Gallet, G. Grassegger, J. Schicho. Counting realizations of Laman graphs on the sphere. *The Electronic Journal of Combinatorics*, Volume 27, Issue 2 (2020).

[2] Jiayue Qi. A calculus for monomials in Chow group of zero cycles in the moduli space of stable curves. *ACM Communications in Computer Algebra*, 54.3 (2021): 91-94.

[3] Sean Keel. Intersection theory of moduli space of stable n-pointed vurves of genus zero. *Transaction of the American Mathematical Society*, **330** (1992), no. 2, 545-574.