Origami

Computing Veechgroups of origamis

1.0

26 February 2018

Pascal Kattler

Sergio Siccha

Andrea Thevis

Pascal Kattler
Email: kattler@math.uni-sb.de
Homepage: http://www.math.uni-sb.de/ag/weitze/
Address: AG Weitze-Schmithüsen
FR 6.1 Mathematik
Universität des Saarlandes
D-66041 Saarbrücken

Sergio Siccha
Email: siccha@mathb.rwth-aachen.de
Homepage: https://www.mathb.rwth-aachen.de/Mitarbeiter/siccha.php
Address: Lehrstuhl B für Mathematik RWTH - Aachen
Pontdriesch 10-16
52062 Aachen
Germany

Andrea Thevis
Email: thevis@math.uni-sb.de
Homepage: http://www.math.uni-sb.de/ag/weitze/
Address: AG Weitze-Schmithüsen
FR 6.1 Mathematik
Universität des Saarlandes
D-66041 Saarbrücken

Contents

Chapter 1

The Origami object

1.1 The action on the Origami

1.1.1 ActionOfSl

▷ `ActionOfSl(word, Origami)` (function)

Returns: the Origami Object word.Origami

This lets act a word in the free group $Group(S,T)$,representing an element of $Sl_2(\mathbb{Z})$ on an Origami and returns $word.Origami$.

1.1.2 ActionOfF2ViaCanonical

▷ `ActionOfF2ViaCanonical(word, Origami)` (function)

Returns: the Origami Object word.Origami

This lets act a word in the free group $Group(S,T)$, representing an element of $Sl_2(\mathbb{Z})$, on an Origami and returns word.Origami. But in contrast to "ActionOfSl" the result is stored in the canonical representation.

1.1.3 RightActionOfF2ViaCanonical

▷ `RightActionOfF2ViaCanonical(word, Origami)` (function)

Returns: the Origami Object Origami.word

This lets act a word in the free group $Group(S,T)$ on an Origami from right and returns $Origami.word = word^{-}1.Origami$, where the left action is the common action of $Sl_2(\mathbb{Z})$ on 2 mannifolds. This action has the same Veechgroup and orbits as the left action. In contrast to "ActionOfSl" the result is stored in the canonical representation.

1.2 The Origami object

1.2.1 CanonicalOrigamiViaDelecroixAndStart

▷ `CanonicalOrigamiViaDelecroixAndStart(Origami, start)` (function)

Returns: An Origami

This calculates a canonical representation of an origami depending on a given number start (Between 1 and the degree of of the Origami). To determine a canonical numbering the algorithm starts at the square with number start and walks over the origami in a certain way and numbers the squares in the order, they are visited . First it walks in horizontal direction one loop. Then it walks one step up (in vertical direection) and then again a loop in horizontal direction. This wil be repeated until the vertical loop is complete or all squares have been visited. If there are unvisited squares, we continue with the smallest number (in the new numbering), that has not been in a vertical loop. An Origami is connected, so that number exists. Two origamis are equal if they are described by the same permutations in their canonical representation.

1.2.2 CanonicalOrigamiViaDelecroix

▷ `CanonicalOrigamiViaDelecroix(`*`Origami`*`)` (function)

Returns: An Origami

This calculates a canonical representation of an origami. It calculates the representation from CanonicalOrigamiViaDelecroixAndStart with several start squares, independent of the given representation. Then it takes the minimum with respect to some order. Two origamis are equal if they are described by the same permutations in their canonical representation.

1.2.3 CanonicalOrigami

▷ `CanonicalOrigami(`*`Origami`*`)` (function)

Returns: An Origami

This calculates a canonical representation of an origami, represented as record rec(d := * , x := *, y := *). Two origamis are equal if they are described by the same permutations in their canonical representation.

1.2.4 VerticalPerm (for IsOrigami)

▷ `VerticalPerm(`*`Origami`*`)` (attribute)

Returns: a permutation

This returns the horizontal permutation σ_x of the Origami.

1.2.5 HorizontalPerm (for IsOrigami)

▷ `HorizontalPerm(`*`Origami`*`)` (attribute)

Returns: a permutation

This returns the vertical permutation σ_y of the Origami.

1.2.6 DegreeOrigami (for IsOrigami)

▷ `DegreeOrigami(`*`Origami`*`)` (attribute)

Returns: an integer

This returns the degree of an Origami.

1.2.7 Stratum (for IsOrigami)

▷ `Stratum(`*`Origami`*`)` (attribute)

Returns: a list of integers

This calculates the stratum of an Origami. That is a list of the orders of the singularities.

1.2.8 VeechGroup (for IsOrigami)

▷ `VeechGroup(Origami)` (attribute)

Returns: a ModularSubgroup object

This calculates the Veechgroup of an Origami. This is a subgroup of $Sl_2(\mathbb{Z})$ of finite degree. The group is stored as ModularSubgroup from the ModularSubgroup package. The Veechgroup is represented as the coset permutations σ_S and σ_T with respect to the generators S and T. This means if i is the integer associated to the rigth coset G (Cosets(O) [i] VeechGroup = H) then we have for the coset H, associated to $\sigma_S(i)$, that $SG = H$. Dito for σ_T.

1.2.9 Cosets (for IsOrigami)

▷ `Cosets(Origami)` (attribute)

Returns: a list of words in the Free group, generated by S and T.

This Calculates the right cosets of the Veechgroup of an Origami.

1.2.10 Equals

▷ `Equals(Origami1, Origami2)` (function)

Returns: true or false

This tests wether Origami1 is equal to Origami2 with same numbering of squares. That is, the defining permutations are the same.

1.2.11 EquivalentOrigami

▷ `EquivalentOrigami(Origami1, Origami2)` (function)

Returns: true or false

This tests wether Origami1 is equal up to Origami2 up to numbering of the squares.

1.2.12 ExampleOrigami

▷ `ExampleOrigami(d)` (function)

Returns: a random origami

This creates a random origami of degree d.

1.2.13 CalcVeechGroup

▷ `CalcVeechGroup(Origami)` (function)

Returns: A list with tree entrys

This function is used to calculate some attributes. It calculates the Veechgroup of a given origami and . the veechgroup is stored as ModularGroup Object from the ModularGroup package. The cosets of the veechgroup is stored in a list of words in the generators S and T of the matrix group Sl_2(Z).

1.2.14 CalcVeechGroupViaEquivalentTest

▷ `CalcVeechGroupViaEquivalentTest(Origami)` (function)

Returns: A list with tree entrys

This function is used to calculate some attributes. It calculates the Veechgroup of a given origami and . the veechgroup is stored as ModularGroup Object from the ModularGroup package. The cosets of the veechgroup is stored in a list of words in the generators S and T of the matrix group Sl_2(Z). In Contrast to CalcVeechGroup, this uses equivalent tests instead of canonical Origamis.

1.2.15 CalcVeechGroupWithHashTables

▷ `CalcVeechGroupWithHashTables(`*`Origami`*`)` (function)

Returns: A list with tree entrys

This function is used to calculate some attributes. It calculates the Veechgroup of a given origami and . the veechgroup is stored as ModularGroup Object from the ModularGroup package. The cosets of the veechgroup is stored in a list of words in the generators S and T of the matrix group Sl_2(Z). In contrast to CalcVeechGroup, this uses hash tables to store Origamis.

1.2.16 CalcStratum

▷ `CalcStratum(`*`Origami`*`)` (function)

Returns: nothing

Calculates the stratum of an object and sets its attribute. The stratum is stored as list of integers.

1.2.17 ToRec

▷ `ToRec(`*`Origami`*`)` (function)

Returns: record of the form rec(d := * , x := *, y := *) Describtion This calculates a record representation for an origami object.

Chapter 2

Introduction

This package provides calculations with Origamis. An Origami can be obtained in the following way from two permutations $\sigma_a, \sigma_b \in S_d$. We take d Squares $Q_1, \ldots, Q_d$ and clue the lower side of Q_i to the upper side of $Q_{\sigma_y(i)}$ and the right side of Q_i to the left side of $Q_{\sigma_x(i)}$. So in this Package we identify an Origami with a pair of permutations, witch acts transitive on $\{1 \ldots d\}$ up to simultan conjugation. We store an Origami as Origami object. We are mainly interested in the Veechgroup of an Origami. It can be shown that the Veechgroup of an Origami is a subgroup of $Sl_2(\mathbb{Z})$ of finite index. So we fix two generators

$$S = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

and

$$T = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

2.1 The Free Group

In this package we fix the Free Group F generated by S and T.

Index