Massey Products for Graph Homology

Ben Ward

Bowling Green State University

Warwick Geometry and Topology Online Seminar
November 11, 2021.

Recollection of A_∞-algebras

Definition: The associahedron K_n is a polytope of dimension $n-2$ such that:

Recollection of A_∞-algebras

Definition: The associahedron K_n is a polytope of dimension $n-2$ such that:

- vertices = ways to parenthesize a string of n letters,

Recollection of A_∞-algebras

Definition: The associahedron K_n is a polytope of dimension $n-2$ such that:

- vertices $=$ ways to parenthesize a string of n letters,
- edges $=$ strings of n letters missing one pair of parentheses,

Recollection of A_∞-algebras

Definition: The associahedron K_n is a polytope of dimension $n-2$ such that:

- vertices = ways to parenthesize a string of n letters,
- edges = strings of n letters missing one pair of parentheses,
- faces = strings of n letters missing two pair of parentheses

Recollection of A_∞-algebras

Definition: The associahedron K_n is a polytope of dimension $n-2$ such that:

- vertices = ways to parenthesize a string of n letters,
- edges = strings of n letters missing one pair of parentheses,
- faces = strings of n letters missing two pair of parentheses and so on,
- adjacency is determined by adding parentheses.

Recollection of A_∞-algebras

Definition: The associahedron K_n is a polytope of dimension $n-2$ such that:

- vertices = ways to parenthesize a string of n letters,
- edges = strings of n letters missing one pair of parentheses,
- faces = strings of n letters missing two pair of parentheses and so on,
- adjacency is determined by adding parentheses.

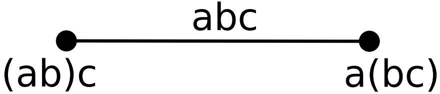
Case $n = 3$:

Recollection of A_∞-algebras

Definition: The associahedron K_n is a polytope of dimension $n-2$ such that:

- vertices = ways to parenthesize a string of n letters,
- edges = strings of n letters missing one pair of parentheses,
- faces = strings of n letters missing two pair of parentheses and so on,
- adjacency is determined by adding parentheses.

Case $n=3$:



Recollection of A_∞-algebras

Example of K_4.

String of 4 letters

Polytope

abcd

Recollection of A_∞-algebras

Example of K_4.

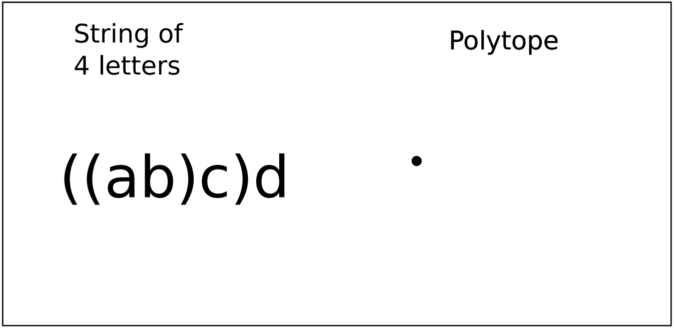
String of
4 letters

Polytope

((ab)c)d

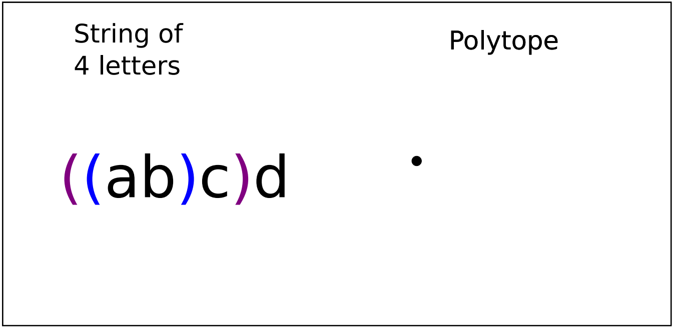
Recollection of A_∞-algebras

Example of K_4.



Recollection of A_∞-algebras

Example of K_4.



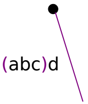
Recollection of A_∞-algebras

Example of K_4.

String of
4 letters

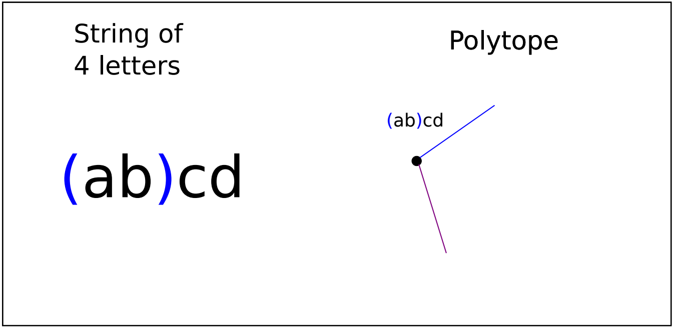
(abc)d

Polytope



Recollection of A_∞-algebras

Example of K_4.



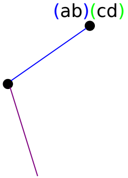
Recollection of A_∞-algebras

Example of K_4.

String of
4 letters

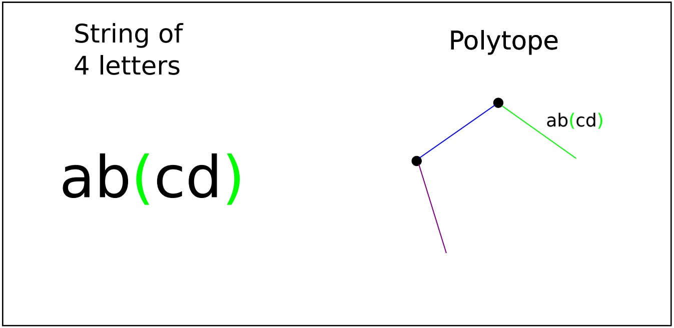
(ab)(cd)

Polytope



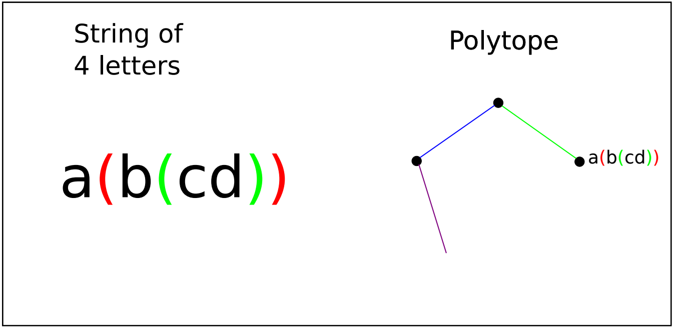
Recollection of A_∞-algebras

Example of K_4.



Recollection of A_∞-algebras

Example of K_4.



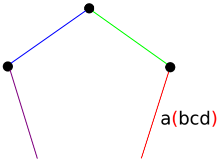
Recollection of A_∞-algebras

Example of K_4.

String of
4 letters

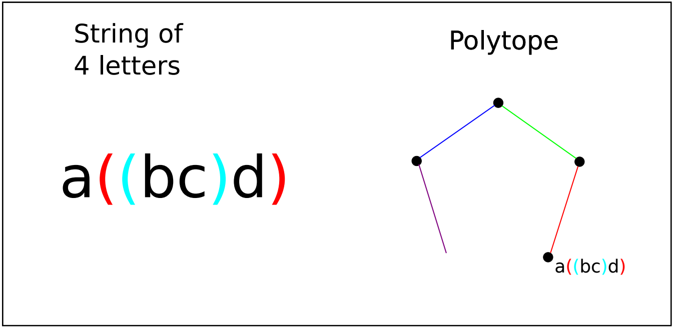
a(bcd)

Polytope



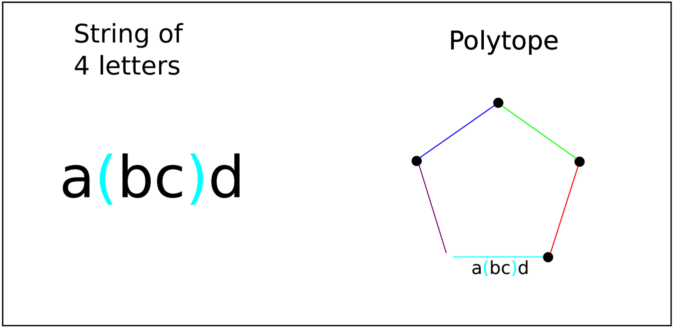
Recollection of A_∞-algebras

Example of K_4.



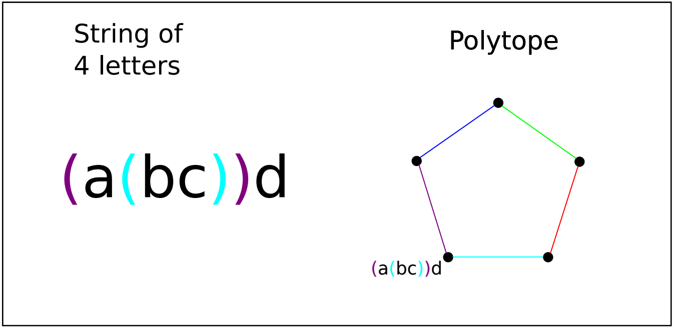
Recollection of A_∞-algebras

Example of K_4.



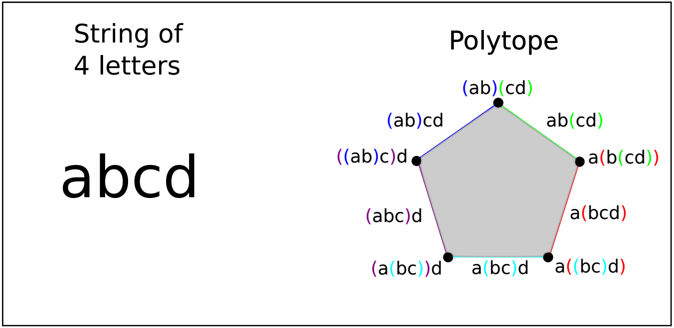
Recollection of A_∞-algebras

Example of K_4.



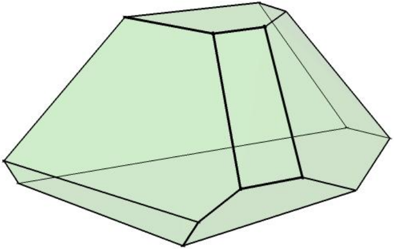
Recollection of A_∞-algebras

Example of K_4.



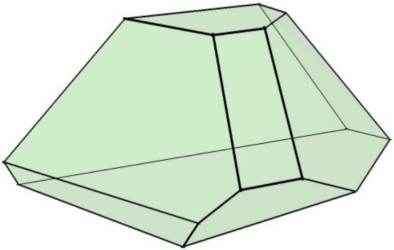
Associahedra

Example: K_5



Associahedra

Example: K_5



Example: $K_2 = \bullet$

Associahedra encode A_∞-algebras

Informal **Definition**:

Associahedra encode A_∞-algebras

Informal **Definition**: An A_∞ algebra is a chain complex A with a compatible linear map $A^{\otimes n} \to A$ for every cell in K_n (for each n).

Associahedra encode A_∞-algebras

Informal **Definition**: An A_∞ algebra is a chain complex A with a compatible linear map $A^{\otimes n} \to A$ for every cell in K_n (for each n).

What is needed to specify an A_∞-algebra?

Associahedra encode A_∞-algebras

Informal **Definition**: An A_∞ algebra is a chain complex A with a compatible linear map $A^{\otimes n} \to A$ for every cell in K_n (for each n).

What is needed to specify an A_∞-algebra?

K_2		need $\mu_2 \colon A^{\otimes 2} \to A$ of degree 0

Associahedra encode A_∞-algebras

Informal **Definition**: An A_∞ algebra is a chain complex A with a compatible linear map $A^{\otimes n} \to A$ for every cell in K_n (for each n).

What is needed to specify an A_∞-algebra?

<table>
<tr><td>K_2</td><td>
</td><td>need $\mu_2 \colon A^{\otimes 2} \to A$ of degree 0</td></tr>
<tr><td>K_3</td><td>
</td><td></td></tr>
</table>

Associahedra encode A_∞-algebras

Informal **Definition**: An A_∞ algebra is a chain complex A with a compatible linear map $A^{\otimes n} \to A$ for every cell in K_n (for each n).

What is needed to specify an A_∞-algebra?

K_2		need $\mu_2\colon A^{\otimes 2} \to A$ of degree 0
K_3		need $\mu_3 : A^{\otimes 3} \to A$ of degree 1

Associahedra encode A_∞-algebras

Informal **Definition**: An A_∞ algebra is a chain complex A with a compatible linear map $A^{\otimes n} \to A$ for every cell in K_n (for each n).

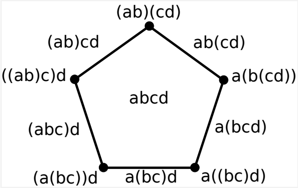
What is needed to specify an A_∞-algebra?

K_2		need μ_2: $A^{\otimes 2} \to A$ of degree 0
K_3		need μ_3 : $A^{\otimes 3} \to A$ of degree 1

Associahedra encode A_∞-algebras

Informal **Definition**: An A_∞ algebra is a chain complex A with a compatible linear map $A^{\otimes n} \to A$ for every cell in K_n (for each n).

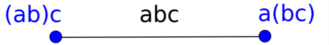
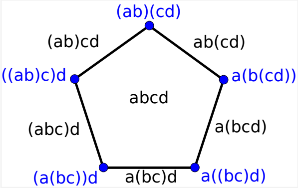
What is needed to specify an A_∞-algebra?

K_2		need $\mu_2 \colon A^{\otimes 2} \to A$ of degree 0
K_3		need $\mu_3 : A^{\otimes 3} \to A$ of degree 1
K_4		need $\mu_4 \colon A^{\otimes 4} \to A$ of degree 2

Associahedra encode A_∞-algebras

Informal **Definition**: An A_∞ algebra is a chain complex A with a compatible linear map $A^{\otimes n} \to A$ for every cell in K_n (for each n).

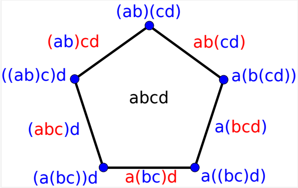
What is needed to specify an A_∞-algebra?

K_2		need $\mu_2 : A^{\otimes 2} \to A$ of degree 0
K_3		need $\mu_3 : A^{\otimes 3} \to A$ of degree 1
K_4		need $\mu_4 : A^{\otimes 4} \to A$ of degree 2

Associahedra encode A_∞-algebras

Informal **Definition**: An A_∞ algebra is a chain complex A with a compatible linear map $A^{\otimes n} \to A$ for every cell in K_n (for each n).

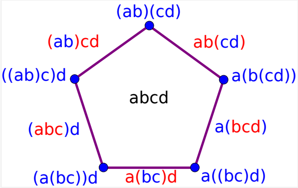
What is needed to specify an A_∞-algebra?

K_2		need $\mu_2 \colon A^{\otimes 2} \to A$ of degree 0
K_3		need $\mu_3 : A^{\otimes 3} \to A$ of degree 1
K_4		need $\mu_4 \colon A^{\otimes 4} \to A$ of degree 2

Associahedra encode A_∞-algebras

Informal **Definition**: An A_∞ algebra is a chain complex A with a compatible linear map $A^{\otimes n} \to A$ for every cell in K_n (for each n).

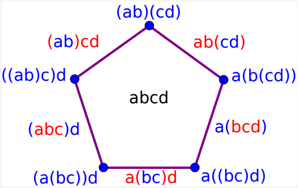
What is needed to specify an A_∞-algebra?

K_2		need $\mu_2 \colon A^{\otimes 2} \to A$ of degree 0
K_3		need $\mu_3 : A^{\otimes 3} \to A$ of degree 1
K_4		need $\mu_4 \colon A^{\otimes 4} \to A$ of degree 2

Associahedra encode A_∞-algebras

Informal **Definition**: An A_∞ algebra is a chain complex A with a compatible linear map $A^{\otimes n} \to A$ for every cell in K_n (for each n).

What is needed to specify an A_∞-algebra?

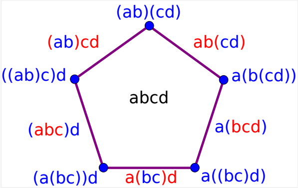
K_2		need $\mu_2 \colon A^{\otimes 2} \to A$ of degree 0
K_3		need $\mu_3 : A^{\otimes 3} \to A$ of degree 1
K_4		need $\mu_4 \colon A^{\otimes 4} \to A$ of degree 2

and so on...

Associahedra encode A_∞-algebras

Informal **Definition**: An A_∞ algebra is a chain complex A with a compatible linear map $A^{\otimes n} \to A$ for every cell in K_n (for each n).

What is needed to specify an A_∞-algebra?

K_2		need μ_2: $A^{\otimes 2} \to A$ of degree 0
K_3		need μ_3 : $A^{\otimes 3} \to A$ of degree 1
K_4		need μ_4: $A^{\otimes 4} \to A$ of degree 2

and so on... A_∞-algebra $= (A, \mu_2, \mu_3, \mu_4, \mu_5, \ldots)$

Why A_∞ algebras?

Theorem (Kadeishvili)

Let A be a dg associative algebra over a field of characteristic zero. There exists an A_∞ structure on $H_*(A)$ such that $A \sim H_*(A)$ as A_∞-algebras.

Why A_∞ algebras?

Theorem (Kadeishvili)

Let A be a dg associative algebra over a field of characteristic zero. There exists an A_∞ structure on $H_*(A)$ such that $A \sim H_*(A)$ as A_∞-algebras.

For example let $A = C^*(X; \mathbb{Q})$ with the cup product.

Why A_∞ algebras?

Theorem (Kadeishvili)

Let A be a dg associative algebra over a field of characteristic zero. There exists an A_∞ structure on $H_*(A)$ such that $A \sim H_*(A)$ as A_∞-algebras.

For example let $A = C^*(X;\mathbb{Q})$ with the cup product. This is a finer invariant than cohomology, $H^*(X;\mathbb{Q})$

Why A_∞ algebras?

Theorem (Kadeishvili)

Let A be a dg associative algebra over a field of characteristic zero. There exists an A_∞ structure on $H_*(A)$ such that $A \sim H_*(A)$ as A_∞-algebras.

For example let $A = C^*(X;\mathbb{Q})$ with the cup product. This is a finer invariant than cohomology, $H^*(X;\mathbb{Q})$ until we add higher operations

$$\mu_n \colon H^*(A)^{\otimes n} \to H^{*-n+2}(A).$$

Why A_∞ algebras?

Theorem (Kadeishvili)

Let A be a dg associative algebra over a field of characteristic zero. There exists an A_∞ structure on $H_*(A)$ such that $A \sim H_*(A)$ as A_∞-algebras.

For example let $A = C^*(X; \mathbb{Q})$ with the cup product. This is a finer invariant than cohomology, $H^*(X; \mathbb{Q})$ until we add higher operations

$$\mu_n \colon H^*(A)^{\otimes n} \to H^{*-n+2}(A).$$

Remarks:

- This result is called the homotopy transfer theorem.

Why A_∞ algebras?

Theorem (Kadeishvili)

Let A be a dg associative algebra over a field of characteristic zero. There exists an A_∞ structure on $H_*(A)$ such that $A \sim H_*(A)$ as A_∞-algebras.

For example let $A = C^*(X;\mathbb{Q})$ with the cup product. This is a finer invariant than cohomology, $H^*(X;\mathbb{Q})$ until we add higher operations

$$\mu_n \colon H^*(A)^{\otimes n} \to H^{*-n+2}(A).$$

Remarks:

- This result is called the homotopy transfer theorem.
- The homotopy transfer theorem is a corollary of the fact that the associative operad is Koszul.

Why A_∞ algebras?

Theorem (Kadeishvili)

Let A be a dg associative algebra over a field of characteristic zero. There exists an A_∞ structure on $H_*(A)$ such that $A \sim H_*(A)$ as A_∞-algebras.

For example let $A = C^*(X;\mathbb{Q})$ with the cup product. This is a finer invariant than cohomology, $H^*(X;\mathbb{Q})$ until we add higher operations

$$\mu_n\colon H^*(A)^{\otimes n} \to H^{*-n+2}(A).$$

Remarks:

- This result is called the homotopy transfer theorem.
- The homotopy transfer theorem is a corollary of the fact that the associative operad is Koszul. Koszulity of this operad is essentially the statement that associahedra are contractible.

Why A_∞ algebras?

Theorem (Kadeishvili)

Let A be a dg associative algebra over a field of characteristic zero. There exists an A_∞ structure on $H_*(A)$ such that $A \sim H_*(A)$ as A_∞-algebras.

For example let $A = C^*(X;\mathbb{Q})$ with the cup product. This is a finer invariant than cohomology, $H^*(X;\mathbb{Q})$ until we add higher operations

$$\mu_n \colon H^*(A)^{\otimes n} \to H^{*-n+2}(A).$$

Remarks:

- This result is called the homotopy transfer theorem.
- The homotopy transfer theorem is a corollary of the fact that the associative operad is Koszul. Koszulity of this operad is essentially the statement that associahedra are contractible.
- We will call these higher operations "Massey products".

Associativity revisited

Key feature of associahedra: they are contractible.

Associativity revisited

Key feature of associahedra: they are contractible.

- I claim this feature is present in much greater generality.

Associativity revisited

Key feature of associahedra: they are contractible.

- I claim this feature is present in much greater generality.

- Revisit associativity:

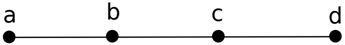
Associativity revisited

Key feature of associahedra: they are contractible.

- I claim this feature is present in much greater generality.

- Revisit associativity:

abcd =

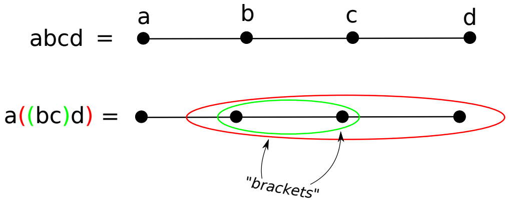


Associativity revisited

Key feature of associahedra: they are contractible.

- I claim this feature is present in much greater generality.

- Revisit associativity:

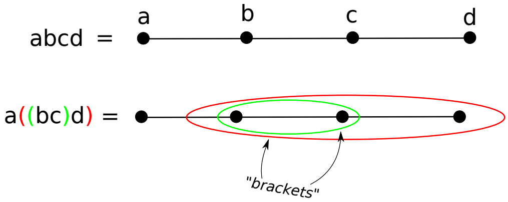


Associativity revisited

Key feature of associahedra: they are contractible.

- I claim this feature is present in much greater generality.

- Revisit associativity:



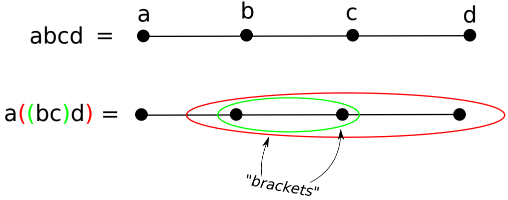
- brackets are either nested or disjoint

Associativity revisited

Key feature of associahedra: they are contractible.

- I claim this feature is present in much greater generality.

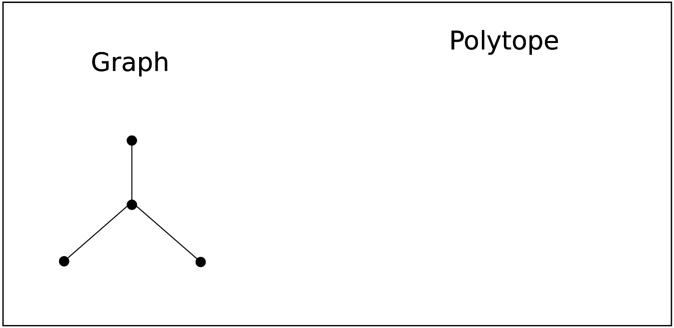
- Revisit associativity:



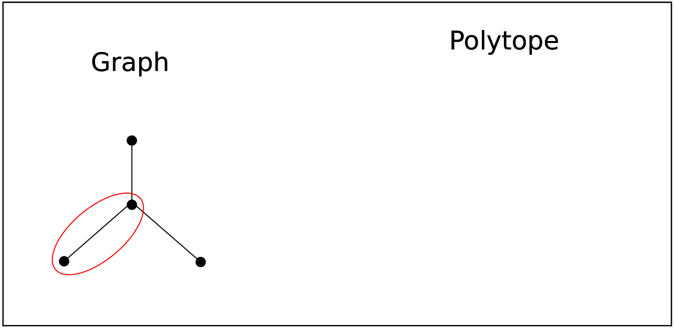
- brackets are either nested or disjoint

Lemma (W.) The space of bracketings of *any graph* is contractible, in fact it is a polytope.

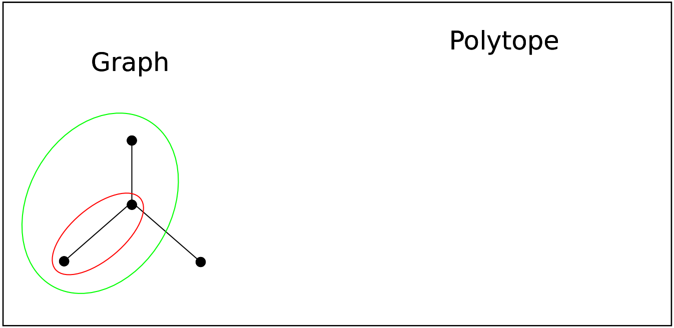
The space of bracketings of any graph is contractible



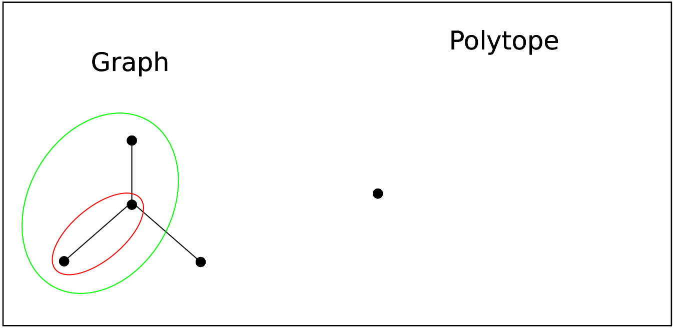
The space of bracketings of any graph is contractible



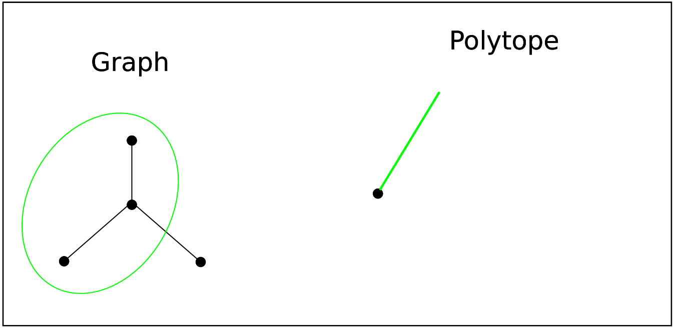
The space of bracketings of any graph is contractible



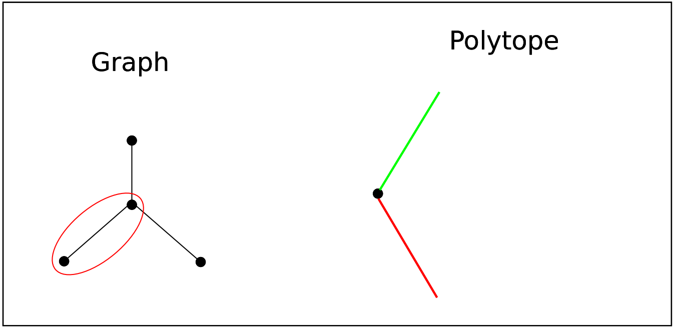
The space of bracketings of any graph is contractible



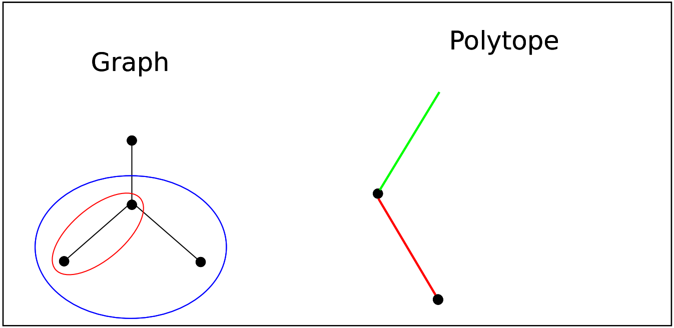
The space of bracketings of any graph is contractible



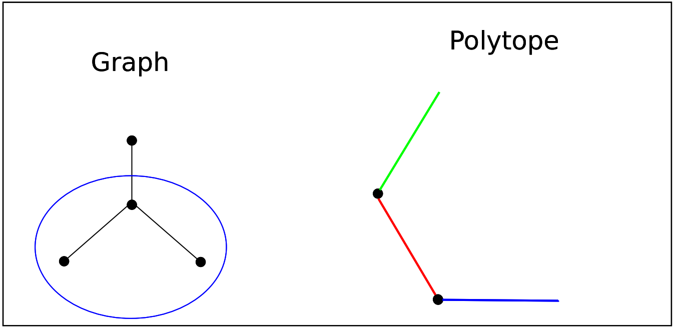
The space of bracketings of any graph is contractible



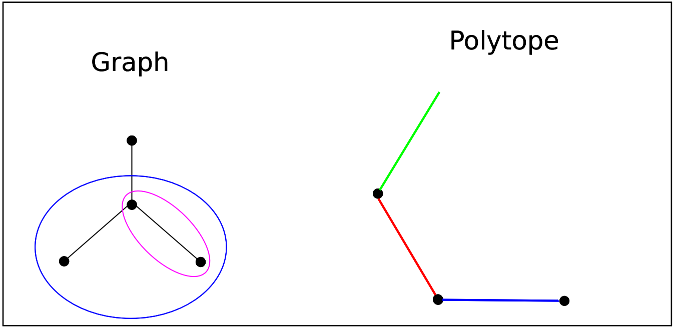
The space of bracketings of any graph is contractible



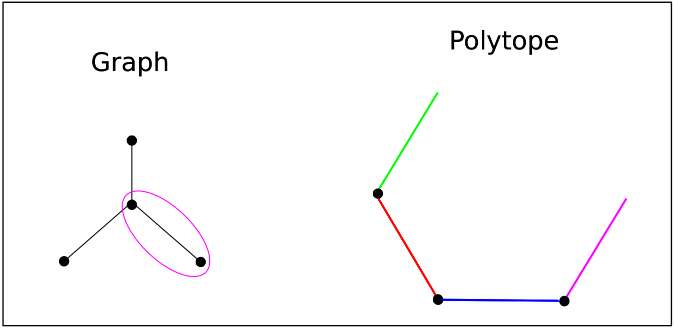
The space of bracketings of any graph is contractible



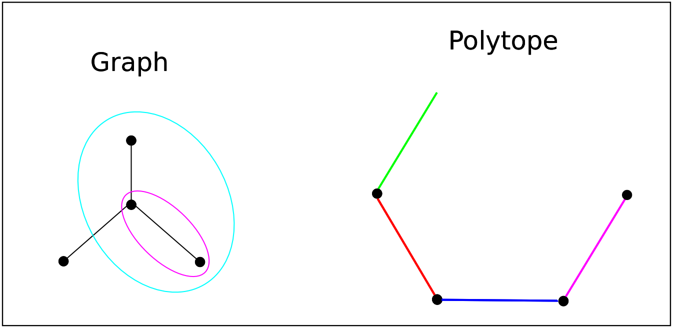
The space of bracketings of any graph is contractible



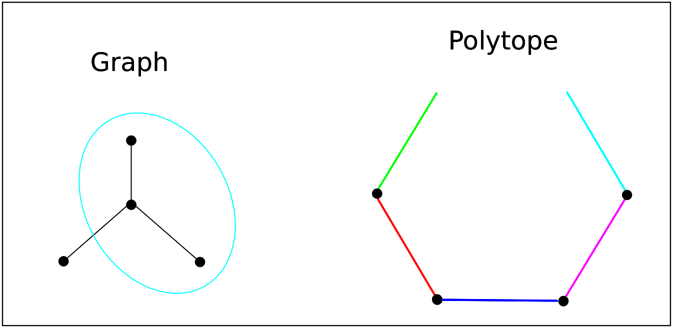
The space of bracketings of any graph is contractible



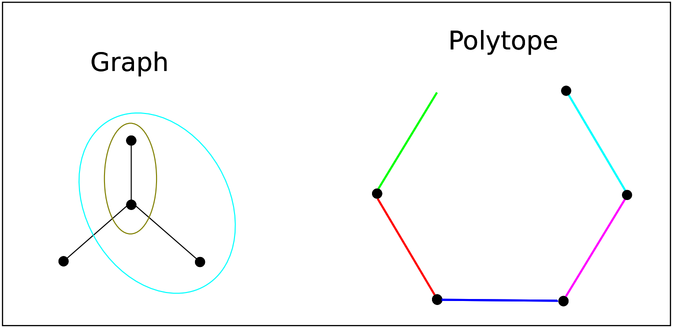
The space of bracketings of any graph is contractible



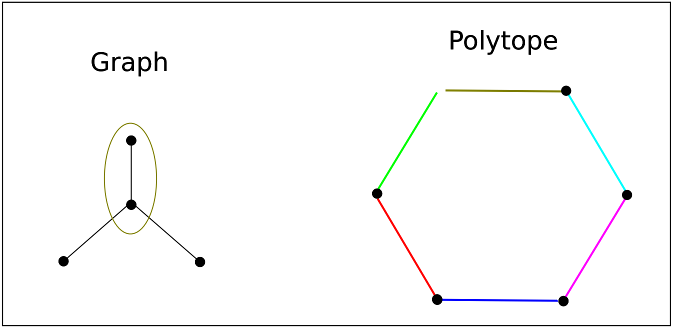
The space of bracketings of any graph is contractible



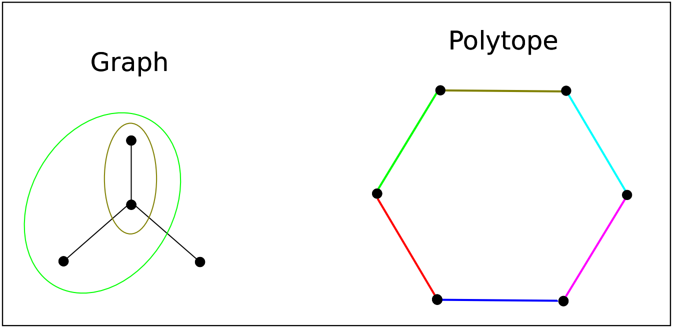
The space of bracketings of any graph is contractible



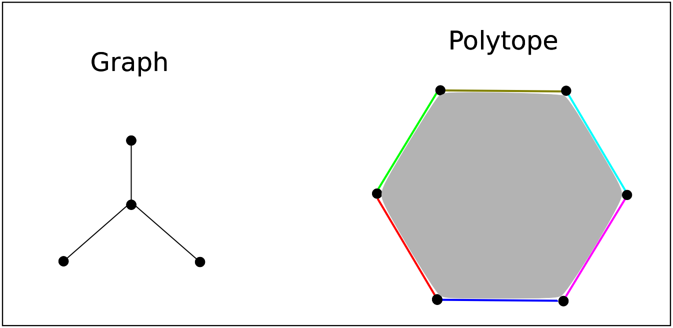
The space of bracketings of any graph is contractible



The space of bracketings of any graph is contractible



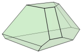
The space of bracketings of any graph is contractible



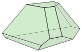
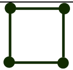
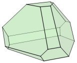
Bracketohedra???

Graph	Picture	Name

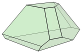
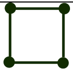
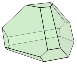
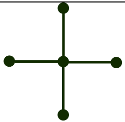
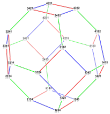
Bracketohedra???

Graph	Picture	Name
		Associahedron

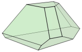
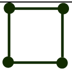
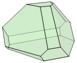
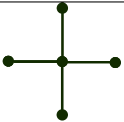
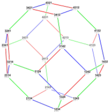
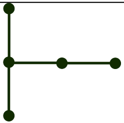
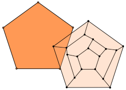
Bracketohedra???

Graph	Picture	Name
		Associahedron
		Cyclohedron

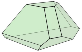
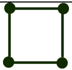
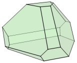
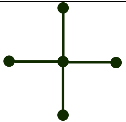
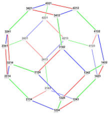
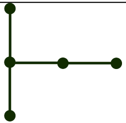
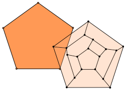
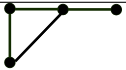
Bracketohedra???

Graph	Picture	Name
		Associahedron
		Cyclohedron
	 (thanks wikipedia)	Permutohedron

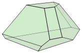
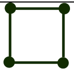
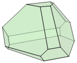
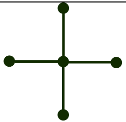
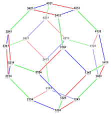
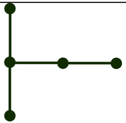
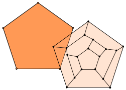
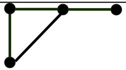
Bracketohedra???

Graph	Picture	Name
		Associahedron
		Cyclohedron
	 (thanks wikipedia)	Permutohedron
		????hedron

Bracketohedra???

Graph	Picture	Name
		Associahedron
		Cyclohedron
	 (thanks wikipedia)	Permutohedron
		????hedron
		????hedron

Bracketohedra???

Graph	Picture	Name
		Associahedron
		Cyclohedron
	 (thanks wikipedia)	Permutohedron
		????hedron
		????hedron

These are the only 3d Bracketohedra.

An analogy

How do we use this generalization?

An analogy

How do we use this generalization?

	then
Algebraic structure	Associativity
Combinatorics	Multiply along a line
Polytopes	Associahedra
Homotopy Transfer	via A_∞-algebras
use to study	Topological spaces

An analogy

How do we use this generalization?

	then	now
Algebraic structure	Associativity	
Combinatorics	Multiply along a line	Multiply along a graph
Polytopes	Associahedra	Bracketohedra
Homotopy Transfer	via A_∞-algebras	
use to study	Topological spaces	

An analogy

How do we use this generalization?

	then	now
Algebraic structure	Associativity	
Combinatorics	Multiply along a line	Multiply along a graph
Polytopes	Associahedra	Bracketohedra
Homotopy Transfer	via A_∞-algebras	
use to study	Topological spaces	

Present Goal: Fill in this table.

An analogy

How do we use this generalization?

	then	now
Algebraic structure	Associativity	Modular Operad
Combinatorics	Multiply along a line	Multiply along a graph
Polytopes	Associahedra	Bracketohedra
Homotopy Transfer	via A_∞-algebras	
use to study	Topological spaces	

Present Goal: Fill in this table.

Modular operads.

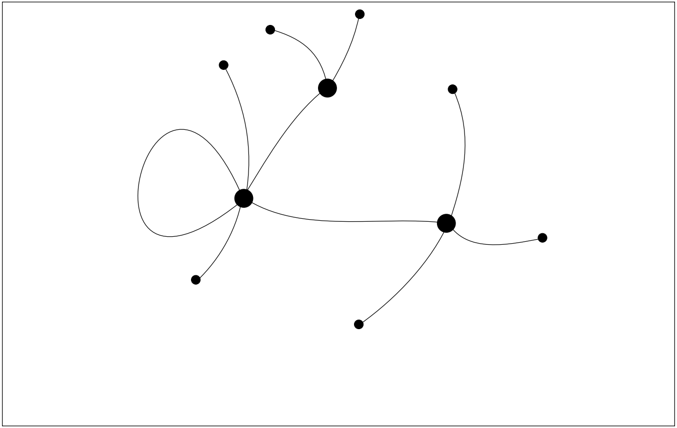
Informal **Definition:** A modular operad is a sequence of objects $(M_2, M_3, M_4, \dots)$

Modular operads.

Informal **Definition:** A modular operad is a sequence of objects $(M_2, M_3, M_4, \dots)$ and an algebraic operation for every graph:

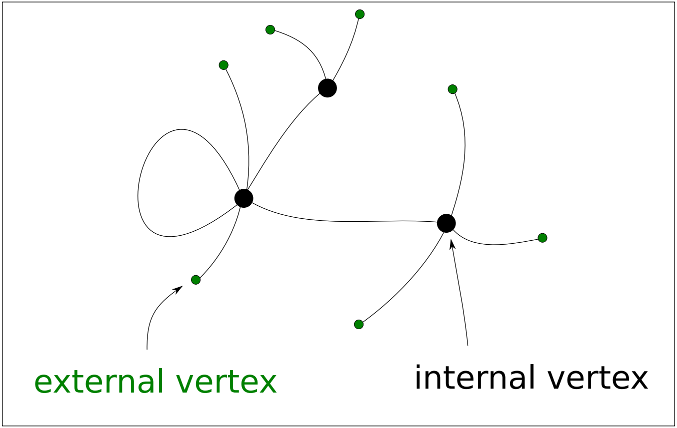
Modular operads.

Informal **Definition:** A modular operad is a sequence of objects $(M_2, M_3, M_4, \dots)$ and an algebraic operation for every graph:



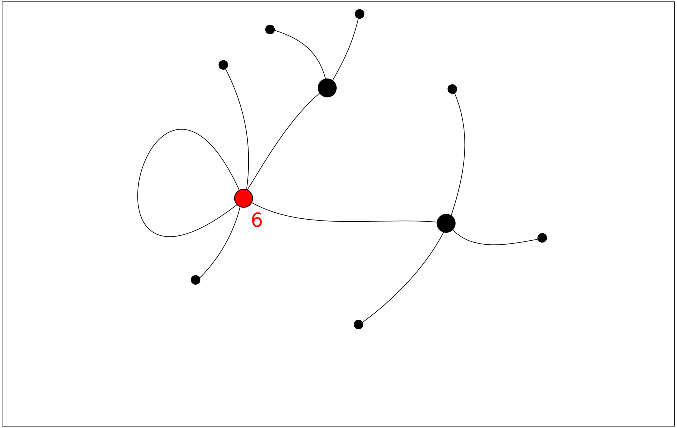
Modular operads.

Informal **Definition:** A modular operad is a sequence of objects $(M_2, M_3, M_4, \ldots)$ and an algebraic operation for every graph:



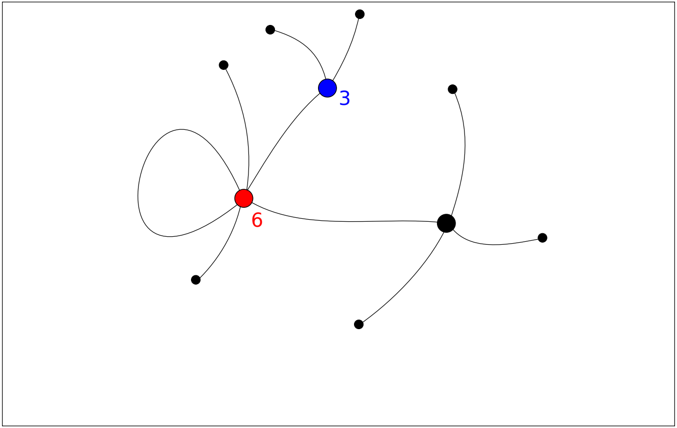
Modular operads.

Informal **Definition:** A modular operad is a sequence of objects $(M_2, M_3, M_4, \dots)$ and an algebraic operation for every graph:



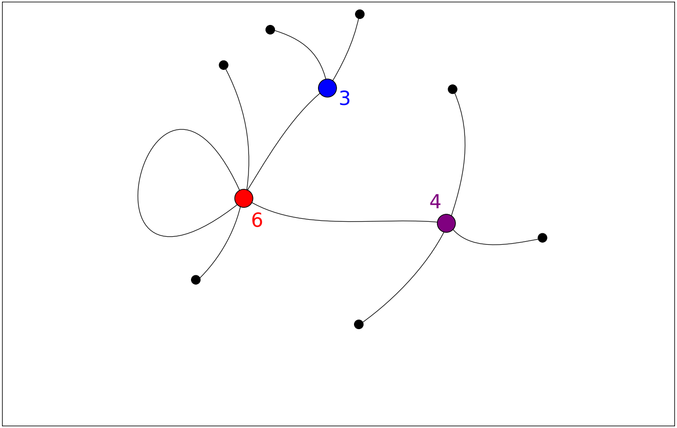
Modular operads.

Informal **Definition:** A modular operad is a sequence of objects $(M_2, M_3, M_4, \ldots)$ and an algebraic operation for every graph:



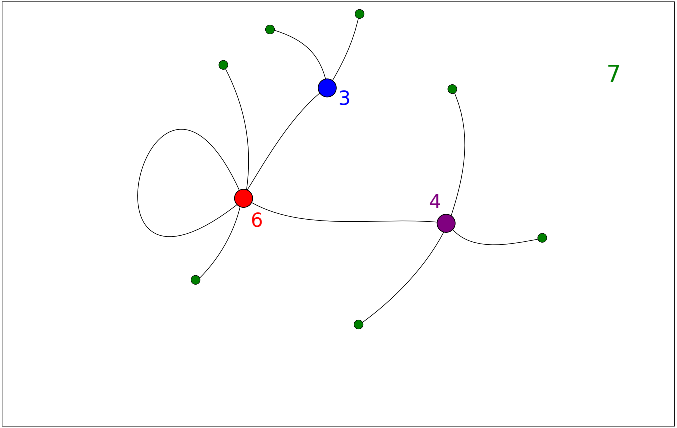
Modular operads.

Informal **Definition:** A modular operad is a sequence of objects $(M_2, M_3, M_4, \dots)$ and an algebraic operation for every graph:



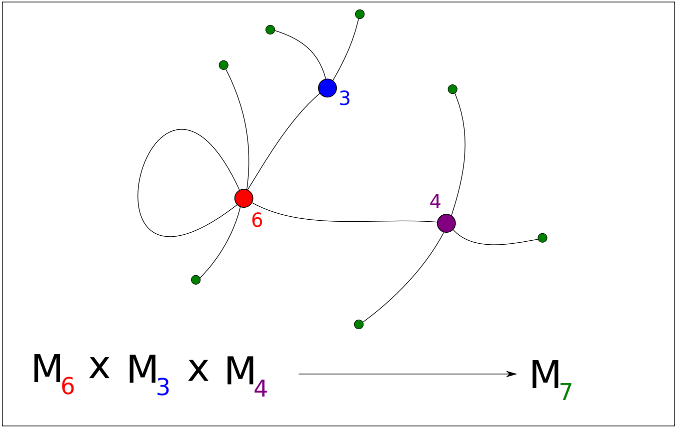
Modular operads.

Informal **Definition:** A modular operad is a sequence of objects $(M_2, M_3, M_4, \dots)$ and an algebraic operation for every graph:



Modular operads.

Informal **Definition:** A modular operad is a sequence of objects $(M_2, M_3, M_4, \ldots)$ and an algebraic operation for every graph:

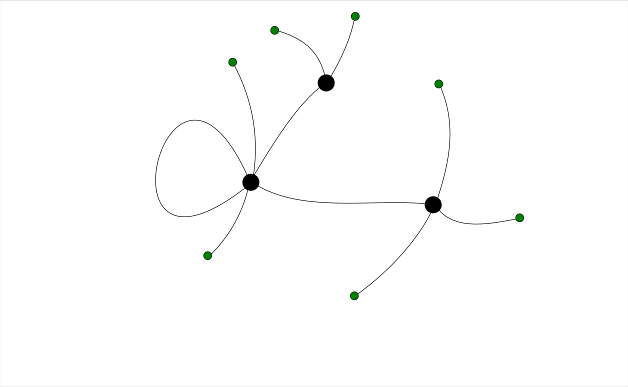


Examples of modular operads

Let A be an associative algebra and define $(M_2, M_3, M_4, ...) = (A, 0, 0, ...)$

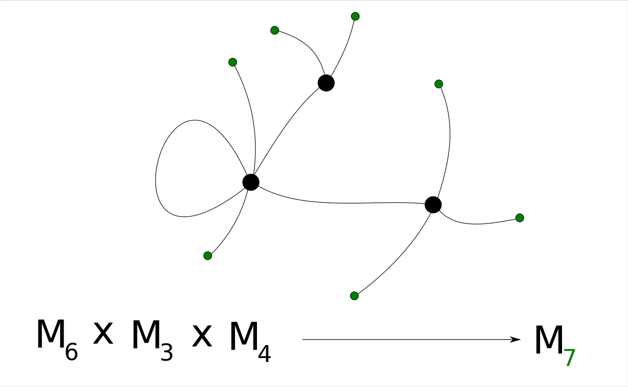
Examples of modular operads

Let A be an associative algebra and define $(M_2, M_3, M_4, ...) = (A, 0, 0, ...)$



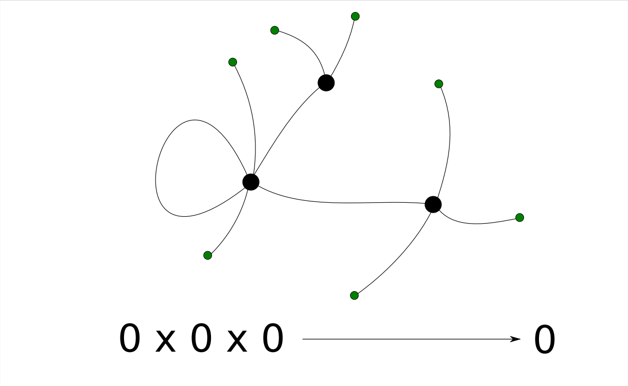
Examples of modular operads

Let A be an associative algebra and define $(M_2, M_3, M_4, \ldots) = (A, 0, 0, \ldots)$



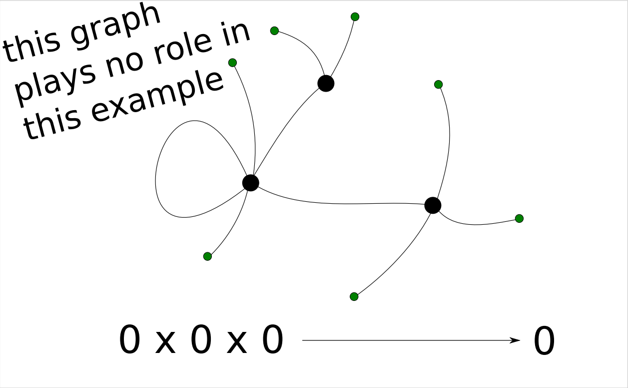
Examples of modular operads

Let A be an associative algebra and define $(M_2, M_3, M_4, ...) = (A, 0, 0, ...)$



Examples of modular operads

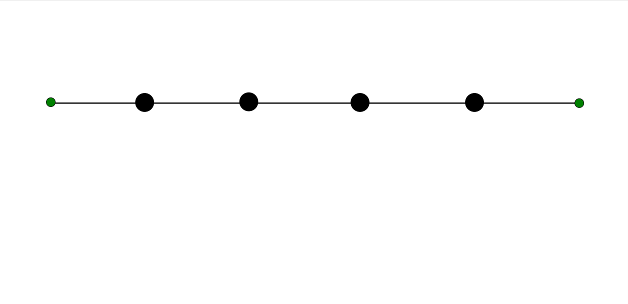
Let A be an associative algebra and define $(M_2, M_3, M_4, ...) = (A, 0, 0, ...)$



Examples of modular operads

Let A be an associative algebra and define $(M_2, M_3, M_4, \ldots) = (A, 0, 0, \ldots)$

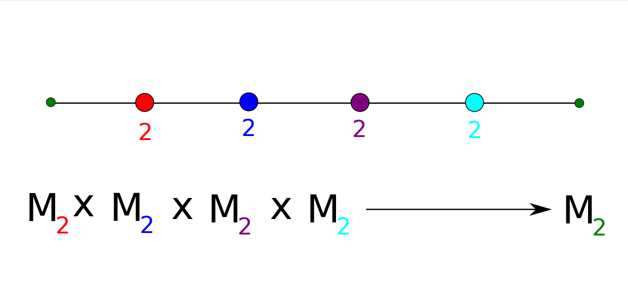
- The operations are trivial unless all internal vertices have valence 2.



Examples of modular operads

Let A be an associative algebra and define $(M_2, M_3, M_4, \ldots) = (A, 0, 0, \ldots)$

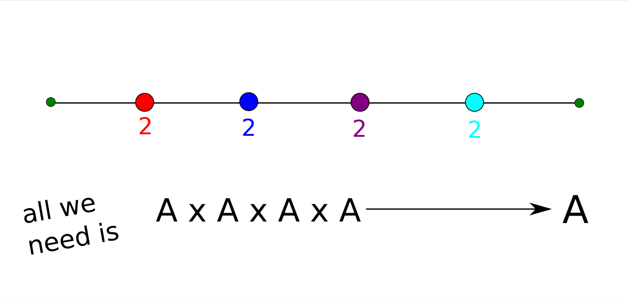
- The operations are trivial unless all internal vertices have valence 2.



Examples of modular operads

Let A be an associative algebra and define $(M_2, M_3, M_4, \ldots) = (A, 0, 0, \ldots)$

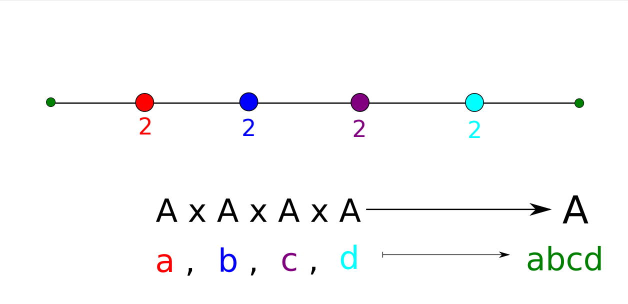
- The operations are trivial unless all internal vertices have valence 2.



Examples of modular operads

Let A be an associative algebra and define $(M_2, M_3, M_4, ...) = (A, 0, 0, ...)$

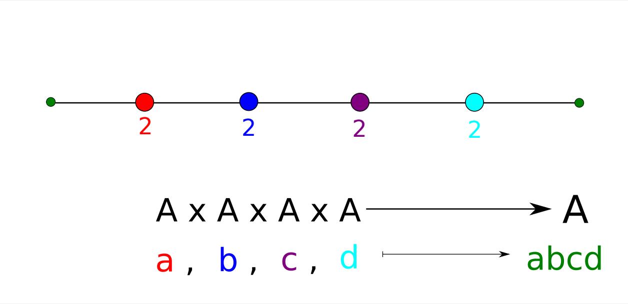
- The operations are trivial unless all internal vertices have valence 2.



Examples of modular operads

Let A be an associative algebra and define $(M_2, M_3, M_4, ...) = (A, 0, 0, ...)$

- The operations are trivial unless all internal vertices have valence 2.



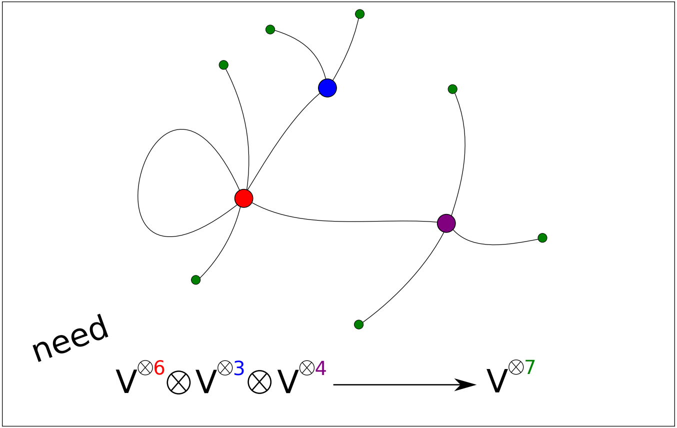
- Modular operads generalize associativity.

Examples of modular operads

Let V be a vector space and $V \otimes V \xrightarrow{\langle -,- \rangle} \mathbb{Q}$ an inner product.
Define $(M_2, M_3, M_4, \ldots) = (V^{\otimes 2}, V^{\otimes 3}, V^{\otimes 4}, \ldots)$.

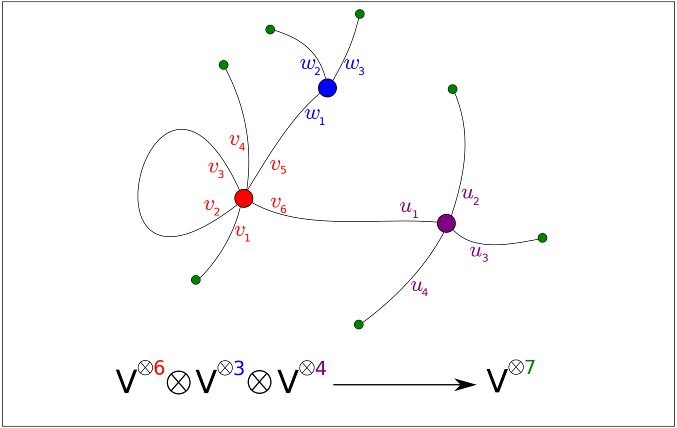
Examples of modular operads

Let V be a vector space and $V \otimes V \xrightarrow{\langle -,- \rangle} \mathbb{Q}$ an inner product.
Define $(M_2, M_3, M_4, \ldots) = (V^{\otimes 2}, V^{\otimes 3}, V^{\otimes 4}, \ldots)$.



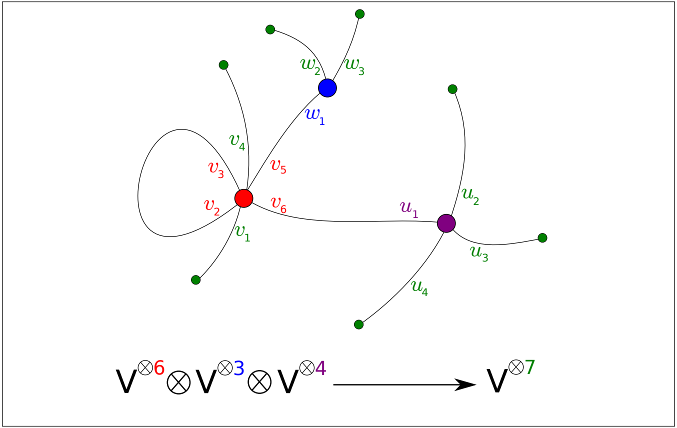
Examples of modular operads

Let V be a vector space and $V \otimes V \xrightarrow{\langle -,- \rangle} \mathbb{Q}$ an inner product.
Define $(M_2, M_3, M_4, \ldots) = (V^{\otimes 2}, V^{\otimes 3}, V^{\otimes 4}, \ldots)$.



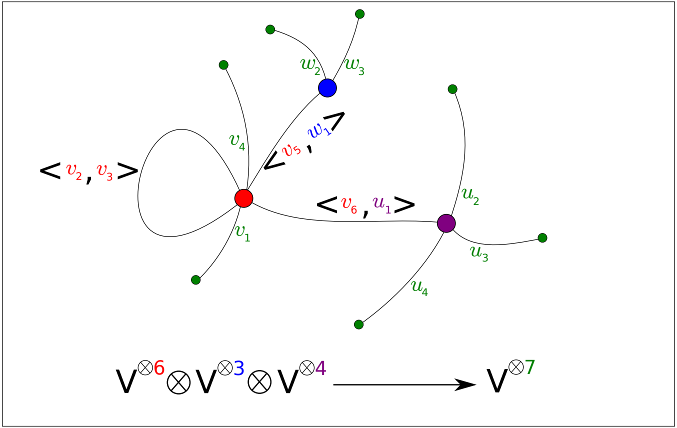
Examples of modular operads

Let V be a vector space and $V \otimes V \xrightarrow{\langle -,- \rangle} \mathbb{Q}$ an inner product.
Define $(M_2, M_3, M_4, \ldots) = (V^{\otimes 2}, V^{\otimes 3}, V^{\otimes 4}, \ldots)$.



Examples of modular operads

Let V be a vector space and $V \otimes V \xrightarrow{\langle -,- \rangle} \mathbb{Q}$ an inner product.
Define $(M_2, M_3, M_4, \ldots) = (V^{\otimes 2}, V^{\otimes 3}, V^{\otimes 4}, \ldots)$.

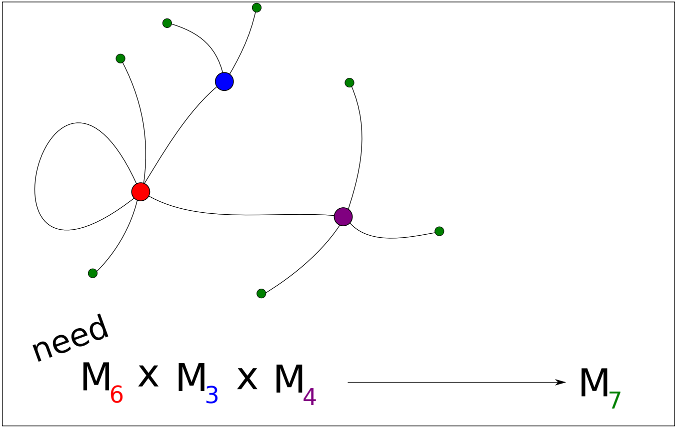


Examples of modular operads

Surfaces: Let M_n be the set of compact, orientable surfaces with n boundary components.

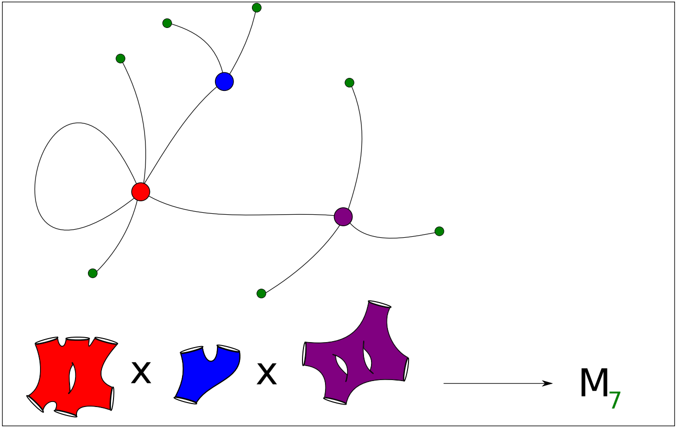
Examples of modular operads

Surfaces: Let M_n be the set of compact, orientable surfaces with n boundary components.



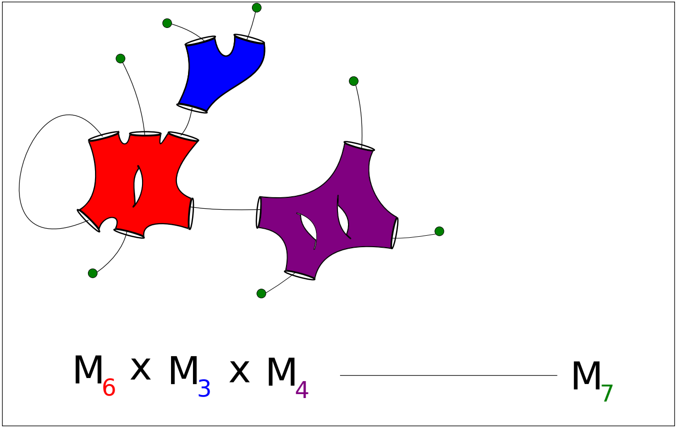
Examples of modular operads

Surfaces: Let M_n be the set of compact, orientable surfaces with n boundary components.



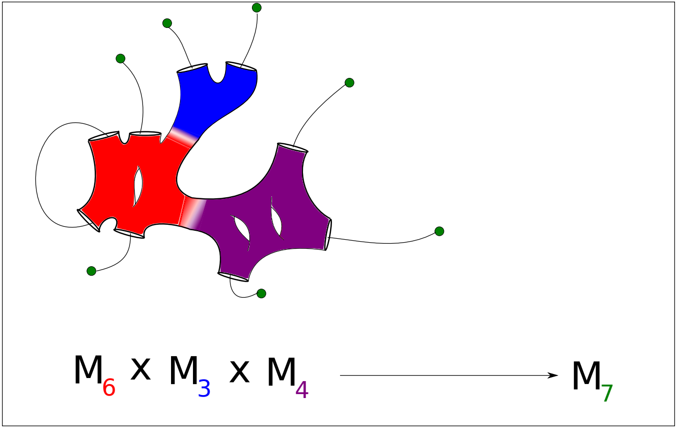
Examples of modular operads

Surfaces: Let M_n be the set of compact, orientable surfaces with n boundary components.



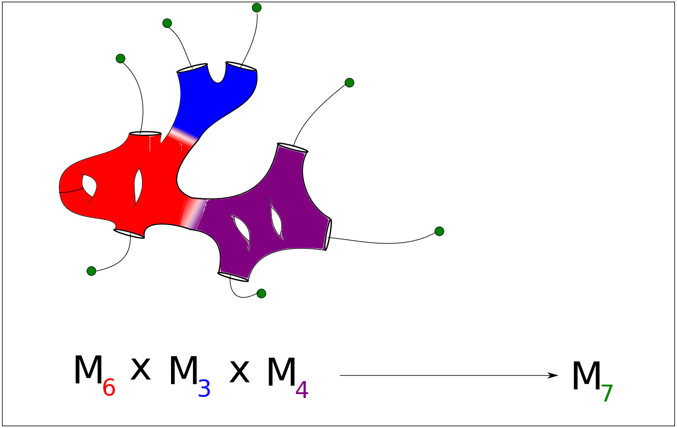
Examples of modular operads

Surfaces: Let M_n be the set of compact, orientable surfaces with n boundary components.



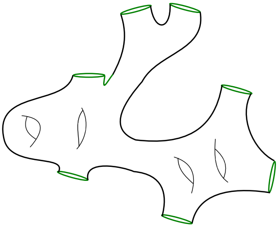
Examples of modular operads

Surfaces: Let M_n be the set of compact, orientable surfaces with n boundary components.



Examples of modular operads

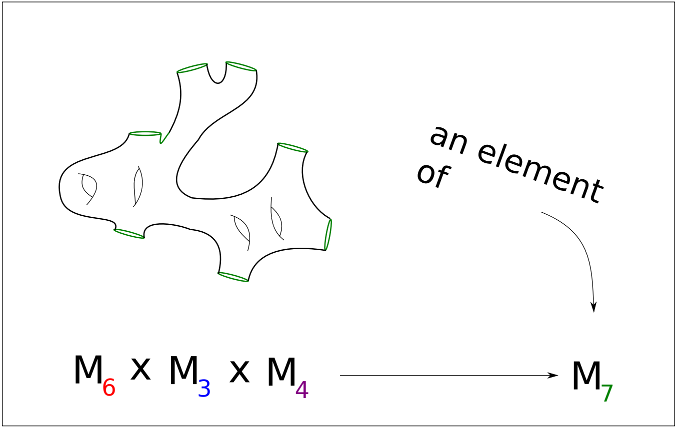
Surfaces: Let M_n be the set of compact, orientable surfaces with n boundary components.



$$M_6 \times M_3 \times M_4 \longrightarrow M_7$$

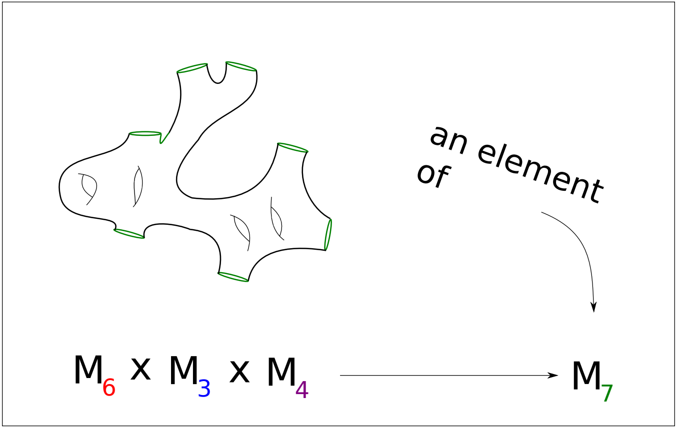
Examples of modular operads

Surfaces: Let M_n be the set of compact, orientable surfaces with n boundary components.



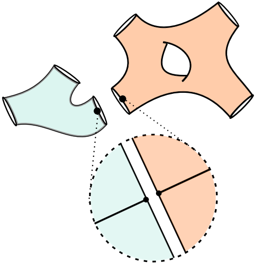
Examples of modular operads

Surfaces: Let M_n be the set of compact, orientable surfaces with n boundary components.

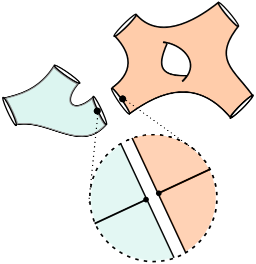


- Surfaces form a modular operad by gluing.

Other examples of modular operads:

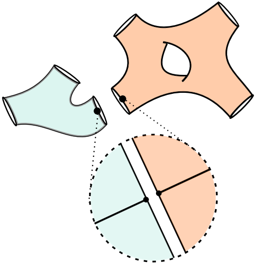


Other examples of modular operads:



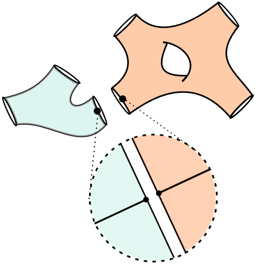
- Moduli spaces of surfaces with boundary.

Other examples of modular operads:



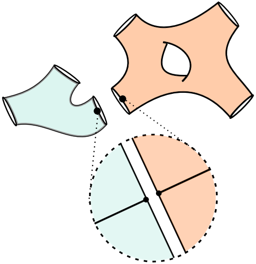
- Moduli spaces of surfaces with boundary.
- Deligne-Mumford compactifications of surfaces with punctures.

Other examples of modular operads:



- Moduli spaces of surfaces with boundary.
- Deligne-Mumford compactifications of surfaces with punctures.
- Graph homology (discussed below).

Other examples of modular operads:



- Moduli spaces of surfaces with boundary.
- Deligne-Mumford compactifications of surfaces with punctures.
- Graph homology (discussed below).

It's preferable to separate out the genus: $\mathcal{M} = \{\mathcal{M}_{g,n}\}$.

Back to the analogy

	then	now
Algebraic structure	Associativity	Modular Operad
Combinatorics	Multiply along a line	Multiply along a graph
Polytopes	Associahedra	Bracketohedra
Homotopy Transfer	via A_∞-algebras	
use to study	Topological spaces	

Present Goal: Fill in this table.

Back to the analogy

	then	now
Algebraic structure	Associativity	Modular Operad
Combinatorics	Multiply along a line	Multiply along a graph
Polytopes	Associahedra	Bracketohedra
Homotopy Transfer	via A_∞-algebras	theorem
use to study	Topological spaces	

Present Goal: Fill in this table.

Homotopy Transfer Theorem for Modular Operads.

Generalizing the classical A_∞ story we have:

Homotopy Transfer Theorem for Modular Operads.

Generalizing the classical A_∞ story we have:

Theorem (W.)

Every dg modular operad is quasi-isomorphic to a "A_∞-modular operad" structure on its homology.

Homotopy Transfer Theorem for Modular Operads.

Generalizing the classical A_∞ story we have:

Theorem (W.)

Every dg modular operad is quasi-isomorphic to a "A_∞-modular operad" structure on its homology.

Proof.

1. Encode modular operads as algebras over a *quadratic operad*.

Homotopy Transfer Theorem for Modular Operads.

Generalizing the classical A_∞ story we have:

Theorem (W.)

Every dg modular operad is quasi-isomorphic to a "A_∞-modular operad" structure on its homology.

Proof.

1. Encode modular operads as algebras over a *quadratic operad*. This requires using colored operads whose colors form not just a set but a groupoid.

Homotopy Transfer Theorem for Modular Operads.

Generalizing the classical A_∞ story we have:

Theorem (W.)

Every dg modular operad is quasi-isomorphic to a "A_∞-modular operad" structure on its homology.

Proof.

1. Encode modular operads as algebras over a *quadratic operad*. This requires using colored operads whose colors form not just a set but a groupoid.
2. Prove that the operad encoding modular operads is Koszul.

Homotopy Transfer Theorem for Modular Operads.

Generalizing the classical A_∞ story we have:

Theorem (W.)

Every dg modular operad is quasi-isomorphic to a "A_∞-modular operad" structure on its homology.

Proof.

1. Encode modular operads as algebras over a *quadratic operad*. This requires using colored operads whose colors form not just a set but a groupoid.
2. Prove that the operad encoding modular operads is Koszul. This is where we use contractibility of bracketohedra.

Homotopy Transfer Theorem for Modular Operads.

Generalizing the classical A_∞ story we have:

Theorem (W.)

Every dg modular operad is quasi-isomorphic to a "A_∞-modular operad" structure on its homology.

Proof.

1. Encode modular operads as algebras over a *quadratic operad*. This requires using colored operads whose colors form not just a set but a groupoid.
2. Prove that the operad encoding modular operads is Koszul. This is where we use contractibility of bracketeohedra.
3. Generalize classical Koszul duality theory from operads to groupoid colored operads.

□

Remarks about A_∞-modular operads.

Remarks about A_∞-modular operads.

1. A modular operad has an operation corresponding to every graph but

Remarks about A_∞-modular operads.

1. A modular operad has an operation corresponding to every graph but these operations are generated by one edge compositions.

Remarks about A_∞-modular operads.

1. A modular operad has an operation corresponding to every graph but these operations are generated by one edge compositions. An A_∞ modular operad has a *generating operation* corresponding to every graph.

Remarks about A_∞-modular operads.

1. A modular operad has an operation corresponding to every graph but these operations are generated by one edge compositions. An A_∞ modular operad has a *generating operation* corresponding to every graph.

2. In an A_∞ modular operad, the degree of a generating operation is $|Edges| - 1$.

Remarks about A_∞-modular operads.

1. A modular operad has an operation corresponding to every graph but these operations are generated by one edge compositions. An A_∞ modular operad has a *generating operation* corresponding to every graph.

2. In an A_∞ modular operad, the degree of a generating operation is $|Edges| - 1$.

3. Crucially, we don't resolve the S_n actions

Remarks about A_∞-modular operads.

1. A modular operad has an operation corresponding to every graph but these operations are generated by one edge compositions. An A_∞ modular operad has a *generating operation* corresponding to every graph.

2. In an A_∞ modular operad, the degree of a generating operation is $|Edges| - 1$.

3. Crucially, we don't resolve the S_n actions and the homotopy transfer holds S_n equivariantly.

Back to the analogy... one last time.

How do we use this generalization?

	then	now
Algebraic structure	Associativity	Modular Operad
Combinatorics	Multiply along a line	Multiply along a graph
Polytopes	Associahedra	Bracketohedra
Homotopy Transfer	via A_∞-algebras	via A_∞- modular operads
use to study	Topological spaces	

Back to the analogy... one last time.

How do we use this generalization?

	then	now
Algebraic structure	Associativity	Modular Operad
Combinatorics	Multiply along a line	Multiply along a graph
Polytopes	Associahedra	Bracketohedra
Homotopy Transfer	via A_∞-algebras	via A_∞- modular operads
use to study	Topological spaces	graph homology

Commutative and Lie structures

Commutative and Lie structures

Define $Lie(n)$ = span of Lie words on n letters.

Commutative and Lie structures

Define $Lie(n)$ = span of Lie words on n letters.

- $Lie(2)$ is dimension 1,

Commutative and Lie structures

Define $Lie(n)$ = span of Lie words on n letters.

- $Lie(2)$ is dimension 1, basis $[a, b]$.
- $Lie(3)$ is dimension 2,

Commutative and Lie structures

Define $Lie(n)$ = span of Lie words on n letters.

- $Lie(2)$ is dimension 1, basis $[a, b]$.
- $Lie(3)$ is dimension 2,

Commutative and Lie structures

Define $Lie(n)$ = span of Lie words on n letters.

- $Lie(2)$ is dimension 1, basis $[a, b]$.
- $Lie(3)$ is dimension 2, basis $[[a, b], c], [[c, a], b]$.

Commutative and Lie structures

Define $Lie(n)$ = span of Lie words on n letters.

- $Lie(2)$ is dimension 1, basis $[a, b]$.
- $Lie(3)$ is dimension 2, basis $[[a, b], c], [[c, a], b]$.
- $Lie(n)$ is dimension $(n-1)!$.

Commutative and Lie structures

Define $Lie(n)$ = span of Lie words on n letters.

- $Lie(2)$ is dimension 1, basis $[a, b]$.
- $Lie(3)$ is dimension 2, basis $[[a, b], c], [[c, a], b]$.
- $Lie(n)$ is dimension $(n-1)!$.

Define $Com(n)$ = span of commutative and associative words on n letters.

Commutative and Lie structures

Define $Lie(n)$ = span of Lie words on n letters.

- $Lie(2)$ is dimension 1, basis $[a,b]$.
- $Lie(3)$ is dimension 2, basis $[[a,b],c],[[c,a],b]$.
- $Lie(n)$ is dimension $(n-1)!$.

Define $Com(n)$ = span of commutative and associative words on n letters.

- $Com(2)$ is dimension 1,

Commutative and Lie structures

Define $Lie(n)$ = span of Lie words on n letters.

- $Lie(2)$ is dimension 1, basis $[a, b]$.
- $Lie(3)$ is dimension 2, basis $[[a, b], c], [[c, a], b]$.
- $Lie(n)$ is dimension $(n-1)!$.

Define $Com(n)$ = span of commutative and associative words on n letters.

- $Com(2)$ is dimension 1, basis ab.

Commutative and Lie structures

Define $Lie(n)$ = span of Lie words on n letters.

- $Lie(2)$ is dimension 1, basis $[a, b]$.
- $Lie(3)$ is dimension 2, basis $[[a, b], c], [[c, a], b]$.
- $Lie(n)$ is dimension $(n-1)!$.

Define $Com(n)$ = span of commutative and associative words on n letters.

- $Com(2)$ is dimension 1, basis ab.
- $Com(n)$ is dimension 1.

Lie vs Commutative words

Both Com and Lie are operads.

Lie vs Commutative words

Both Com and Lie are operads.

For any operad $\mathcal{O}$ we can consider the power series

$$f_{\mathcal{O}}(x) = \sum_{n=1}^{\infty} \frac{dim(\mathcal{O}(n))}{n!} x^n$$

Lie vs Commutative words

Both Com and Lie are operads.

For any operad $\mathcal{O}$ we can consider the power series

$$f_{\mathcal{O}}(x) = \sum_{n=1}^{\infty} \frac{dim(\mathcal{O}(n))}{n!} x^n$$

In particular

$$f_{Lie} = log(1-x) \quad \text{and} \quad f_{Com} = e^x - 1$$

are formal inverses of each other (up to sign).

Lie vs Commutative words

Both Com and Lie are operads.

For any operad $\mathcal{O}$ we can consider the power series

$$f_{\mathcal{O}}(x) = \sum_{n=1}^{\infty} \frac{dim(\mathcal{O}(n))}{n!} x^n$$

In particular

$$f_{Lie} = log(1-x) \quad \text{and} \quad f_{Com} = e^x - 1$$

are formal inverses of each other (up to sign).

This is a manifestation of Koszul duality.

Lie vs Commutative words

Both Com and Lie are operads.

For any operad $\mathcal{O}$ we can consider the power series

$$f_{\mathcal{O}}(x) = \sum_{n=1}^{\infty} \frac{dim(\mathcal{O}(n))}{n!} x^n$$

In particular

$$f_{Lie} = log(1-x) \quad \text{and} \quad f_{Com} = e^x - 1$$

are formal inverses of each other (up to sign).

This is a manifestation of Koszul duality. If I didn't know the dimension of $Lie(n)$ I could compute it from the dimension of Com.

Modular operads associated to Com and Lie

Here are two ways to associate a modular operad to an operad like Com and Lie.

Modular operads associated to Com and Lie

Here are two ways to associate a modular operad to an operad like Com and Lie.

- Declare higher genus graphs to act by 0.

Modular operads associated to Com and Lie

Here are two ways to associate a modular operad to an operad like Com and Lie.

- Declare higher genus graphs to act by 0.
- Form the space of all $\mathcal{O}$-labeled graphs,

Modular operads associated to Com and Lie

Here are two ways to associate a modular operad to an operad like Com and Lie.

- Declare higher genus graphs to act by 0.
- Form the space of all $\mathcal{O}$-labeled graphs, called the Feynman transform $\mathrm{ft}(\mathcal{O})$.

Modular operads associated to Com and Lie

Here are two ways to associate a modular operad to an operad like Com and Lie.

- Declare higher genus graphs to act by 0.
- Form the space of all $\mathcal{O}$-labeled graphs, called the Feynman transform $\mathrm{ft}(\mathcal{O})$. It comes with a differential induced by the operadic structure maps.

Modular operads associated to Com and Lie

Here are two ways to associate a modular operad to an operad like Com and Lie.

- Declare higher genus graphs to act by 0.
- Form the space of all $\mathcal{O}$-labeled graphs, called the Feynman transform $\mathrm{ft}(\mathcal{O})$. It comes with a differential induced by the operadic structure maps.

A few remarks:

1. I'm omitting many important technical details.

Modular operads associated to Com and Lie

Here are two ways to associate a modular operad to an operad like Com and Lie.

- Declare higher genus graphs to act by 0.
- Form the space of all $\mathcal{O}$-labeled graphs, called the Feynman transform $\text{ft}(\mathcal{O})$. It comes with a differential induced by the operadic structure maps.

A few remarks:

1. I'm omitting many important technical details.
2. (Getzler-Kapranov) ft preserves quasi-isomorphisms and $\text{ft}^2 \sim id$.

Modular operads associated to Com and Lie

Here are two ways to associate a modular operad to an operad like Com and Lie.

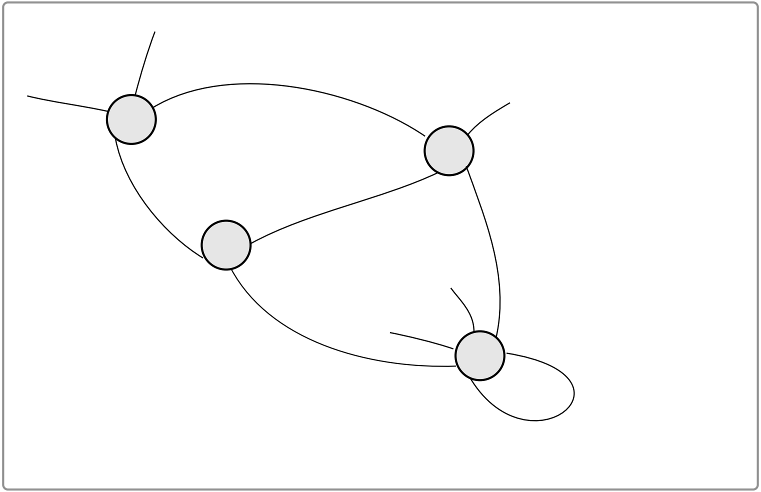
- Declare higher genus graphs to act by 0.
- Form the space of all $\mathcal{O}$-labeled graphs, called the Feynman transform $\mathrm{ft}(\mathcal{O})$. It comes with a differential induced by the operadic structure maps.

A few remarks:

1. I'm omitting many important technical details.
2. (Getzler-Kapranov) ft preserves quasi-isomorphisms and $\mathrm{ft}^2 \sim id$.
3. ft can be generalized to A_∞-modular operads.

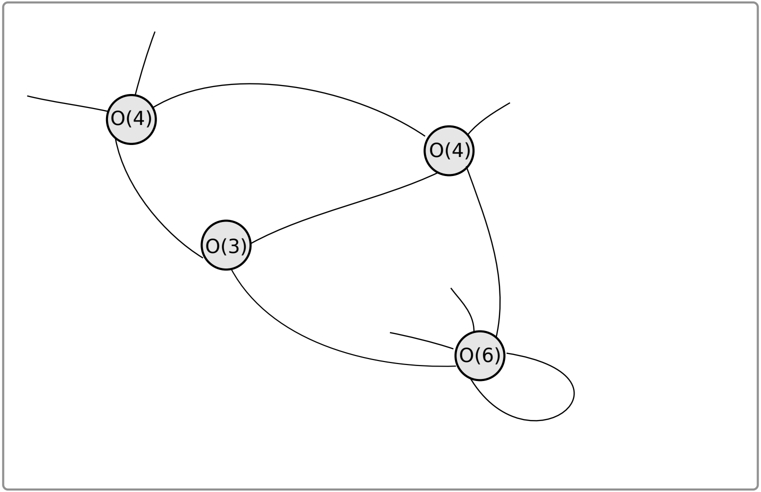
A Picture of the Feynman transform

Input is an operad (or modular operad) $\mathcal{O}$.



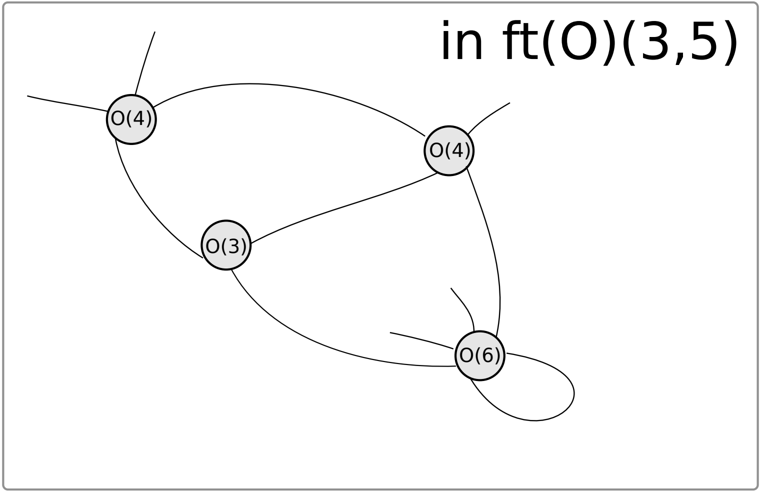
A Picture of the Feynman transform

Input is an operad (or modular operad) $\mathcal{O}$.



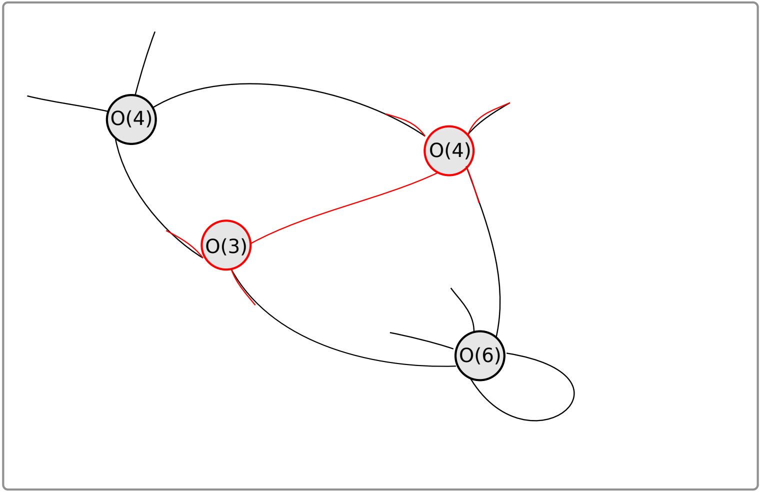
A Picture of the Feynman transform

Input is an operad (or modular operad) $\mathcal{O}$.



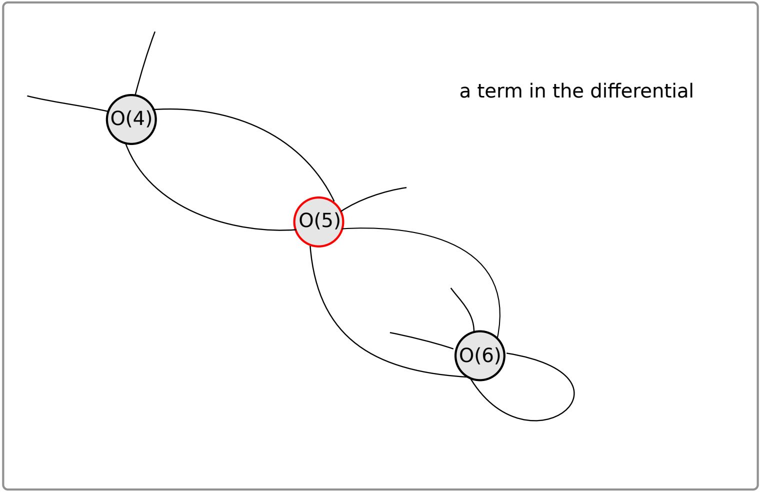
A Picture of the Feynman transform

Input is an operad (or modular operad) $\mathcal{O}$.



A Picture of the Feynman transform

Input is an operad (or modular operad) $\mathcal{O}$.



Why graph homology?

Define “graph homology”:

- $\mathcal{H}_{\text{Com}} := H_*(\text{ft}(\text{Com}))$
- $\mathcal{H}_{\text{Lie}} := H_*(\text{ft}(\text{Lie}))$

Why graph homology?

Define "graph homology":

- $\mathcal{H}_{\mathsf{Com}} := H_*(\mathrm{ft}(\mathsf{Com}))$
- $\mathcal{H}_{\mathsf{Lie}} := H_*(\mathrm{ft}(\mathsf{Lie}))$

Theorem (Willwacher)

$\oplus_g \mathcal{H}_{\mathsf{Com}}(g, 0)_{2g} \cong \mathfrak{grt}_1$

Why graph homology?

Define "graph homology":

- $\mathcal{H}_{\mathsf{Com}} := H_*(\mathsf{ft}(\mathsf{Com}))$
- $\mathcal{H}_{\mathsf{Lie}} := H_*(\mathsf{ft}(\mathsf{Lie}))$

Theorem (Willwacher)

$\oplus_g \mathcal{H}_{\mathsf{Com}}(g,0)_{2g} \cong \mathfrak{grt}_1$ *(modulo technical details)*

Why graph homology?

Define "graph homology":

- $\mathcal{H}_{\mathsf{Com}} := H_*(\mathrm{ft}(\mathrm{Com}))$
- $\mathcal{H}_{\mathsf{Lie}} := H_*(\mathrm{ft}(\mathrm{Lie}))$

Theorem (Willwacher)

$\oplus_g \mathcal{H}_{\mathsf{Com}}(g, 0)_{2g} \cong \mathfrak{grt}_1$ *(modulo technical details)*

Theorem (Kontsevich; Conant-Kassabov-Vogtmann)

$\mathcal{H}_{\mathsf{Lie}}(g, 0) \cong H_*(Out(F_g))$

Why graph homology?

Define "graph homology":

- $\mathcal{H}_{\mathsf{Com}} := H_*(\mathrm{ft}(\mathsf{Com}))$
- $\mathcal{H}_{\mathsf{Lie}} := H_*(\mathrm{ft}(\mathsf{Lie}))$

Theorem (Willwacher)

$\oplus_g \mathcal{H}_{\mathsf{Com}}(g,0)_{2g} \cong \mathfrak{grt}_1$ *(modulo technical details)*

Theorem (Kontsevich; Conant-Kassabov-Vogtmann)

$\mathcal{H}_{\mathsf{Lie}}(g,0) \cong H_*(Out(F_g))$

Question: Can we use Koszul duality between Com and Lie to relate these calculations?

Now add Massey products

Question: Can we use Koszul duality between Com and Lie to relate these calculations?

Now add Massey products

Question: Can we use Koszul duality between Com and Lie to relate these calculations?

Theorem (W.)

For each $g > 0$ *the complex* $\mathrm{ft}(\mathcal{H}_{Lie})(g, n)$

- *is acyclic, and*

Now add Massey products

Question: Can we use Koszul duality between Com and Lie to relate these calculations?

Theorem (W.)

For each $g > 0$ *the complex* $\mathrm{ft}(\mathcal{H}_{Lie})(g, n)$

- *is acyclic, and*
- *contains* $\mathrm{ft}(\mathrm{Com})(g, n)$ *as a subcomplex.*

Now add Massey products

Question: Can we use Koszul duality between Com and Lie to relate these calculations?

Theorem (W.)

For each $g > 0$ the complex $\mathrm{ft}(\mathcal{H}_{Lie})(g, n)$

- *is acyclic, and*
- *contains* $\mathrm{ft}(\mathrm{Com})(g, n)$ *as a subcomplex.*

Thus every cycle in $\mathrm{ft}(\mathrm{Com})(g, n)$ is a boundary in $\mathrm{ft}(\mathcal{H}_{Lie})(g, n)$,

Now add Massey products

Question: Can we use Koszul duality between Com and Lie to relate these calculations?

Theorem (W.)

For each $g > 0$ *the complex* $\mathrm{ft}(\mathcal{H}_{Lie})(g, n)$

- *is acyclic, and*
- *contains* $\mathrm{ft}(\mathrm{Com})(g, n)$ *as a subcomplex.*

Thus every cycle in $\mathrm{ft}(\mathrm{Com})(g, n)$ is a boundary in $\mathrm{ft}(\mathcal{H}_{Lie})(g, n)$, i.e. every commutative graph homology class is represented, via Massey products, by a graph labeled by Lie graph homology classes.

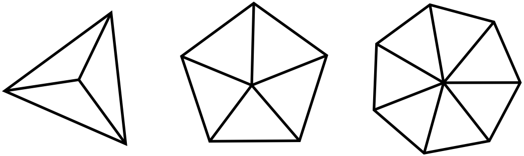
An example

Every commutative graph homology class is represented, via Massey products, by a graph labeled by Lie graph homology classes.

An example

Every commutative graph homology class is represented, via Massey products, by a graph labeled by Lie graph homology classes.

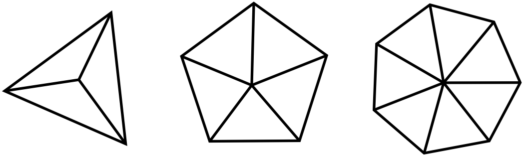
- Willwacher used the correspondence with $\mathfrak{grt}_1$ to construct a family of commutative graph homology classes σ_{2j+1} with $(g, n) = (2j + 1, 0)$.



An example

Every commutative graph homology class is represented, via Massey products, by a graph labeled by Lie graph homology classes.

- Willwacher used the correspondence with $\mathfrak{grt}_1$ to construct a family of commutative graph homology classes σ_{2j+1} with $(g, n) = (2j + 1, 0)$.

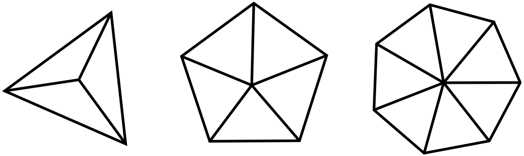


- Conant-Hatcher-Kassabov-Vogtmann used a group theoretic interpretation of Lie graph homology to construct a sequence of classes α_{2j+1} with $(1, 2j + 1)$.

An example

Every commutative graph homology class is represented, via Massey products, by a graph labeled by Lie graph homology classes.

- Willwacher used the correspondence with $\mathfrak{grt}_1$ to construct a family of commutative graph homology classes σ_{2j+1} with $(g, n) = (2j + 1, 0)$.



- Conant-Hatcher-Kassabov-Vogtmann used a group theoretic interpretation of Lie graph homology to construct a sequence of classes α_{2j+1} with $(1, 2j + 1)$.
  To relate these two, I need to know what the genus 1 Massey products are.

Theorem (W.)

Massey products for Lie graph homology are given by contracting odd polygons with commutative labels to hit α_{2j+1}.

Theorem (W.)

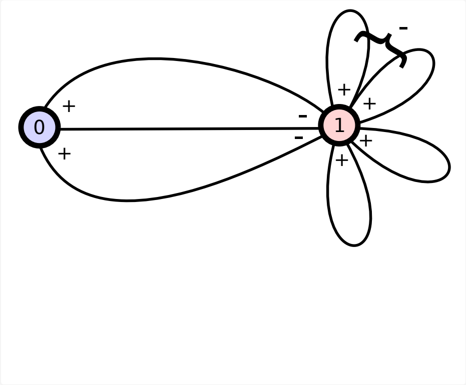
Massey products for Lie graph homology are given by contracting odd polygons with commutative labels to hit α_{2j+1}.

Define $\theta_{2j+1} \in \mathrm{ft}(\mathcal{H}_{\mathsf{Lie}})$ to be:

Theorem (W.)

Massey products for Lie graph homology are given by contracting odd polygons with commutative labels to hit α_{2j+1}.

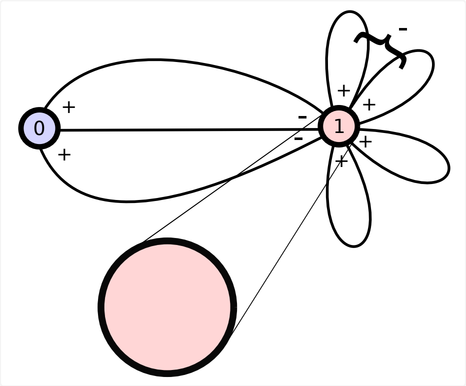
Define $\theta_{2j+1} \in \mathrm{ft}(\mathcal{H}_{\mathsf{Lie}})$ to be:



Theorem (W.)

Massey products for Lie graph homology are given by contracting odd polygons with commutative labels to hit α_{2j+1}.

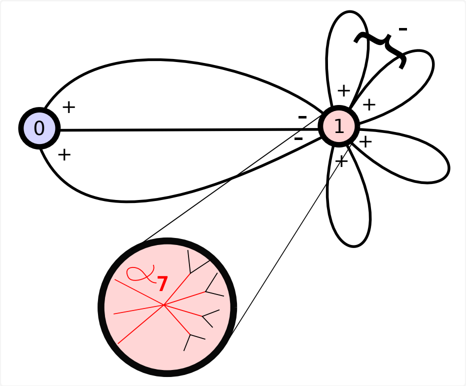
Define $\theta_{2j+1} \in \mathrm{ft}(\mathcal{H}_{\mathsf{Lie}})$ to be:



Theorem (W.)

Massey products for Lie graph homology are given by contracting odd polygons with commutative labels to hit α_{2j+1}.

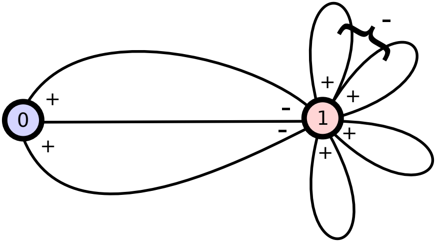
Define $\theta_{2j+1} \in \mathrm{ft}(\mathcal{H}_{\mathrm{Lie}})$ to be:



Theorem (W.)

Massey products for Lie graph homology are given by contracting odd polygons with commutative labels to hit α_{2j+1}.

Define $\theta_{2j+1} \in \mathrm{ft}(\mathcal{H}_{\mathrm{Lie}})$ to be:



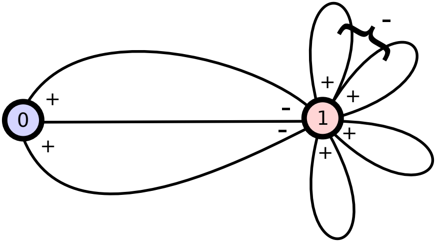
Theorem (W.)

In the derived Feynman transform $\mathrm{ft}(\mathcal{H}_{Lie})$, $d(\theta_{2j+1}) = \sigma_{2j+1}$.

Theorem (W.)

Massey products for Lie graph homology are given by contracting odd polygons with commutative labels to hit α_{2j+1}.

Define $\theta_{2j+1} \in \mathrm{ft}(\mathcal{H}_{\mathrm{Lie}})$ to be:



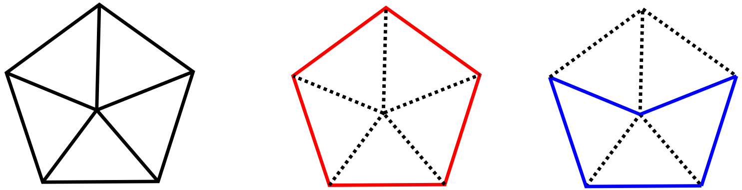
Theorem (W.)

In the derived Feynman transform $\mathrm{ft}(\mathcal{H}_{Lie})$, $d(\theta_{2j+1}) = \sigma_{2j+1}$. In particular σ_{2j+1} is a cycle in ft(Com).

Theorem (W.)

Massey products for Lie graph homology are given by contracting odd polygons with commutative labels to hit α_{2j+1}.

Define $\theta_{2j+1} \in \mathrm{ft}(\mathcal{H}_{\mathrm{Lie}})$ to be:



Theorem (W.)

In the derived Feynman transform $\mathrm{ft}(\mathcal{H}_{Lie})$, $d(\theta_{2j+1}) = \sigma_{2j+1}$. In particular σ_{2j+1} is a cycle in ft(Com).

Lie graph homology model for $\mathfrak{grt}_1$?

Question: Can we describe the filtered Lie algebra $\mathfrak{grt}_1$ via Lie graph homology?

Lie graph homology model for $\mathfrak{grt}_1$?

Question: Can we describe the filtered Lie algebra $\mathfrak{grt}_1$ via Lie graph homology?

Theorem (W. – in progress)

$\overline{\mathsf{ft}}(\mathcal{H}_{Lie})(2,n) \otimes_{S_n} \Lambda_n \sim \mathcal{F}_2(\mathfrak{grt}_1)/\mathcal{F}_3(\mathfrak{grt}_1)$

Lie graph homology model for $\mathfrak{grt}_1$?

Question: Can we describe the filtered Lie algebra $\mathfrak{grt}_1$ via Lie graph homology?

Theorem (W. – in progress)

$\overline{\mathsf{ft}}(\mathcal{H}_{Lie})(2, n) \otimes_{S_n} \Lambda_n \sim \mathcal{F}_2(\mathfrak{grt}_1)/\mathcal{F}_3(\mathfrak{grt}_1)$

"To understand filtration degree 2 requires understanding Lie graph homology upto genus 2".

Lie graph homology model for $\mathfrak{grt}_1$?

Question: Can we describe the filtered Lie algebra $\mathfrak{grt}_1$ via Lie graph homology?

Theorem (W. – in progress)

$\overline{\mathrm{ft}}(\mathcal{H}_{Lie})(2,n) \otimes_{S_n} \Lambda_n \sim \mathcal{F}_2(\mathfrak{grt}_1)/\mathcal{F}_3(\mathfrak{grt}_1)$

"To understand filtration degree 2 requires understanding Lie graph homology upto genus 2".

This isomorphism sees the fact that $\mathcal{H}_{Lie}(2,n) \otimes_{S_n} \Lambda_n$ coincides with

Lie graph homology model for $\mathfrak{grt}_1$?

Question: Can we describe the filtered Lie algebra $\mathfrak{grt}_1$ via Lie graph homology?

Theorem (W. – in progress)

$\overline{\mathrm{ft}}(\mathcal{H}_{Lie})(2,n) \otimes_{S_n} \Lambda_n \sim \mathcal{F}_2(\mathfrak{grt}_1)/\mathcal{F}_3(\mathfrak{grt}_1)$

"To understand filtration degree 2 requires understanding Lie graph homology upto genus 2".

This isomorphism sees the fact that $\mathcal{H}_{Lie}(2,n) \otimes_{S_n} \Lambda_n$ coincides with the space of relations among the $\{\sigma_{2i+1}, \sigma_{2n-2i+1}\} \in \mathcal{F}_2(\mathfrak{grt}_1)/\mathcal{F}_3(\mathfrak{grt}_1)$.

Lie graph homology model for $\mathfrak{grt}_1$?

Question: Can we describe the filtered Lie algebra $\mathfrak{grt}_1$ via Lie graph homology?

Theorem (W. – in progress)

$\overline{\mathrm{ft}}(\mathcal{H}_{Lie})(2,n)\otimes_{S_n}\Lambda_n \sim \mathcal{F}_2(\mathfrak{grt}_1)/\mathcal{F}_3(\mathfrak{grt}_1)$

"To understand filtration degree 2 requires understanding Lie graph homology upto genus 2".

This isomorphism sees the fact that $\mathcal{H}_{Lie}(2,n)\otimes_{S_n}\Lambda_n$ coincides with the space of relations among the $\{\sigma_{2i+1},\sigma_{2n-2i+1}\}\in\mathcal{F}_2(\mathfrak{grt}_1)/\mathcal{F}_3(\mathfrak{grt}_1)$. Both spaces, after work of Conant-Kassabov-Hatcher-Vogtmann on the one hand and Schneps on the other, have dimension = space of cusp forms of weight n+2.

Bibliography

- Massey Products for graph homology. arXiv:1903.12055; to appear in Int. Math. Res. Not.
- Toward a minimal model for $H_*(\overline{\mathcal{M}})$. arXiv:2011.01171
- Wheel graph homology classes via Lie graph homology. arXiv:2102.09522

Bibliography

- Massey Products for graph homology. arXiv:1903.12055; to appear in Int. Math. Res. Not.
- Toward a minimal model for $H_*(\overline{\mathcal{M}})$. arXiv:2011.01171
- Wheel graph homology classes via Lie graph homology. arXiv:2102.09522

Thank you!