OPERADS PACKAGE

1. Introduction

This package contains a Coq formalization of the definition of a symmetric colored operad, along with a demonstration of an example (the operad of sets) that satisfies this definition. These files were developed as part of work on the DARPA V-SPELLS (Verified Security and Performance Enhancement of Large Legacy Software) program. More detailed information about the background of this program and about this work can be found in our accompanying paper (`https://arxiv.org/abs/2303.08894`).

The package contains the following Coq files:

- `Typecast.v` contains some type-casting definitions, lemmas, and tactics used by the other files.
- `ColoredOperads.v` contains the formal definition of a **Type**-colored operad.
- `OperadOfSets.v` contains the operad of sets example which satisfies that definition.
- `TColoredOperads.v` contains the formal definition of a T-colored operad for an example of a Tarski universe T.
- `TOperadOfSets.v` contains the operad of sets example which satisfies that definition.

The package contains a standard makefile allowing `make` and `make install` commands to build and install.

2. Mathematical Definition of an Operad

Our definition in Coq of an operad is based on the following mathematical definition for an operad $\mathcal{O}$. This definition is an adaptation of the definition of operads in [1].

Definition 2.1. Let S be a set. We call $\mathcal{O}$ an *S-colored operad* (where we have just been using "operad" for short), if it consists of the following data:

(1) For any $d \in S$ and sequence $c_1, \ldots, c_n$ of length $n \geq 1$ of elements in S, $\mathcal{O}$ is equipped with a set

$$\mathcal{O}\binom{d}{\underline{c}} = \mathcal{O}\binom{d}{c_1, \ldots c_n}$$

(2) For any $d \in S$, sequence $c_1, \ldots, c_n$ of length $n \geq 1$ of elements in S, and bijection σ of the set $\{1, \ldots, n\}$, we have a bijection of sets:

$$\mathcal{O}\binom{d}{c_1, \ldots, c_n} \longrightarrow \mathcal{O}\binom{d}{c_{\sigma(1)}, \ldots, c_{\sigma(n)}}$$

For example, this means there is an isomorphism between the set,

$$\mathcal{O}\begin{pmatrix} d \\ c_2, c_3, c_1 \end{pmatrix}$$

and the set,

$$\mathcal{O}\begin{pmatrix} d \\ c_1, c_2, c_3 \end{pmatrix}$$

That is, switching up the order of the c_i has no impact on what the set $\mathcal{O}\binom{d}{c_1,\ldots,c_n}$ looks like.

(3) For each $c \in S$, $\mathcal{O}$ is equipped with a specific element $\mathbb{1}_c \in \mathcal{O}\binom{c}{c}$ called the *c-colored unit* of $\mathcal{O}$. We can think of these as as something similar to an identity function on a set.

(4) For each $d \in S$, sequence of length $\underline{c} = c_1, \ldots, c_n$ with $n \geq 1$ of elements in S, sequence of length m, $\underline{b} = b_1, \ldots, b_m$ of elements in S, and $1 \leq i \leq n$, we write

$$\underline{c} \bullet_i \underline{b}$$

for the sequence of elements of length $n + m - 1$ in S given by

$$\underbrace{c_1, \ldots, c_{i-1}}_{\emptyset \text{ if } i=1}, \underline{b}, \underbrace{c_{i+1}, \ldots, c_n}_{\emptyset \text{ if } i=n}$$

We require that $\mathcal{O}$ is equipped with a function, denoted by $\circ_i$ such that:

$$\mathcal{O}\begin{pmatrix} d \\ \underline{c} \end{pmatrix} \times \mathcal{O}\begin{pmatrix} c_i \\ \underline{b} \end{pmatrix} \xrightarrow{\circ_i} \mathcal{O}\begin{pmatrix} d \\ \underline{c} \bullet_i \underline{b} \end{pmatrix}$$

For example, if $\underline{c} = c_1, c_2, c_3$, $\underline{b} = b_1, b_2$, and $i = 2$, then $\circ_2$ is a map from

$$\mathcal{O}\begin{pmatrix} d \\ c_1, c_2, c_3 \end{pmatrix} \times \mathcal{O}\begin{pmatrix} c_2 \\ b_1, b_2 \end{pmatrix} \longrightarrow \mathcal{O}\begin{pmatrix} d \\ c_1, b_1, b_2, c_3 \end{pmatrix}$$

We also require that the data given in $(1) - (4)$ above be subject to the following axioms:

Suppose $d \in S$, $\underline{c} = c_1, \ldots, c_n, \underline{a} = a_1, \ldots, a_\ell$, and $\underline{b} = b_1, \ldots, b_m$, where all sequences come from the set S.

Associativity Axioms (1) Suppose $n \geq 2$ and $1 \leq i < j \leq n$, and $(\alpha, \beta, \gamma) \in \mathcal{O}\binom{d}{\underline{c}} \times \mathcal{O}\binom{c_i}{\underline{a}} \times \mathcal{O}\binom{c_j}{\underline{b}}$, we require that

$$(\star) \qquad (\alpha \circ_i \beta) \circ_{j-1+\ell} \gamma = (\alpha \circ_j \gamma) \circ_i \beta$$

Note the left-hand side of $(\star)$ is an element of

$$\mathcal{O}\begin{pmatrix} d \\ (\underline{c} \bullet_i \underline{a}) \bullet_{j-1+\ell} \underline{b} \end{pmatrix}$$

and the right-hand side of $(\star)$ an element of

$$\mathcal{O}\begin{pmatrix} d \\ (\underline{c} \bullet_j \underline{b}) \bullet_i \underline{a} \end{pmatrix}$$

To ensure sanity, and that everything type-checks, since $i < j$, note that

$$\begin{aligned}(\underline{c} \bullet_i \underline{a}) \bullet_{j-1+\ell} \underline{b} &= (c_1, \ldots, c_{i-1}, \underline{a}, c_{i+1}, \ldots, c_n) \bullet_{j-1+\ell} \underline{b} \\ &= c_1, \ldots, c_{i-1}, \underline{a}, c_{i+1}, \ldots c_{j-1}, \underline{b}, c_{j+1}, \ldots, c_n \\ &= (\underline{c} \bullet_j \underline{b}) \bullet_i \underline{a}\end{aligned}$$

We call this the *horizontal associativity axiom.*

(2) Suppose $m, n \geq 1$, $1 \leq i \leq n$, and $1 \leq j \leq m$. Then for $(\alpha, \beta, \gamma) \in \mathcal{O}\binom{d}{\underline{c}} \times \mathcal{O}\binom{c_i}{\underline{b}} \times \mathcal{O}\binom{b_j}{\underline{a}}$, we require that

$$(\star\star) \qquad (\alpha \circ_i \beta) \circ_{i-1+j} \gamma = \alpha \circ_i (\beta \circ_j \gamma)$$

We note that left-hand side of $(\star\star)$ is an element of

$$\mathcal{O}\binom{d}{(\underline{c} \bullet_i \underline{b}) \bullet_{i-1+j} \underline{a}}$$

And the right-hand side of $(\star\star)$ is an element of

$$\mathcal{O}\binom{d}{c \bullet_i (\underline{b} \bullet_j \underline{a})}$$

We again provide a sanity check (which will also provide a type-check!):

$$\begin{aligned}\underline{c} \bullet_i (\underline{b} \bullet_j \underline{a}) &= c_1, \ldots, c_{i-1}, (\underline{b} \bullet_j \underline{a}), c_{i+1}, \ldots, c_n \\ &= c_1, \ldots, c_{i-1}, b_1, \ldots, b_{j-1}, \underline{a}, b_{j+1}, \ldots, b_m, c_{i+1}, \ldots, c_n \\ &= (\underline{c} \bullet_i \underline{b}) \bullet_{i-1+j} \underline{a}\end{aligned}$$

We call this the *vertical associativity axiom.*

Unity Axioms (1) For $\alpha \in \mathcal{O}\binom{d}{c_1, \ldots, c_n}$,

$$\alpha = \mathbb{1}_d \circ_1 \alpha.$$

We call this the *left-unity axiom.*

(2) For $n \geq 1$ and $1 \leq i \leq n$, and $\alpha \in \mathcal{O}\binom{d}{c_1, \ldots, c_n}$,

$$\alpha \circ_i \mathbb{1}_{c_i} = \alpha.$$

We call this the *right-unity axiom.*

Remark 2.2. We omit the *equivariance axiom* that is part of the definition of operads in [1] because it is not relevant for our purposes.

3. Two Versions of the Operad Definition

As noted in the introduction, `ColoredOperads.v` contains one version of the operad definition where S (of Definition 2.1) is the Coq universe **Type**.

In the `TColoredOperads.v` version of the definition, S is a subset of **Type** represented as a *Tarski universe*, i.e., a type T whose terms represent the types of interest. The following definitions formalize our example of a Tarski universe:

- **baseTypes** defines the type sigils representing the three base types of our example, **unit**, **nat**, and **bool**.
- **interpT** is the interpretation function matching a base type sigil to the type it represents.
- **T** is the Tarski universe constructed from the base type sigils.
- **El** is the interpretation function matching a term of type **T** to the type it represents.

Other than using **T** instead of **Type** and several calls to the **El** function, `TColoredOperads.v` and `TOperadOfSets.v` are nearly identical to their non-Tarski universe counterparts.

4. Formal Definition of an Operad

We define an operad as a record named **Operad**, with one field containing the first piece of data in Definition 2.1. The remainder of Definition 2.1 is contained a second record named **operadLaws**.

The second piece of data in Definition 2.1 requires a concept of isomorphism in **Type**. Definition **iso** states this concept, relying on definitions of function composition and inverse. We use this, together with Coq's build-in inductive **Permutation** type, to state the bijection condition in the **perm** field of the **operadLaws** record.

The third piece of data in Definition 2.1 is specified in the **opId** field.

For the fourth piece of data in Definition 2.1, we need to define an operation that replaces an element of a list with a new list. Definition **insert** accomplishes this.

Definition **lookup** allows a statement identifying a particular entry of a list. We use this condition in the **operadComp** field to formalize the fourth piece of data in Definition 2.1.

In order to formalize the remaining axioms, we require type-casting functions. For example, in the above informal statement of the left unity axiom, the left-hand side α is of type $\mathcal{O}\binom{d}{\underline{c}}$, while the right-hand side $\mathbb{1}_d \circ_1 \alpha$ is of type $\mathcal{O}\binom{d}{d \bullet_1 \underline{c}}$. Coq will not recognize these as the same type, and thus will not allow a direct statement of equality formalizing the left unity axiom; however, given a proof of equality of the two types, we may utilize the `rewrite` tactic to define a function casting a term of one type into the other, and then state the axiom using that function.

For the left unity axiom, the proof of equality of types is given by lemma **unityLInsert**, and the casting function by definition **unityLCast**. Field **unityL** uses this casting function to state the axiom.

Analogously, lemmas **unityRInsert**, **vertInsert**, and **horizInsert** prove equalities of types relevant to the right unity, vertical associativity, and horizontal associativity axioms, respectively. The longest portion of the operad definition files is devoted to proving these lemmas. The corresponding casting functions are **unityRCast**, **vertCast**, and **horizCast**, and the axiom statements appear in fields **unityR**, **vertAssoc**, and **horizAssoc**.

5. Type-casting Helper Lemmas

Because the operad laws contain type-casting functions, demonstrating that the operad of sets satisfies these laws requires reasoning about these functions. Thus, `Typecast.v` contains definitions, lemmas, and tactics related to type-casting which are used as helpers in the proof that the operad of sets satisfies the operad laws.

Definition **typecast** generalizes the type-casting functions we mentioned above: given a proof that types A and B are equal and term of type A, **typecast** returns the corresponding term of type B. As a slight variation, given a proof that two terms a and b of the same type A are equal, a function f from A to **Type**, and a term of type $f(a)$, **dep_typecast** returns the corresponding term of type $f(b)$. (We prove that every **dep_typecast** is equal to an appropriately constructed **typecast**, and that the casts mentioned in the previous section are equal to appropriately constructed **dep_typecast**s and thus to appropriately constructed **typecast**s.)

We note that sometimes two differently formulated types can be recognized by the Coq type-checker as the same; in this case **typecast** is unnecessary. However, interaction of different type formulations with **typecast**s may hamper proof efforts. We therefore define the **retypecast** tactic, which rewrites **typecast**s formulated by the type equation $A_1 = B_1$ to **typecast**s formulated by the type equation $A_2 = B_2$, where A_1 and A_2 are two different formulations recognized by Coq as the same type, as are B_1 and B_2. We also define the **redepcast** tactic which does the same for **dep_typecast**s.

In addition to these definitions and tactics, `Typecast.v` contains several helper lemmas for manipulating these type-casts. We do not detail them all here; each is explained briefly in comments in the code.

6. The Operad of Sets

Let S represent either **Type** or a Tarski universe T. Our example of an operad is the operad of sets, which we will denote this by $\mathbf{Sets}_S$. Given types $c_1, \ldots, c_n, d \in S$, we let $\mathbf{Sets}_S\binom{d}{c_1,\ldots,c_n}$ be the set of functions from $c_1 \times \cdots \times c_n$ to d. Definition **setsOp** defines this operad.

Then in order for $\mathbf{Sets}_S$ to be an operad, we need to ensure it has all of the data in Definition 2.1, as well as satisfying the associativity and unity axioms. This is accomplished in the following lemmas:

- Lemma **setsPerm** proves the following. For $n \geq 1$, $c_1, \ldots, c_n, d \in S$, and σ a permutation of the set $\{1, \ldots, n\}$, we need to ensure there is a bijection between the set of functions $\mathbf{Sets}_S\binom{d}{c_1,\ldots,c_n}$ and $\mathbf{Sets}_S\binom{d}{c_{\sigma(1)},\ldots,c_{\sigma(n)}}$. That is, there is a map a F that takes a function $f : c_1 \times \cdots \times c_n \longrightarrow d$, and returns a function $F(f) : c_{\sigma(1)} \times \cdots \times c_{\sigma(n)} \longrightarrow d$. Moreover, the function F is invertible.
- Definition **setsId** defines the c-colored unit in $\mathbf{Sets}_S\binom{c}{c}$. This is the identity function $1_c : c \longrightarrow c$. That is, for $x \in c$, $1_c(x) = x$.
- Definition **setsComp** defines the functions

$$\circ_i : \mathbf{Sets}_S\begin{pmatrix} d \\ \underline{c} \end{pmatrix} \times \mathbf{Sets}_S\begin{pmatrix} c_i \\ \underline{b} \end{pmatrix} \longrightarrow \mathbf{Sets}_S\begin{pmatrix} d \\ \underline{c} \bullet_i \underline{b} \end{pmatrix}$$

where $\underline{c} = c_1, \ldots, c_n$ with $n \geq 1$, $\underline{b} = b_1, \ldots, b_m$, and we recall that

$$\underline{c} \bullet_i \underline{b} = c_1, \ldots, c_{i-1}, b_1, \ldots, b_m, c_{i+1}, \ldots, c_n.$$

Given $f \in \mathbf{Sets}_S\binom{d}{\underline{c}}, g \in \mathbf{Sets}_S\binom{c_i}{\underline{b}}$, we let $f \circ_i g \in \mathbf{Sets}_S\binom{d}{\underline{c}\bullet_i\underline{b}}$ be the function for which

$$(f \circ_i g)(x_1, \ldots, x_{n+m-1}) = f(x_1, \ldots, x_{i-1}, g(x_i, \ldots, x_{i+m-1}), x_{i+1}, \ldots, x_n).$$

- Lemmas **sets_unityL**, **sets_unityR**, **sets_vertAssoc**, and **sets_horizAssoc** prove that $\mathbf{Sets}_S$ satisfies the unity and associativity axioms of Definition 2.1. These proofs are straightforward to write informally; the formal proofs are somewhat complicated by type-casting.

Finally, definition **setsLaws** invokes the above lemmas to show that our operad satisfies all the operad laws.

References

[1] Donald Yau. *Operads of wiring diagrams*, volume 2192 of *Lecture Notes in Mathematics*. Springer, Cham, 2018.