Category-Graded Algebraic Theories and Effect Handlers

Takahiro Sanada [1]

[1]RIMS, Kyoto University

6 September 2022
ERATO MMSD Colloquium

Overview

I propose a novel effects system based on a category-graded extension of algebraic theories that correspond to category-graded monads.

monad	algebraic theory	effect system with handlers
category-graded monad	category-graded algebraic theory	category-graded effect system with handlers

Outline

Lax functors and extensions of monads

Category-Graded Algebraic Theories

Category-Graded Effect System with Effect Handlers

Lax functors and extensions of monads

Category-Graded Algebraic Theories

Category-Graded Effect System with Effect Handlers

Monads

Monads are used to capture computation in terms of category theory.
A monad is a functor $T\colon \mathbb{C} \to \mathbb{C}$ with

$$
\begin{tikzcd}
\mathbb{C} \arrow[d, bend right=50, "\mathrm{Id}_{\mathbb{C}}"'{name=L}] \arrow[d, bend left=50, "T"{name=R}] \arrow[from=L, to=R, Rightarrow, "\eta"'] \\
\mathbb{C}
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathbb{C} \arrow[d, "T"'] \arrow[dd, bend left=60, "T"{name=R}] \\
\mathbb{C} \arrow[d, "T"'] \arrow[r, phantom, "\xRightarrow{\mu}"] & {} \\
\mathbb{C}
\end{tikzcd}
$$

that satisfies appropriate axioms.

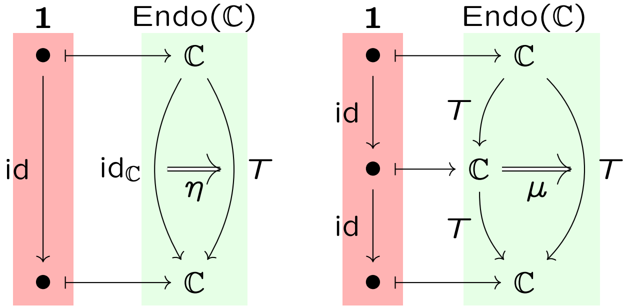
Monads can be seen as a special case of lax functor.
$F : \mathbb{S} \to \mathcal{K}$ where $\mathcal{K}$ is a 2-category.

$$
\begin{tikzcd}
\mathbb{S} & \mathcal{K} \\
a \arrow[r, mapsto] \arrow[d, "\mathrm{id}_a"'] & Fa \arrow[d, bend right=40, "\mathrm{id}_{Fa}"'] \arrow[d, bend left=40, "F(\mathrm{id}_a)"] \arrow[d, phantom, "\Rightarrow_{\eta_a}"] \\
a \arrow[r, mapsto] & Fa
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathbb{S} & \mathcal{K} \\
a \arrow[r, mapsto] \arrow[d, "f"'] & Fa \arrow[d, "Ff"'] \arrow[dd, bend left=60, "F(f;g)"] \\
b \arrow[r, mapsto] \arrow[d, "g"'] & Fb \arrow[d, "Fg"'] \arrow[r, phantom, "\Rightarrow_{\mu_{f,g}}"] & {} \\
c \arrow[r, mapsto] & Fc
\end{tikzcd}
$$

that satisfies appropriate axioms.
If η_a and $\mu_{f,g}$ are identities, the lax functor is an ordinary functor.

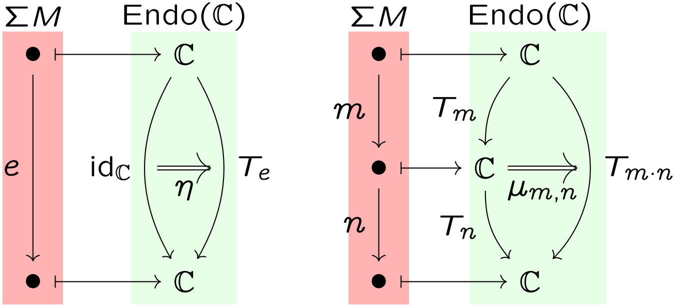
Monads as lax functors

A monad is a lax functor $T\colon \mathbf{1} \to \mathrm{Endo}(\mathbb{C})$ [Benabou, 1967].



Graded monads as lax functors

For a monoid $M = (M, e, \cdot)$, an M-graded monad (w/o order) is a lax functor $T \colon \Sigma M \to \mathrm{Endo}(\mathbb{C})$ [Katsumata, 2014].



Category-graded monads as lax functors

For a category $\mathbb{S}$, an $\mathbb{S}$-graded monad is a lax functor $T\colon \mathbb{S}^{\mathrm{op}} \to \mathrm{Endo}(\mathbb{C})$ [Orchard, Wadler and Eades III, 2020].

$$
\begin{tikzcd}
\mathbb{S}^{\mathrm{op}} & \mathrm{Endo}(\mathbb{C}) \\
a \arrow[r, mapsto] & \mathbb{C} \arrow[d, bend right=40, "\mathrm{id}_{\mathbb{C}}"'{name=L}] \arrow[d, bend left=40, "T_{\mathrm{id}_a}"{name=R}] \arrow[from=L, to=R, Rightarrow, "\eta"'] \\
a \arrow[u, "\mathrm{id}_a"] \arrow[r, mapsto] & \mathbb{C}
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathbb{S}^{\mathrm{op}} & \mathrm{Endo}(\mathbb{C}) & \\
a \arrow[r, mapsto] & \mathbb{C} \arrow[d, "T_f"'] \arrow[dd, bend left=50, "T_{f \circ g}"{name=R}] & \\
b \arrow[u, "f"] \arrow[r, mapsto] & \mathbb{C} \arrow[d, "T_g"'] \arrow[to=R, Rightarrow, "\mu_{f,g}"'] & \\
c \arrow[u, "g"] \arrow[r, mapsto] & \mathbb{C} &
\end{tikzcd}
$$

Lax functors and extensions of monads

Category-Graded Algebraic Theories

Category-Graded Effect System with Effect Handlers

Let $\Sigma = \{\mathbf{op}, \ldots\}$ be a set of operations

For each $\mathbf{op} \in \Sigma$, a set called coarity P and a set called arity Q are assigned $(\mathbf{op} \colon P \to Q)$.
The set $\mathit{Term}_\Sigma(X)$ of terms on a set X is defined inductively by:

$$\frac{x \in X}{x \in \mathit{Term}_\Sigma(X)}$$

$$\frac{\mathbf{op} \colon P \to Q \qquad p \in P \qquad \{t_i\}_{i \in Q} \subset \mathit{Term}_\Sigma(X)}{\mathbf{do}(\mathbf{op}, p, \{t_i\}_{i \in Q}) \in \mathit{Term}_\Sigma(X)}$$

$$\mathbf{do}(\mathbf{op}, p, \{t_i\}_{i \in Q})$$

$$= \begin{array}{c} \mathbf{op}, p \\ \diagup \quad \diagdown \\ \underbrace{t_1 \cdots \quad t_n \cdots}_{Q\text{-many}} \end{array}$$

Proposition

$\mathit{Term}_\Sigma(-) \colon \mathbf{Set} \to \mathbf{Set}$ forms a monad.

Examples of terms

Consider algebraic theories for a state. Let S be a set of values of the state.

$$\Sigma = \{\mathbf{put} : S \rightarrowtriangle 1,\ \mathbf{get} : 1 \rightarrowtriangle S\}$$

For example, a term $\mathbf{do}(\mathbf{get}, (), \{\mathbf{do}(\mathbf{put}, s, s)\}_s)$ gets a value from state, put it to state again, and return the value s.
Equations for this signature are:

$$\begin{aligned}
\mathbf{do}(\mathbf{put}, s, \mathbf{do}(\mathbf{put}, s', t)) &= \mathbf{do}(\mathbf{put}, s', t) \\
\mathbf{do}(\mathbf{put}, s, \mathbf{do}(\mathbf{get}, (), \{t_{s'}\}_{s' \in S})) &= \mathbf{do}(\mathbf{put}, s, t_s) \\
\mathbf{do}(\mathbf{get}, (), \{\mathbf{do}(\mathbf{put}, s, t_s)\}_{s \in S}) &= \mathbf{do}(\mathbf{get}, (), \{t_s\}_s) \\
\mathbf{do}(\mathbf{get}, (), \{\mathbf{do}(\mathbf{get}, (), \{t_{ss'}\}_{s'})\}_s) &= \mathbf{do}(\mathbf{get}, (), \{t_{ss}\}_s).
\end{aligned}$$

Category-graded algebraic theories

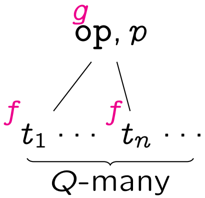
For each $\mathbf{op} \in \Sigma$, a coarity P, an arity Q, and a grade $f : b \to a$ in $\mathbb{S}$ are assigned $(\mathbf{op} \colon P \twoheadrightarrow Q; f)$.
The set $Term_\Sigma(f, X)$ of terms graded by f on a set X is defined inductively by:

$$\frac{a \in \mathbb{S} \quad x \in X}{x \in Term_\Sigma(\mathrm{id}_a, X)}$$

$$\frac{\mathbf{op} \colon P \twoheadrightarrow Q; g \quad p \in P \quad \{t_i\}_{i \in Q} \subset Term_\Sigma(f, X)}{\mathbf{do}(\mathbf{op}, p, \{t_i\}_{i \in Q}) \in Term_\Sigma(f \circ g, X)}$$

$\mathbf{do}(\mathbf{op}, p, \{t_i\}_{i \in Q})$

$=$



Proposition

$\{Term_\Sigma(f, -)\}_f$ forms an $\mathbb{S}$-graded monad.

Lax functors and extensions of monads

Category-Graded Algebraic Theories

Category-Graded Effect System with Effect Handlers

Example

Let $\mathbb{S}$ be the category generated by the following graph

$$\begin{tikzcd}
nat \arrow[loop above, "r_{nn}"] \arrow[loop below, "s_n"'] \arrow[r, bend left=40, "r_{nb}"] \arrow[r, bend left=15, "\tau_b"'] & bool \arrow[loop above, "r_{bb}"] \arrow[loop below, "s_b"'] \arrow[l, bend left=15, "\tau_n"'] \arrow[l, bend left=40, "r_{bn}"]
\end{tikzcd}$$

Consider a situation where we have a state of types that can change, and we can send or receive its stored value.

$$\Sigma = \left\{ \begin{array}{ll} \mathtt{send}_x : 1 \twoheadrightarrow 1; s_x, & \mathtt{recv}_{xy} : 1 \twoheadrightarrow 1; r_{xy}, \\ \mathtt{put}_{bn} : \mathtt{nat} \twoheadrightarrow 1; \tau_n, & \mathtt{put}_{bn} : \mathtt{bool} \twoheadrightarrow 1; \tau_b \\ \mathtt{put}_{nn} : \mathtt{nat} \twoheadrightarrow 1; \mathrm{id}_{nat}, & \mathtt{put}_{bb} : \mathtt{bool} \twoheadrightarrow 1; \mathrm{id}_{bool} \\ \mathtt{get}_n : 1 \twoheadrightarrow \mathtt{nat}; \mathrm{id}_{nat}, & \mathtt{get}_b : 1 \twoheadrightarrow \mathtt{bool}; \mathrm{id}_{bool} \end{array} \right\}$$

If we have $\vdash^{s_b \circ \tau_b \circ r_{nn}} M : A$, then M is a computation of type A that receives a data of type $\mathtt{nat}$ and then sends a data of type $\mathtt{bool}$ with some operations on the state.

Category-graded effect system

Fix a grading category $\mathbb{S}$ and a signature Σ.

Type	$A ::= 1 \mid A \to B; f \mid A \times B \mid A + B$
Value	$V, W ::= x \mid () \mid \lambda x : A.M \mid \ldots$
Computation	$M, N ::= \mathbf{val}_a\, V \mid \mathbf{let}\, x \leftarrow M\, \mathbf{in}\, N$ $\mid \mathbf{op}(V) \mid VW \mid \ldots$

The type $A \to B; f$ means that a value which has this type is a function from A to B and performs effects graded by f.

Typing rule

Judgements $\Gamma \vdash V : A$ for values
$\Gamma \vdash^{f} M : A$ for computations

The judgement $\Gamma \vdash^{f} M : A$ means that the computation M has a type A and invokes effects graded by f.

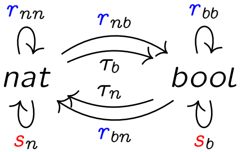
Value

$$\frac{x : A \in \Gamma}{\Gamma \vdash x : A} \qquad \frac{\Gamma, x : A \vdash^{f} M : A}{\Gamma \vdash \lambda x : A.M : A \to B; f} \qquad \cdots$$

Computation

$$\frac{\Gamma \vdash V : A}{\Gamma \vdash^{\mathrm{id}_a} \mathbf{val}_a\, V : A} \qquad \frac{\Gamma \vdash V : P \quad \mathbf{op} : P \twoheadrightarrow Q; f}{\Gamma \vdash^{f} \mathbf{op}(V) : Q}$$

$$\frac{\Gamma \vdash^{g:c \to b} M : A \quad \Gamma, x : A \vdash^{f:b \to a} N : B}{\Gamma \vdash^{f \circ g} \mathbf{let}\, x \leftarrow M\, \mathbf{in}\, N : B} \qquad \cdots$$



$\vdash^{s_b \circ \tau_b \circ r_{nn}}$

$\mathbf{let}\ \_ \leftarrow \mathtt{recv}_{nn}()\ \mathbf{in}$
$\mathbf{let}\ x \leftarrow \mathtt{get}_n()\ \mathbf{in}$
$\mathbf{let}\ \_ \leftarrow \mathtt{put}_{nb}(\mathtt{true})\ \mathbf{in}$
$\mathbf{let}\ \_ \leftarrow \mathtt{send}_{bb}()\ \mathbf{in}$
$\mathbf{val}\ x : \mathtt{nat}$

The above program receives data before sends data.

$\vdash^{r_{nb} \circ s_n}$

$\mathbf{let}\ \_ \leftarrow \mathtt{put}_{nn}(3)\ \mathbf{in}$
$\mathbf{let}\ \_ \leftarrow \mathtt{send}_n()\ \mathbf{in}$
$\mathbf{let}\ \_ \leftarrow \mathtt{recv}_{nb}()\ \mathbf{in}$
$\mathbf{let}\ x \leftarrow \mathtt{get}_b()\ \mathbf{in}$
$\mathbf{val}\ x : \mathtt{bool}$

The above program receives data after sends data.

Denotational semantics

We define denotation of $\Gamma \vdash^{f} M : A$ using $\mathbb{S}$-graded monad $\{Term_\Sigma(f, -)\}_f$.

Value $[\![\Gamma \vdash V : A]\!] : [\![\Gamma]\!] \to [\![A]\!]$

Computation $\left[\!\left[\Gamma \vdash^{f} M : A\right]\!\right] : [\![\Gamma]\!] \to Term_\Sigma(f, [\![A]\!])$

- $\left[\!\left[\Gamma \vdash^{\mathrm{id}_a} \mathbf{val}\, V : A\right]\!\right](s) := \eta_a([\![V]\!](s)) \in Term_\Sigma(\mathrm{id}_a, [\![A]\!])$
- $\left[\!\left[\Gamma \vdash^{f \circ g} \mathbf{let}\, x \leftarrow M\, \mathbf{in}\, N : B\right]\!\right](s) := \mu_{f,g}(Term(g, [\![N]\!](s, -))([\![M]\!](s))) \in Term(f \circ g, [\![B]\!])$

...

We can also define operational semantics, and show soundness and adequacy theorem.

Handlers for exception

An exception handler catches an exception and executes exception handling.

$$
\begin{array}{l}
\textbf{handle} \\
\quad \textbf{if}\ x = 0 \\
\quad \textbf{then}\ \texttt{raise}(\text{“devided by zero”}) \\
\quad \textbf{else}\ 1/x \\
\textbf{with}\{ \\
\quad \texttt{raise}(e) \mapsto \textbf{val}\ 0 \\
\}
\end{array}
$$

Handlers for effects

Effect handlers [Plotkin and Pretner, 2009] are generalization of exception handlers.

$$
\begin{array}{l}
\textbf{handle} \\
\quad \textbf{let}\, x \leftarrow \mathtt{op}_i(V)\, \textbf{in}\, L \\
\textbf{with}\{ \\
\quad \textbf{val}\ x \mapsto N \\
\quad \mathtt{op}_1(p_1), r_1 \mapsto M_1 \\
\quad \vdots \\
\quad \mathtt{op}_n(p_n), r_n \mapsto M_n \\
\}
\end{array}
$$

Effect handlers catch general effect operations and continuation, and execute operation handling.

Handlers are homomorphism from the free algebra

Categorically, effect handlers are homomorphism from the free algebra.

handle
  L
with{
  val $x \mapsto$ N
  $\mathrm{op}_1(p_1), r_1 \mapsto M_1$
  $\vdots$
  $\mathrm{op}_n(p_n), r_n \mapsto M_n$
}

$$\begin{tikzcd} X \arrow[r, "\eta"] \arrow[dr, "\phi"'] & TX \arrow[d, "\phi^\dagger"] \\ & Y \end{tikzcd}
\qquad
\begin{tikzcd} (\mu \colon TTX \to TX) \arrow[d, dashed, "\phi^\dagger"] \\ (\alpha \colon TY \to Y) \end{tikzcd}$$

- The term N determines the morphism $\phi \colon X \to Y$.
- The terms $M_1, \ldots, M_n$ determine the algebra α.

The handler is interpreted by $\phi^\dagger$.

Typing rule for handlers

Typing rule for handlers are as follows:

$$\frac{\Gamma, x : A \vdash_{\Sigma'} N : B \qquad \big(\Gamma, p : P, r : Q \to B \vdash_{\Sigma'} M_{\mathrm{op}} : B\big)_{(\mathrm{op}:P \twoheadrightarrow Q) \in \Sigma}}{\Gamma \vdash_{\Sigma \Rightarrow \Sigma'} \{\mathbf{val}\, x \mapsto N\} \cup \{\mathrm{op}(p), r \mapsto M_{\mathrm{op}}\}_{\mathrm{op} \in \Sigma} : A \rightsquigarrow B}$$

$$\frac{\Gamma \vdash_{\Sigma} L : A \qquad \Gamma \vdash H : A \rightsquigarrow B}{\Gamma \vdash_{\Sigma'} \mathbf{handle}\, L\, \mathbf{with}\, H : B}$$

Operational semantics:

$$\mathbf{handle}\, \mathbf{val}\, V\, \mathbf{with}\, H \to N[V/x]$$

$$\begin{aligned}&\mathbf{handle}\, \mathcal{E}[\mathrm{op}(V)]\, \mathbf{with}\, H \\ &\to M_{\mathrm{op}}[V/p, \lambda y.\, \mathbf{handle}(\mathcal{E}[\mathbf{val}\, y])\, \mathbf{with}\, H/r]\end{aligned}$$

Handlers for category-graded effects

Category-graded version of effect handlers can also be constructed. Let $\mathbb{S}$ and $\mathbb{S}'$ be grading categories, $\Sigma = \{\mathtt{op}\colon P \twoheadrightarrow Q; f, \ldots\}$ (f is in $\mathbb{S}$) and $\Sigma' = \{\mathtt{op}'\colon P' \twoheadrightarrow Q'; f', \ldots\}$ (f' is in $\mathbb{S}'$) signatures.

$$
\begin{array}{l}
\textbf{handle} \\
\quad L \\
\textbf{with}\{\textbf{val}_{a}\, x \mapsto N\} \\
\quad \cup\, \{\mathtt{op}_1(p_1), r_1 \mapsto {M_1}^{k}\}_{k\colon b \to a} \\
\quad \vdots \\
\quad \cup\, \{\mathtt{op}_n(p_n), r_n \mapsto {M_n}^{k}\}_{k\colon b \to a}
\end{array}
$$

Handlers for category-graded effects

If we have

- a functor $G \colon \mathbb{S} \to \mathbb{S}'$, $a \in \mathrm{Ob}\,\mathbb{S}$,
- $\phi \colon X \to \mathit{Term}_{\Sigma'}(\mathrm{id}_{Ga}, Y)$, and
- $|\mathbf{op}|_k \colon P \times \mathit{Term}_{\Sigma'}(Gk, Y)^Q \to \mathit{Term}_{\Sigma'}(G(k \circ f), Y)$ for each $\mathbf{op} \in \Sigma$, $k \colon b \to a$ in $\mathbb{S}$

then

$$\begin{tikzcd} X \arrow[r, "\eta_{a,X}"] \arrow[dr, "\phi"'] & \mathit{Term}_{\Sigma}(\mathrm{id}_a, X) \arrow[d, "\phi^{\dagger}_{\mathrm{id}_a}"] \\ & \mathit{Term}_{\Sigma'}(\mathrm{id}_{Ga}, Y) \end{tikzcd} \qquad \begin{tikzcd} \mathit{Term}_{\Sigma}(-, X) \arrow[d, dashed, "\phi^{\dagger}"] \\ \mathit{Term}_{\Sigma'}(G-, Y) \end{tikzcd}$$

Typing rules for handlers

By the above argument, we can construct a handler $H := \{\mathbf{val}_a\, x \mapsto N\} \cup \{\mathbf{op}(p), r \mapsto M^k_{\mathtt{op}}\}^k_{\mathtt{op}\in\Sigma}$ from the following judgements:

- $\Gamma, x : A \vdash^{\mathrm{id}_{Ga}} N : B$
- $\Gamma, p : P, r : Q \to B; Gk \vdash^{G(k \circ f)} M^k_{\mathtt{op}} : B$ for each $k : b \to a$, $\mathbf{op} : P \twoheadrightarrow Q; f \in \Sigma$.

Operational semantics is defined as follows:

$$\mathbf{handle}\, \mathbf{val}_a\, V\, \mathbf{with}\, H \to N[V/x]$$

$$\begin{aligned}&\mathbf{handle}\, \mathcal{E}[\mathrm{op}(V)]\, \mathbf{with}\, H \\ &\to M^k_{\mathtt{op}}[V/p, \lambda y.\, \mathbf{handle}(\mathcal{E}[\mathbf{val}_{Gb}\, y])\, \mathbf{with}\, H/r]\end{aligned}$$

Example of Handlers

$$G : \left(\begin{tikzcd} \mathit{nat} \arrow[loop above, "r_{nn}"] \arrow[loop below, "s_n"] \arrow[r, bend left, "r_{nb}"] \arrow[r, bend left=15, "\tau_b"'] & \mathit{bool} \arrow[loop above, "r_{bb}"] \arrow[loop below, "s_b"] \arrow[l, bend left=15, "\tau_n"'] \arrow[l, bend left, "r_{bn}"] \end{tikzcd} \right) \to \left(\; \mathit{nat} + \mathit{bool} \; \right)$$

$$x : A \vdash^{\mathrm{id}_{bool+nat}} \mathbf{val}\, x : A$$

$$\begin{aligned} & p : 1, r : \mathtt{nat} \to A; \mathrm{id} \vdash^{\mathrm{id}} \\ & \quad \mathbf{let}\, x \leftarrow \texttt{recv}()\, \mathbf{in} \\ & \quad \mathbf{match}\, x\, \mathbf{with} \\ & \qquad \mathrm{in}_1(n) \to r n \\ & \qquad \mathrm{in}_2(b) \to \mathtt{raise}() : A \end{aligned} \qquad \text{for } \texttt{recv}_{nn}, \cdots$$

$$\begin{aligned} & p : \mathtt{nat}, r : 1 \to A; \mathrm{id} \vdash^{\mathrm{id}} \\ & \quad \mathbf{let}\, \_ \leftarrow \texttt{send}(\mathrm{in}_1(p))\, \mathbf{in}\, r() : A \end{aligned} \qquad \text{for } \texttt{send}_n, \cdots$$

Conclusion

monad $T : \mathbf{1} \to \mathrm{Endo}(\mathbb{C})$	algebraic theory $\mathit{Term}_\Sigma(X)$	effect system with handlers $\Gamma \vdash M : A$
category-graded monad $T : \mathbb{S}^{\mathrm{op}} \to \mathrm{Endo}(\mathbb{C})$	category-graded algebraic theory $\mathit{Term}_\Sigma(f, X)$	category-graded effect system with handlers $\Gamma \vdash^f M : A$