How to construct graded monads

Dylan McDermott

Computational effects, with grades

Each computation has a grade $e \in \mathcal{G}$, where $(\mathcal{G}, \leq, \mathbf{1}, \cdot)$ is an ordered monoid

$$\frac{\Gamma \vdash V : A}{\Gamma \vdash \mathrm{return}\, V : A \,\&\, \mathbf{1}} \qquad \frac{\Gamma \vdash M_1 : A \,\&\, e_1 \qquad \Gamma, x : A \vdash M_2 : A \,\&\, e_2}{\Gamma \vdash \ (\texttt{do}\ x\ \texttt{<-}\ M_1\ \texttt{;}\ M_2) : A \,\&\, (e_1 \cdot e_2)} \qquad \frac{\Gamma \vdash M : A \,\&\, e \qquad e \leq e'}{\Gamma \vdash M : A \,\&\, e'}$$

Interpret computations using a graded monad R:

$$[\![\Gamma \vdash M : A \,\&\, e]\!] : [\![\Gamma]\!] \to Re[\![A]\!]$$

instead of a monad T:

$$[\![\Gamma \vdash M : A \,\&\, e]\!] : [\![\Gamma]\!] \to T[\![A]\!]$$

Example: may analysis for global state uses the ordered monoid

$$(\mathcal{P}\{\mathrm{get}, \mathrm{put}\}, \subseteq, \emptyset, \cup)$$

so that $e \subseteq \{\mathrm{get}, \mathrm{put}\}$

Example: backtracking with cut

$$\begin{aligned}&\texttt{or}(\texttt{or}(\texttt{or}(\texttt{or}(\texttt{return}\,\texttt{1}, \texttt{return}\,\texttt{2}), \texttt{fail}),\\ &\qquad\qquad \texttt{or}(\texttt{return}\,\texttt{3}, \texttt{cut})), \texttt{return}\,\texttt{4}) \;:\; \texttt{int} \,\&\, \top\end{aligned}$$

Effectful operations:

$$\frac{\Gamma \vdash M_1 : A \,\&\, e_1 \qquad \Gamma \vdash M_2 : A \,\&\, e_2}{\Gamma \vdash \texttt{or}(M_1, M_2) : A \,\&\, (e_1 \sqcap e_2)} \qquad \frac{}{\Gamma \vdash \texttt{fail} : A \,\&\, \top} \qquad \frac{}{\Gamma \vdash \texttt{cut} : A \,\&\, \bot}$$

Ordered monoid $(\{\bot, 1, \top\}, \leq, 1, \cdot)$:

$\top$ don't know anything

$\vee$|

1 definitely cuts or returns something

$\vee$|

$\bot$ definitely cuts

$$\begin{aligned}\top \cdot e &= \top\\ 1 \cdot e &= e\\ \bot \cdot e &= \bot\end{aligned}$$

Monads

A *monad* T consists of

- a set TX for each set X;
- a function $\mathit{return} : X \to TX$ for each X;
- a function $(\gg\!=) : TX \to (X \to TY) \to TY$ for each X, Y;

satisfying the monad laws:

$$\begin{aligned} \mathit{return}\, x \gg\!= f &= f\, x && \text{(left unit)} \\ t &= t \gg\!= \mathit{return} && \text{(right unit)} \\ (t \gg\!= f) \gg\!= g &= t \gg\!= \lambda y.(f y \gg\!= g) && \text{(associativity)} \end{aligned}$$

Graded monads

A *graded monad* R consists of

- an ordered monoid $(\mathcal{G}, \leq, \mathbf{1}, \cdot)$ (grades);
- a set ReX for each grade $e \in \mathcal{G}$, set X (computations of grade e);
- a function $return : X \to R\mathbf{1}X$ for each X;
- a function $(\overset{e_1,e_2}{\gg=}) : Re_1X \to (X \to Re_2Y) \to R(e_1 \cdot e_2)Y$ (bind) for each $e_1, e_2 \in \mathcal{G}$, X, Y;
- a function $R(e{\leq}e')X : ReX \to Re'X$ (subgrading) for each $e \leq e'$, X

satisfying the monad laws

$$return\, x \overset{\mathbf{1},e}{\gg=} f = f\, x \qquad \text{(left unit)}$$

$$r = r \overset{e,\mathbf{1}}{\gg=} return \qquad \text{(right unit)}$$

$$(r \overset{e_1,e_2}{\gg=} f) \overset{e_1 \cdot e_2, e_3}{\gg=} g = r \overset{e_1, e_2 \cdot e_3}{\gg=} \lambda y.(f\, y \overset{e_2,e_3}{\gg=} g) \qquad \text{(associativity)}$$

and some laws about subgrading

Example: may analysis for global state

For a non-empty set S of states:

- ordered monoid $(\mathcal{P}\{\mathrm{get}, \mathrm{put}\}, \subseteq, \emptyset, \cup)$
- sets of computations

$$R\emptyset X = X \qquad R\{\mathrm{get}\}X = S \Rightarrow X$$

$$R\{\mathrm{put}\}X = (1 + S) \times X \qquad R\{\mathrm{get}, \mathrm{put}\}X = S \Rightarrow S \times X$$

- $return = id : X \longrightarrow R\emptyset X$
- 16 cases of $\overset{e_1,e_2}{\gg\!=}$
- 9 cases of $R(e \leq e')$
- 64 associativity laws
- some other laws

Example: backtracking with cut

$\top$ don't know anything

$\leq$

1 definitely `cuts` or `returns` something

$\leq$

$\bot$ definitely `cuts`

$$
\begin{aligned}
ReX = \{(\mathrm{xs}, c) \in \mathrm{List}X \times \{\mathsf{cut}, \mathsf{nocut}\} \\
\mid (e = \bot \Rightarrow c = \mathsf{cut}) \\
\wedge\ (e = 1 \Rightarrow c = \mathsf{cut} \vee \mathrm{xs} \neq [\,])\}
\end{aligned}
$$

Gradings of monads

A *grading* R of a (non-graded) monad T consists of

- an ordered monoid $(\mathcal{G}, \leq, \mathbf{1}, \cdot)$
- a subset $ReX \subseteq TX$ for each $e \in \mathcal{G}$, set X

such that

- $ReX \subseteq Re'X$ for $e \leq e'$
- the return and bind functions

$$\mathit{return} : X \to TX \qquad (\gg\!=) : TX \to (X \to TY) \to TY$$

restrict to functions

$$\mathit{return} : X \to R\mathbf{1}X \qquad (\overset{e_1,e_2}{\gg\!=}) : Re_1X \to (X \to Re_2Y) \to R(e_1 \cdot e_2)Y$$

The restricted functions are the return and bind of a graded monad R, with subgrading functions $R(e \leq e')X : ReX \subseteq Re'X$

Gradings of monads

- Backtracking:

$$
\begin{aligned}
ReX = \{(\mathrm{xs}, c) \in TX \;\; | & \;\; (e = \bot \Rightarrow c = \text{cut}) \\
\wedge & \;\; (e = 1 \Rightarrow c = \text{cut} \vee \mathrm{xs} \neq [\,])\}
\end{aligned}
$$

where

$$TX = \mathrm{List}X \times \{\text{cut}, \text{nocut}\}$$

- Global state:

$$
\begin{aligned}
ReX \cong \{t \in TX & \\
| & \;\; (\text{put} \notin e \Rightarrow (\mathrm{fst} \circ t) \text{ is identity}) \\
\wedge & \;\; (\text{get} \notin e \Rightarrow (\mathrm{fst} \circ t) \text{ is constant or identity} \wedge (\mathrm{snd} \circ t) \text{ is constant})\}
\end{aligned}
$$

where

$$TX = S \Rightarrow S \times X$$

Gradings are good for program reasoning

If T forms an adequate model

$$\llbracket \Gamma \vdash M : A \rrbracket_{\mathsf{T}} : \llbracket \Gamma \rrbracket_{\mathsf{T}} \to T\llbracket A \rrbracket_{\mathsf{T}}$$
$$\llbracket M \rrbracket_{\mathsf{T}} = \llbracket N \rrbracket_{\mathsf{T}} \quad \Rightarrow \quad M \simeq_{\mathsf{ctx}} N$$

then any grading R of T also forms an adequate model

$$\llbracket M \rrbracket_{\mathsf{R}} = \llbracket N \rrbracket_{\mathsf{R}} \quad \Rightarrow \quad M \simeq_{\mathsf{ctx}} N \qquad \text{where } \llbracket \Gamma \vdash M : A \,\&\, e \rrbracket_{\mathsf{R}} : \llbracket \Gamma \rrbracket_{\mathsf{R}} \to Re\llbracket A \rrbracket_{\mathsf{R}}$$

but $\llbracket M \rrbracket_{\mathsf{R}} = \llbracket N \rrbracket_{\mathsf{R}}$ is usually weaker (and easier to prove) than $\llbracket M \rrbracket_{\mathsf{T}} = \llbracket N \rrbracket_{\mathsf{T}}$

How to construct graded monads

Supply some data:

1. a (non-graded) monad T;
2. an ordered set of grades $(\mathcal{G}, \leq)$, and unit grade $\mathbf{1}$;
3. a subset $ReX \subseteq TX$ for each $e \in \mathcal{G}$;
4. a multiplication $(\cdot) : \mathcal{G} \times \mathcal{G} \to \mathcal{G}$

such that $(\mathcal{G}, \leq, \mathbf{1}, \cdot)$ and R form a grading of T

The canonical grading of a monad

For each monad T, there is[1] an ordered monoid $(\mathbf{Sub}(\mathsf{T}), \sqsubseteq, I, \otimes)$, where

- $\mathbf{Sub}(\mathsf{T})$ is the set of *subfunctors* P of T, i.e. set-indexed families of subsets

$$PX \subseteq TX$$

  closed under $Tf = (\lambda t.\, t \gg\!= (f \circ \mathit{return})) : TX \to TY$ for each $f : X \to Y$
- $P \sqsubseteq P'$ iff $\forall X.\, PX \subseteq P'X$
- $IX = \{\mathit{return}\, x \mid x \in X\}$
- $(P_1 \otimes P_2)X = \{t \gg\!= f \mid Y \in \mathbf{Set}, t \in P_1 Y, f : Y \to P_2 X\}$

[1] ignoring some size issues

Constructing $(\cdot)$

A grading of T is equivalently

- an ordered monoid $(\mathcal{G}, \leq, \mathbf{1}, \cdot)$
- together with a lax homomorphism of ordered monoids $R : \mathcal{G} \longrightarrow \mathbf{Sub}(\mathsf{T})$

$$e \leq e' \;\Rightarrow\; Re \sqsubseteq Re' \qquad I \sqsubseteq R\mathbf{1} \qquad Re_1 \otimes Re_2 \sqsubseteq R(e_1 \cdot e_2)$$

So if the following is associative and unital, we get a grading:

$$e_1 \cdot e_2 = LRe_1 \otimes Re_2$$

assuming R has a left adjoint $L : \mathbf{Sub}(\mathsf{T}) \longrightarrow \mathcal{G}$:

$$\forall e \in \mathcal{G}.\; LP \leq e \quad \Leftrightarrow \quad P \sqsubseteq Re$$

Constructing $(\cdot)$

- For

$$ReX = \{(\mathrm{xs}, c) \in TX \;\mid\; (e = \bot \Rightarrow c = \mathsf{cut}) \;\wedge\; (e = 1 \Rightarrow c = \mathsf{cut} \vee \mathrm{xs} \neq [\,])\}$$

  we get

$$LP = \begin{cases} \bot & \text{if } \exists X.\, ([\,], \mathsf{nocut}) \in PX \\ \mathbf{1} & \text{if } \exists X, \mathrm{xs}.\, (\mathrm{xs}, \mathsf{nocut}) \in PX \\ \top & \text{otherwise} \end{cases} \qquad \begin{aligned} \top \cdot e &= \top \\ \mathbf{1} \cdot e &= e \\ \bot \cdot e &= \bot \end{aligned}$$

- For

$$ReX \cong \{t \in TX \;\mid\; (\mathsf{put} \notin e \Rightarrow (\mathrm{fst} \circ t) \text{ is identity}) \;\wedge\; (\mathsf{get} \notin e \Rightarrow (\mathrm{fst} \circ t) \text{ is constant or identity} \wedge (\mathrm{snd} \circ t) \text{ is constant})\}$$

  we get

$$LP = \{\mathsf{get}, \mathsf{put}\} \setminus \{\mathrm{op} \mid \forall X.\, R\{\mathrm{op}\}X \subseteq PX\}$$

$$e_1 \cdot e_2 = L(Re_1 \otimes Re_2) = e_1 \cup e_2$$

How to construct graded monads

Supply some data:

1. a (non-graded) monad T;
2. an ordered set of grades $(\mathcal{G}, \leq)$, and unit grade $\mathbf{1}$;
3. a subset $ReX \subseteq TX$ for each $e \in \mathcal{G}$;

such that $R : \mathcal{G} \to \mathbf{Sub}(\mathsf{T})$, and such that $(\mathcal{G}, \leq, \mathbf{1}, \cdot)$ and R form a grading of T

Constructing the subsets $ReX \subseteq TX$

Given a collection of operations $(\mathrm{op} : n)$, each with

- an algebraic operation
$$[\![\mathrm{op}]\!] : (TX)^n \longrightarrow TX$$
- a choice of grading function
$$\theta_{\mathrm{op}} : \mathcal{G}^n \longrightarrow \mathcal{G}$$

we can define R as the smallest family of subsets such that

- *return* restricts to a function $\mathit{return} : X \longrightarrow R\mathbf{1}X$
- $[\![\mathrm{op}]\!]$ restricts to a function $[\![\mathrm{op}]\!] : Rd_1X \times \cdots \times Rd_nX \longrightarrow R(\theta_{\mathrm{op}}(d_1, \ldots, d_n))X$

How to construct graded monads

Supply some data:

1. a (non-graded) monad T;
2. an ordered set of grades $(\mathcal{G}, \leq)$, and unit grade $\mathbf{1}$;
3. a subset $ReX \subseteq TX$ for each $e \in \mathcal{G}$
   (in many cases, generated by considering algebraic operations);

such that $R : \mathcal{G} \to \mathbf{Sub}(\mathsf{T})$, and such that $(\mathcal{G}, \leq, 1, \cdot)$ and R form a grading of T

An alternative: present a graded monad by graded operations and equations