Graded Modal Types

Category Theory

Category Theory Everywhere

Harley Eades III
School of Computer and Cyber Sciences
Augusta University
Granule Project

Team Augusta

- Harley Eades III (PI)
- Aubrey Bryant (PhD Student)

Team Kent

- Dominic Orchard (PI)
- Ben Moon (PhD Student)
- Jack Hughes (PhD Student)
Graded Monads & Effects
Monadic Effects

- State
- Exceptions
- Continuations
- Partiality
- Non-termination
- Errors
- Non-determinism
- Input/Output
- ..........
Effect Systems

- State
- Exceptions
- Continuations
- Partiality
- Non-termination
- Errors
- Non-determinism
- Input/Output
- ........

Strict Languages.
Monads + Effect Systems

Combined effect systems and monads.
Parametric Effect and Indexed Monads are now called **Graded Monads**.
So, what's a graded monad?
So, what's a graded monad?
A monad is just a monoid in the category of endofunctors.

What's the problem?
Monoids in Sets

\[ M : 1 \rightarrow \text{Set} \]

\[ \eta : \top \rightarrow M \]

\[ \mu : M \otimes M \rightarrow M \]
Monoids in $\mathcal{C}$

$M : 1 \rightarrow \mathcal{C}$

$\eta : \top \rightarrow M$

$\mu : M \otimes M \rightarrow M$
Monoids in $\mathcal{C}$

$M : 1 \rightarrow \mathcal{C}$

$\eta : \top \rightarrow M$

$\mu : M \otimes M \rightarrow M$
Monoid-Graded Monoids in $\mathcal{C}$

$(E, 0, +)$  \hspace{1cm} $\eta: T \to M_0$

$M : E \to \mathcal{C}$  \hspace{1cm} $\mu_{e_1, e_2}: M_{e_1} \otimes M_{e_2} \to M_{e_1 + e_2}$
Monoid-Graded Monoids in $\mathcal{C}$

$(E,0,+)$

$\eta : T \rightarrow M_0$

$M : E \rightarrow \mathcal{C}$

$\mu_{e_1,e_2} : M_{e_1} \otimes M_{e_2} \rightarrow M_{e_1 + e_2}$

Commutative-additive monoid
Monoid-Graded Monoids in $\mathcal{C}$

$(E, 0, +)$

$\eta : T \rightarrow M_0$

$\mu_{e_1, e_2} : M_{e_1} \otimes M_{e_2} \rightarrow M_{e_1 + e_2}$

$E$-indexed family of objects in $\mathcal{C}$
Monoid-Graded Monoids in $\mathcal{C}$

$\eta : T \to M_0$

$\mu_{e_1,e_2} : M_{e_1} \otimes M_{e_2} \to M_{e_1+e_2}$

A monoidal unit
Monoid-Graded Monoids in $\mathcal{C}$

$(E,0,+)$  \hspace{1cm} \eta : T \rightarrow M_0$

$M : E \rightarrow \mathcal{C}$  \hspace{1cm} $\mu_{e_1,e_2} : M_{e_1} \otimes M_{e_2} \rightarrow M_{e_1+e_2}$

A monoidal multiplication
Monoid-Graded Monoids in $\mathcal{C}$

$\langle E, 0, + \rangle$

$\eta : T \to M_0$

$M : E \to \mathcal{C}$

$\mu_{e_1, e_2} : M_{e_1} \otimes M_{e_2} \to M_{e_1 + e_2}$
Graded modalities are indexed-families of objects from one monoidal category to the another such that their tensor products are related in a lax or colax manner.
Monoids in $\mathcal{C}$

$M : 1 \to \mathcal{C}$

$\eta : \top \to M$

$\mu : M \otimes M \to M$
Monads

\[ M : 1 \rightarrow [\mathcal{C}, \mathcal{C}] \quad \eta : \text{Id} \rightarrow M \]

\[ \mu : M \circ M \rightarrow M \]
Monoid-Graded Monads

\[(E, 0, +) \quad \eta : \text{Id} \rightarrow M_0\]

\[M : E \rightarrow [C, C] \quad \mu_{e_1, e_2} : M_{e_1} \circ M_{e_2} \rightarrow M_{e_1 + e_2}\]
Graded Monads

$\eta : \text{Id} \to M_T$

$M : E \to [C, C] \quad \mu_{e_1, e_2} : Me_1 \circ Me_2 \to Me_1 \otimes e_2$
Example: Environment Monad

\[ M_X : \mathbf{1} \rightarrow [\text{Set}, \text{Set}] \]

\[ M_X(A) = X \Rightarrow A \]

\[ \eta_A : A \rightarrow M_X A \]

\[ \mu_A : M_X M_X A \rightarrow M_X A \]
Example: Environment Monad

\[ M_X : 1 \rightarrow [\text{Set}, \text{Set}] \]

\[ M_X(A) = X \Rightarrow A \]

\[ \eta_A : A \rightarrow M_X A \]

\[ \mu_A : M_X M_X A \rightarrow M_X A \]
Example: Environment Monad

$M_X : 1 \to [\text{Set}, \text{Set}]$

$M_X(A) = X \implies A$

$\eta_A : A \to M_X A$

$\mu_A : M_X M_X A \to M_X A$
Example: Powerset Monoid

\[ \mathcal{P}(X) : 1 \rightarrow \text{Set} \quad \emptyset : T \rightarrow \mathcal{P}(X) \]

\[ \cup : \mathcal{P}(X) \otimes \mathcal{P}(X) \rightarrow \mathcal{P}(X) \]
Example: Graded Environment Monad

\[ M : \mathcal{P}(E) \rightarrow [\text{Set}, \text{Set}] \]

\[ M_X(A) = X \Rightarrow A \]

\[ \eta_A : A \rightarrow M_\emptyset A \]

\[ \mu_A : M_XM_YA \rightarrow M_{X \cup Y}A \]

\[ \triangleright\triangleright = : M_XA \rightarrow (A \rightarrow M_YB) \rightarrow M_{X \cup Y}B \]
Towards a Formal Theory of Graded Monads

1 Department of Mathematics

3 Laboratoire Loria, Nancy, France

Abstract. We propose is to adapt by Street in the evident manner a monad can be factored along a left adjoint construction generally the Eilenberg-Moore construction on the graded state monad induced by any object \( V \) in a symmetric monoidal closed category \( \mathcal{C} \).
Typing for Graded Monads

Given: \((E, \top, \otimes, \leq)\)

\[
\begin{align*}
\Gamma &\vdash t : B \\
\Gamma &\vdash \langle t \rangle : M_{\top}B \\
\Gamma_1 &\leq \Gamma_2 \\
A &\leq B \\
\Gamma_1 &\vdash t : A \\
\Gamma_2 &\vdash t : B \\
\Gamma_2 &\vdash t_1 : M_{e_1}A \\
\Gamma_1, x : A &\vdash t_2 : M_{e_2}B \\
\Gamma_1, \Gamma_2 &\vdash \text{let } \langle x \rangle = t_1 \text{ in } t_2 : M_{e_1 \otimes e_2}B
\end{align*}
\]
Graded Comonads & Data Usage
Data as a Resource

- File handles
- Communication channels (session typing)
- Secure data
- Memory usage
- Time complexity
- Ordered data

.....
Data as a Resource

- File handles
- Communication channels (session typing)
- Secure data
- Memory usage
- Time complexity
- Ordered data
- .....
Intuitionistic Linear Logic

Supports the following data-usage constraints:

• Linear usage (one)
• Affine usage (one or none)
• Non-linear usage (tons)
Intuitionistic Linear Logic

\[
\frac{\Gamma_1, \Gamma_2 \vdash B}{\Gamma_1, \neg\neg A, \Gamma_2 \vdash B} \quad W
\]

\[
\frac{\Gamma_1, \neg\neg A, \neg\neg A, \Gamma_2 \vdash B}{\Gamma_1, \neg\neg A, \Gamma_2 \vdash B} \quad C
\]

\[
\frac{\neg\neg \Gamma \vdash B}{\neg\neg \Gamma \vdash \neg\neg B} \quad P
\]

\[
\frac{\neg\neg \Gamma \vdash \nega_1, \ldots, \neg\neg \Gamma_i \vdash \nega_i}{\neg\neg \Gamma_1, \ldots, \neg\neg \Gamma_i \vdash B} \quad D
\]
Intuitionistic Linear Logic

Supports the following data-usage constraints:

• Linear usage (one)
• Affine usage (one or none)
• Non-linear usage (tons)

What about the spectrum between none and tons?
Bounded Linear Logic

Supports the following data-usage constraints:

• none to tons

Time complexity!
(Simplified) Bounded Linear Logic

\[
\frac{\Gamma_1, \Gamma_2 \vdash B}{\Gamma_1, !_0 A, \Gamma_2 \vdash B} \quad \text{W}
\]

\[
\frac{\Gamma_1, !_{p_1} A, !_{p_2} A, \Gamma_2 \vdash B}{\Gamma_1, !_{p_1+p_2} A, \Gamma_2 \vdash B} \quad \text{C}
\]

\[
\frac{!_{\overrightarrow{p}} \Gamma \vdash B}{!_{\overrightarrow{p^*}} \Gamma \vdash !_{\overrightarrow{p}} B} \quad \text{P}
\]

\[
\frac{\Gamma, A \vdash B}{\Gamma, !_{1} A \vdash B} \quad \text{D}
\]
(Simplified) Bounded Linear Logic

\[
\frac{\Gamma_1, \Gamma_2 \vdash B}{\Gamma_1, !_0 A, \Gamma_2 \vdash B} \quad W
\]

\[
\frac{\Gamma_1, !_{p_1} A, !_{p_2} A, \Gamma_2 \vdash B}{\Gamma_1, !_{p_1+p_2} A, \Gamma_2 \vdash B} \quad C
\]

The precursor to graded comonads.

\[
\frac{!_{p} \Gamma \vdash B}{!_{p^*} \Gamma \vdash !_p B} \quad P
\]

\[
\frac{\Gamma, A \vdash B}{\Gamma, !_1 A \vdash B} \quad D
\]
Bounded Linear Logic in a Semiring

• Data-usage annotations are from a semiring
• Externally graded: no modality, all hypothesis are give a grade
Bounded Linear Logic in a Semiring

Given: \((R, 1, *, 0, +)\)

\[
\begin{align*}
&\frac{\Gamma_1, \Gamma_2 \vdash B}{\Gamma_1, A \otimes 0, \Gamma_2 \vdash B} \textcolor{red}{W} \\
&\frac{\Gamma_1, A \otimes r_1, A \otimes r_2, \Gamma_2 \vdash B}{\Gamma_1, A \otimes (r_1 + r_2), \Gamma_2 \vdash B} \textcolor{red}{C}
\end{align*}
\]
Graded comonads generalize the modality in bounded linear logic to use bounded semiring data-usage annotations.
Graded Comonads

Supports the following data-usage constraints:

- Linear usage (one)
- Affine usage (one or none)
- Non-linear usage (tons)
- None to tons
- Privacy
- Time complexity
- Session typing
Graded Comonads

\[
\frac{\Gamma_1, \Gamma_2 \vdash B}{\Gamma_1, \Box_0 A, \Gamma_2 \vdash B} \quad W
\]

\[
\frac{\Gamma_1, \Box_{r_1} A, \Box_{r_2} A, \Gamma_2 \vdash B}{\Gamma_1, \Box_{r_1 + r_2} A, \Gamma_2 \vdash B} \quad C
\]

\[
\frac{\Gamma, A \vdash B}{\Gamma, \Box_1 A \vdash B} \quad D
\]
Graded Comonads

\[
\begin{align*}
\Gamma_2 & \vdash \Box_r A \\
\Gamma_1, A \odot r & \vdash B \\
\hline
\Gamma_1, \Gamma_2 & \vdash B \\
\hline
\end{align*}
\]

\[
\begin{align*}
\odot \Gamma & \vdash B \\
\hline
p * \Gamma & \vdash \Box_p B
\end{align*}
\]

\text{P}
See you in the future.
Category-Graded Monads
Parameterised Monads

Monads parameterised by pre and post conditions:

\[ \eta : A \rightarrow P(I, I)A \]

\[ \mu : P(I, J)P(J, K)A \rightarrow P(I, K)A \]
Can graded monads and parameterised monads be unified?
Category-Graded Monads

Grades are morphisms in a category:

\[ \eta : A \rightarrow \square_{\text{id}_I} A \]

\[ \mu : \square_f \square_g A \rightarrow \square_{f;g} A \]
Subsume both graded monads and parameterised monads.


Graded Type Theory
Graded Modal Types

- Linear Base
- Graded Modalities
Graded Type Theory

Dependent Linear Base

Graded Modalities
Why Dependent Types?

- Practical programming with graded modalities requires dependency.
- Extrinsic verification.
Why Dependent Types?

map : forall {a : Type, b : Type}
    . (a -> b) []
    -> List a
    -> List b
map [f] Empty = Empty;
map [f] (Cons x xs) = Cons (f x) (map [f] xs)
map : \forall \{a : \text{Type}, b : \text{Type}, n : \text{Nat}\} . (a \to b) [n] \to \text{Vec}_n a \to \text{Vec}_n b

map [f] \text{Empty} = \text{Empty};
map [f] (\text{Cons } x \text{ xs}) = \text{Cons } (f \ x) (\text{map } [f] \text{ xs})
Graded Type Theory

Dependent Linear Base

Graded Modalities
Linear Dependent Types
Long standing open problem!
Linear Dependent Types

Non-Linear Dependent Type Theory:

\[ \Gamma_1, x : A, \Gamma_2 \vdash t : B \]
How should inputs be managed?

\[
\text{If } \Gamma_3 \vdash B : Type_0 \text{ then }
\]

\[
\Gamma_1, x : A, \Gamma_2 \vdash t : B
\]

Linear in the subject

In types?
How should inputs be managed?

\[ \text{If } \Gamma_3 \vdash B : Type \text{ and } l > 0 \text{ then} \]

In types?

\[ \Gamma_1, x : A, \Gamma_2 \vdash t : B \]

In the subject?

It depends on who you talk to!
How should inputs be managed?

- (McBride & Atkey) Quantitative Type Theory (QTT):
  - Specificational free variables are non-linear
  - Computational free variables are linear
- (Luo & Zhang) A Linear Dependent Type Theory
  - Use a weaker notion of linearity, but not fully non-linear
How should inputs be managed?

Dream: Users get to decide how their data is managed in both computations and specifications.
Linear Everywhere Dependent Type Theory (LEDTT)

Enforce linearity in both computations and specifications.
Linear Everywhere Dependent Type Theory (LEDTT)

Every variable must be used:

Let $\Gamma \vdash t : B$. For every $x : A \in \Gamma$ then either $x \in \text{FV}(\Gamma)$ or $x \in \text{FV}(t)$ or $x \in \text{FV}(B)$.

Linearity across judgments:

Let $\Gamma \vdash t : B$. For every $x : A \in \Gamma$ then $x$ appears only once in $\Gamma$, or only once in $t$, or only once in $B$. 
Linear Everywhere Dependent Type Theory (LEDTT)

Variable localization:

Let $\Gamma \vdash t : B$. For every $x : A \in \Gamma$ then the following holds:
- If $x \in \text{FV}(\Gamma)$, then $x \notin \text{FV}(t)$
- If $x \in \text{FV}(t)$, then $x \notin \text{FV}(\Gamma)$
Linear Everywhere Dependent Type Theory (LEDTT)

Key Concept: Usability of dependent types requires the ability to mix non-dependent types with dependent types, but linearity prevents the former leading to an unusable system.
Trivialization:

If \( \emptyset \vdash t : A \), then \( t \) is Type\(_{l_1}\) and \( A \) is Type\(_{l_2}\) for some \( l_1 \) and \( l_2 \) where \( l_1 < l_2 \).
LEDTT must be relaxed in order to regain the expressiveness of dependent types
Key idea: Double the grades

Graded Comonads:

\[ \Gamma_1, x : s A, \Gamma_2 \vdash t : B \]
Key idea: Double the grades

\[ \Gamma_1, x : \Delta \vdash A, \Gamma_2 \vdash t : B \]

where \( \Delta : \text{Vars.} \rightarrow \mathcal{R} \) is called a usage map.
Key idea: Double the grades

\[ \Gamma_1, x : \frac{\Delta}{7} A, y : B(x), z : C(x) \vdash t : D(x) \]

where

\[ \Delta := \{ y \mapsto 4, z \mapsto 42, \cdot \mapsto 2 \} \]
Example: Polymorphic Identity Function

\[ \emptyset \vdash \lambda a. \lambda x. x : (a : \text{Type}) \rightarrow (x : a) \rightarrow a \]
Example: Polymorphic Identity Function

\[ \emptyset \vdash \lambda [a]. \lambda [x]. x : (a : ^2_0 \text{ Type}) \to (x : ^0_1 a) \to a \]
Graded Type Theory (GrDTT)

GrTT = LEDTT + Graded Types

Demo Time!
Granule Design and Meta-theory

D. Orchard, V. Liepelt, H. Eades III.
"Quantitative Program Reasoning with Graded Modal Types."
In ICFP 2019.
Thank you!

Contacts:
Twitter: @heades
Email: harley.eades@gmail.com
Blog: blog.metatheorem.org

https://granule-project.github.io/

Download & Install Granule
Backup Slides