# Graded Modal Types


**Harley Eades III**
**School of Computer and Cyber Sciences**
**Augusta University**

# Granule Project

## Team Augusta

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

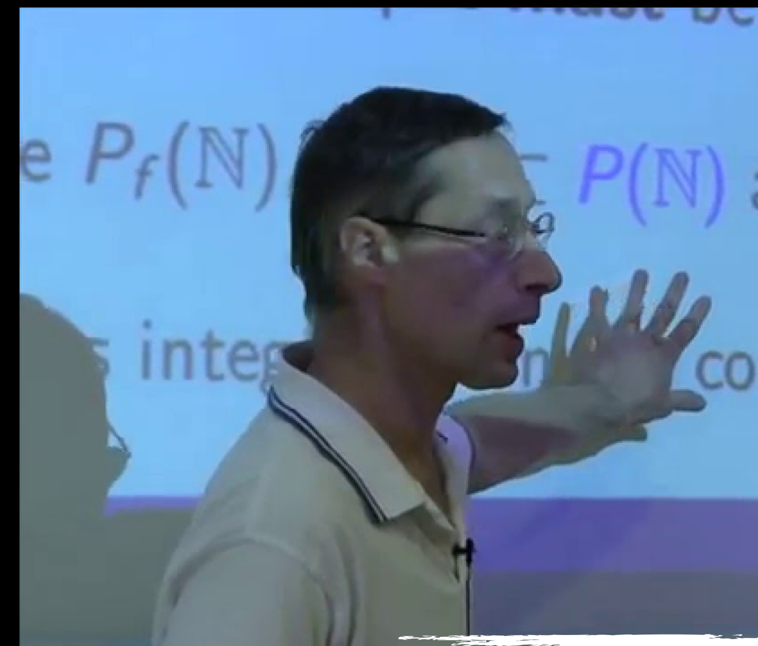
## Team Kent

- 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
- ........



Lazy Languages.

# Effect Systems

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

**Strict Languages.**

# Monads + Effect Systems

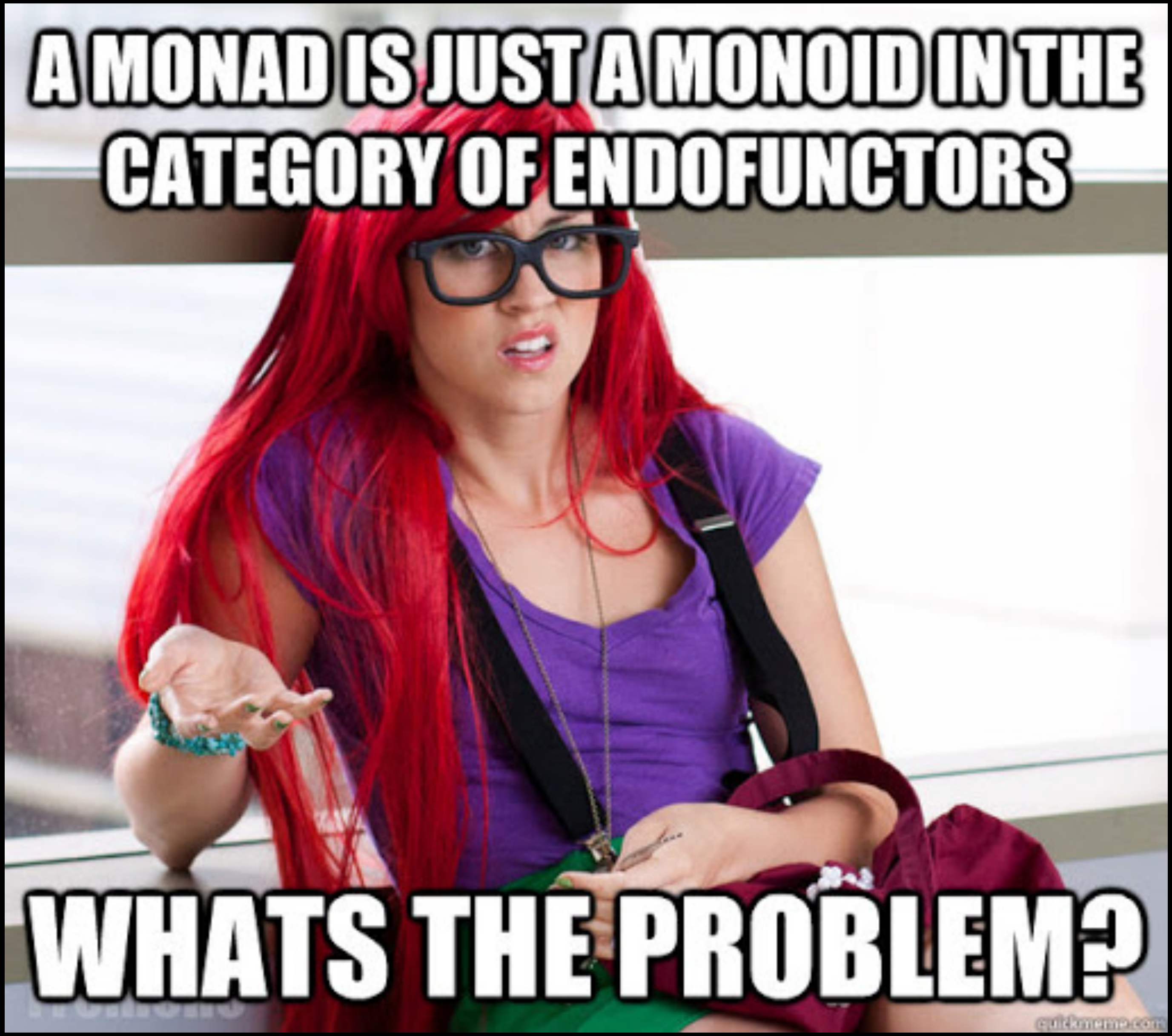
**Combined effect systems and monads.**

**Parametric Effect and Indexed Monads are now called Graded Monads.**

# So, what's a graded monad?

monoids

So, what's a graded ~~monad~~?


A MONAD IS JUST A MONOID IN THE CATEGORY OF ENDOFUNCTORS
WHATS THE PROBLEM?

# Monoids in Sets

$M : \mathbf{1} \to \mathsf{Set}$

$\eta : \top \to M$

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

# Monoids in $\mathscr{C}$

$M : \mathbf{1} \to \mathscr{C}$

$\eta : \top \to M$

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

# Monoids in $\mathscr{C}$

$$\boxed{M : \mathbf{1} \to \mathscr{C}}$$

$$\eta : \top \to M$$

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

# Monoid-Graded Monoids in $\mathscr{C}$

$(\mathsf{E},0,+)$ $\qquad$ $\eta : \top \to M_0$

$M : \mathsf{E} \to \mathscr{C}$ $\qquad$ $\mu_{e_1,e_2} : M_{e_1} \otimes M_{e_2} \to M_{e_1+e_2}$

# Monoid-Graded Monoids in $\mathscr{C}$

$\boxed{(\mathsf{E}, 0, +)}$

$$\eta : \top \to M_0$$

$$M : \mathsf{E} \to \mathscr{C} \qquad \mu_{e_1,e_2} : M_{e_1} \otimes M_{e_2} \to M_{e_1+e_2}$$

**Commutative-additive monoid**

# Monoid-Graded Monoids in $\mathscr{C}$

$(\mathsf{E},0,+)$ $\qquad$ $\eta : \top \to M_0$

$\boxed{M : \mathsf{E} \to \mathscr{C}}$ $\qquad$ $\mu_{e_1,e_2} : M_{e_1} \otimes M_{e_2} \to M_{e_1+e_2}$

**$\mathsf{E}$-indexed family of objects in $\mathscr{C}$**

# Monoid-Graded Monoids in $\mathscr{C}$

$(\mathsf{E},0,+)$ $\boxed{\eta : \top \to M_0}$

$M : \mathsf{E} \to \mathscr{C}$ $\mu_{e_1,e_2} : M_{e_1} \otimes M_{e_2} \to M_{e_1+e_2}$

**A monoidal unit**

# Monoid-Graded Monoids in $\mathscr{C}$

$(\mathsf{E}, 0, +)$

$\eta : \top \to M_0$

$M : \mathsf{E} \to \mathscr{C}$

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

**A monoidal multiplication**

# Monoid-Graded Monoids in $\mathscr{C}$

$(\mathsf{E},0,+)$

$\eta : \top \to M_0$

$M : \mathsf{E} \to \mathscr{C}$

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

**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 $\mathscr{C}$

$M : \mathbf{1} \to \mathscr{C}$

$\eta : \top \to M$

$\mu : M \bigotimes M \to M$

# Monads

$$M : \mathbf{1} \to [\mathscr{C}, \mathscr{C}] \qquad \eta : \mathtt{Id} \to M$$

$$\mu : M \circ M \to M$$

# Monoid-Graded Monads

$(\mathsf{E},0,+)$ $\eta : \mathsf{Id} \to M_0$

$M : \mathsf{E} \to [\mathscr{C}, \mathscr{C}]$ $\mu_{e_1,e_2} : M_{e_1} \circ M_{e_2} \to M_{e_1+e_2}$

# Graded Monads

$(\mathsf{E}, \top, \otimes)$ $\eta : \mathtt{Id} \to M_\top$

$M : \mathsf{E} \to [\mathscr{C}, \mathscr{C}]$ $\mu_{e_1,e_2} : M_{e_1} \circ M_{e_2} \to M_{e_1 \otimes e_2}$

# Example : Environment Monad

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

$$M_X(A) = X \Rightarrow A$$

$$\eta_A : A \to M_X A$$

$$\mu_A : M_X M_X A \to M_X A$$

# Example : Environment Monad

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

$M_X(A) = X \Rightarrow A$

$\eta_A : A \to M_X A$

$\boxed{\mu_A : M_X M_X A \to M_X A}$

# Example : Environment Monad

$\boxed{M_X : \mathbf{1} \to [\mathsf{Set}, \mathsf{Set}]}$

$$M_X(A) = X \Rightarrow A$$

$$\eta_A : A \to M_X A$$

$$\mu_A : M_X M_X A \to M_X A$$

# Example : Powerset Monoid

$\mathscr{P}(X) : \mathbf{1} \to \mathsf{Set}$ $\quad$ $\varnothing : \top \to \mathscr{P}(X)$

$\cup : \mathscr{P}(X) \otimes \mathscr{P}(X) \to \mathscr{P}(X)$

# Example : Graded Environment Monad

$\boxed{M : \mathscr{P}(E) \to [\mathsf{Set}, \mathsf{Set}]}$

$\eta_A : A \to M_{\emptyset} A$

$M_X(A) = X \Rightarrow A$

$\boxed{\mu_A : M_X M_Y A \to M_{X \cup Y} A}$

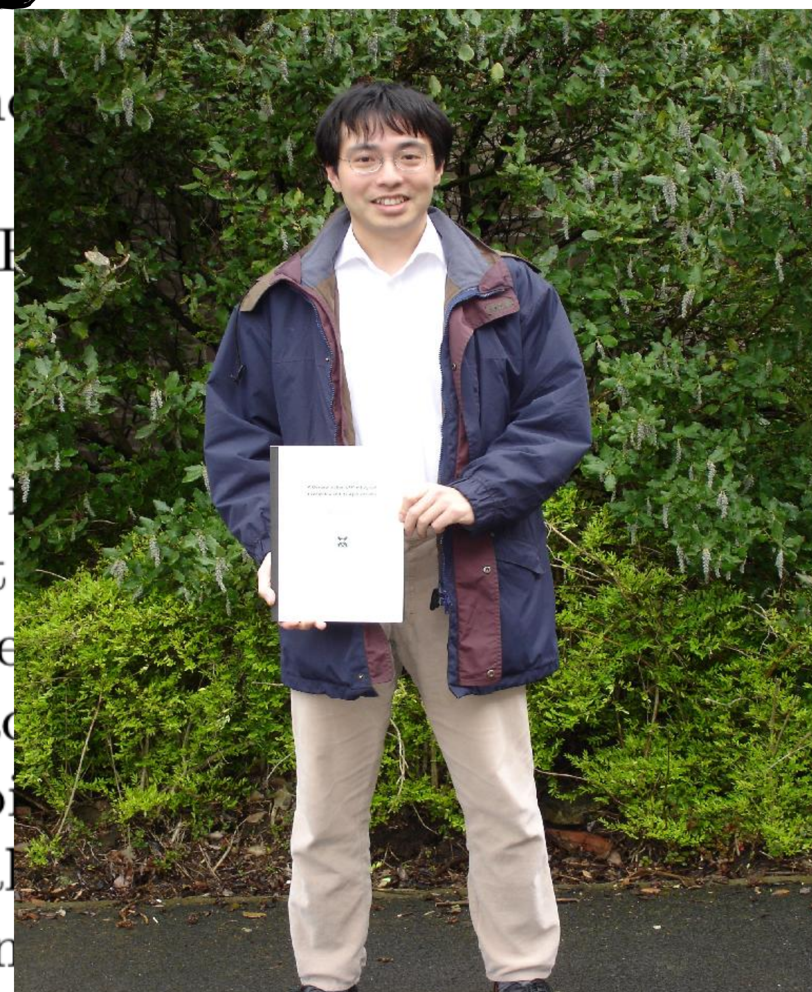
$$\texttt{>>=} : M_x A \to (A \to M_Y B) \to M_{X \cup Y} B$$

# Towards a Formal Theory of Graded Monads


**Soichiro Fujii** , Shin-ya Katsumata[2], and Paul-André Melliès[3]

[1] Departm

[3] Laboratoire I


**Abstract.** We
pose is to adapt
by Street in the e … articular that every graded
monad can be fac … a strict action transported
along a left adjo … n what sense the first con-
struction general … struction while the second
construction gen … ion. Finally, we illustrate
the Eilenberg-Moore construction on the graded state monad induced
by any object $V$ in a symmetric monoidal closed category $\mathscr{C}$.

# Typing for Graded Monads

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

$$\frac{\Gamma \vdash t : B}{\Gamma \vdash \langle t \rangle : M_{\top} B}\ \eta$$

$$\frac{\Gamma_1 \leq \Gamma_2 \quad A \leq B \qquad \Gamma_1 \vdash t : A}{\Gamma_2 \vdash t : B}\ \leq$$

$$\frac{\Gamma_2 \vdash t_1 : M_{e_1} A \qquad \Gamma_1, x : A \vdash t_2 : M_{e_2} B}{\Gamma_1, \Gamma_2 \vdash \mathsf{let}\ \langle x \rangle = t_1\ \mathsf{in}\ t_2 : M_{e_1 \otimes e_2} B}\ \mu$$

# 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
- .....

**Misusing data can lead to various bugs.**

# Intuitionistic Linear Logic

**Supports the following data-usage constraints:**

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

# Intuitionistic Linear Logic

$$\frac{\Gamma_1, \Gamma_2 \vdash B}{\Gamma_1, !A, \Gamma_2 \vdash B}\ W$$

$$\frac{\Gamma_1, !A, !A, \Gamma_2 \vdash B}{\Gamma_1, !A, \Gamma_2 \vdash B}\ C$$

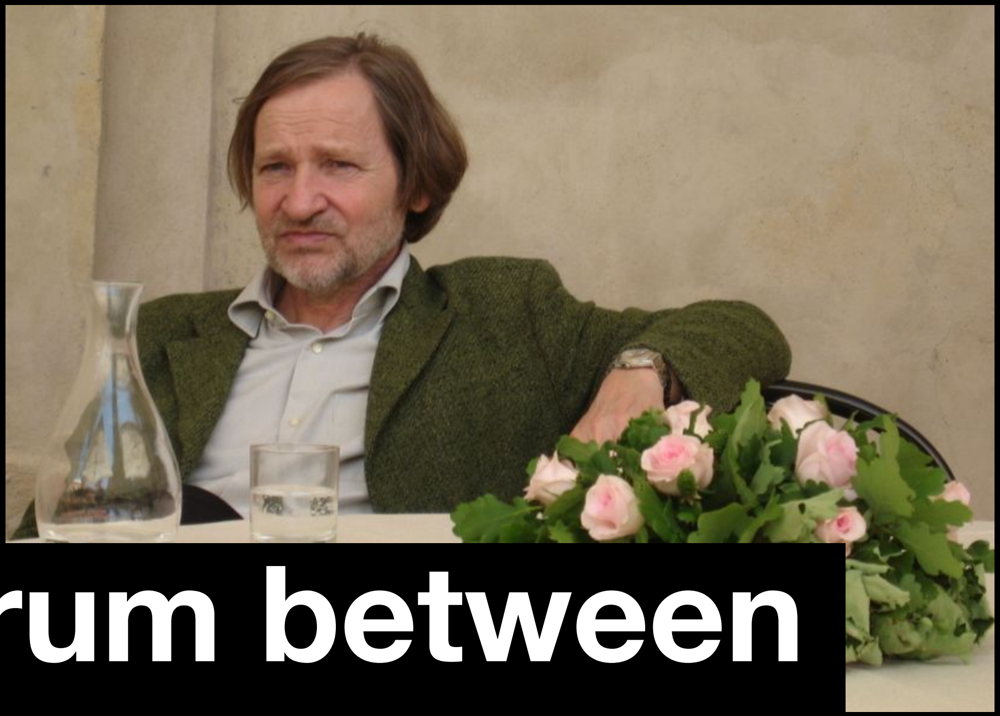
$$\frac{!\Gamma \vdash B}{!\Gamma \vdash !B}\ P$$

$$\frac{\Gamma_1 \vdash !A_1, \ldots, \Gamma_i \vdash !A_i \qquad !A_1, \ldots, !A_i \vdash B}{\Gamma_1, \ldots, \Gamma_i \vdash B}\ D$$

# Intuitionistic Linear Logic

**Supports the following data-usage constraints:**

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

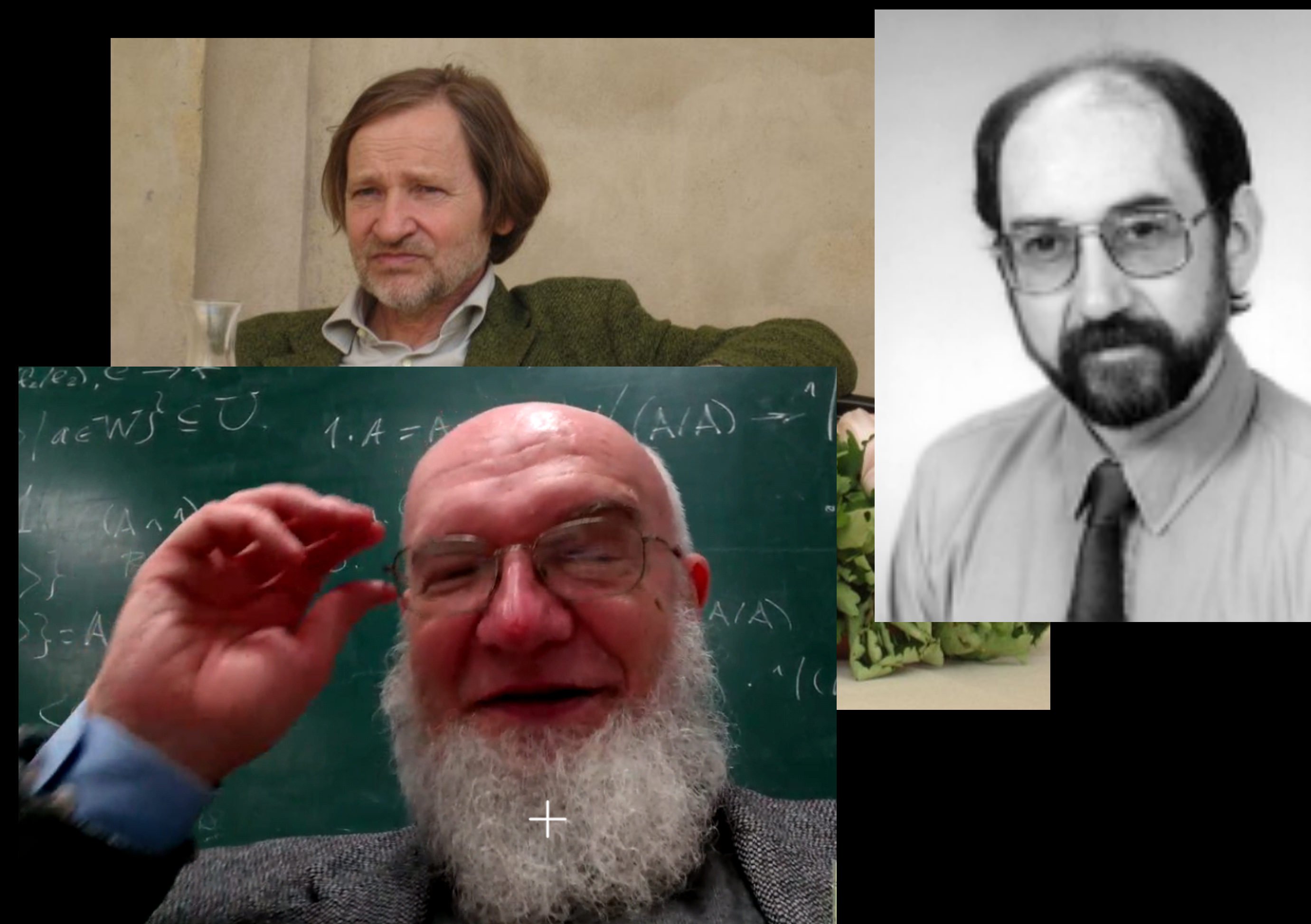


**What about the spectrum between none and tons?**

# Bounded Linear Logic

**Supports the following data-usage constraints:**

- **none to tons**



**Time complexity!**

# (Simplified) Bounded Linear Logic

$$\frac{\Gamma_1, \Gamma_2 \vdash B}{\Gamma_1, !_0 A, \Gamma_2 \vdash B}\ W$$

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

$$\frac{!_{\overrightarrow{p}} \Gamma \vdash B}{!_{p * \overrightarrow{p}} \Gamma \vdash !_p B}\ P$$

$$\frac{\Gamma, A \vdash B}{\Gamma, !_1 A \vdash B}\ D$$

# (Simplified) Bounded Linear Logic

$$\frac{\Gamma_1, \Gamma_2 \vdash B}{\Gamma_1, !_0 A, \Gamma_2 \vdash B}\ W$$

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

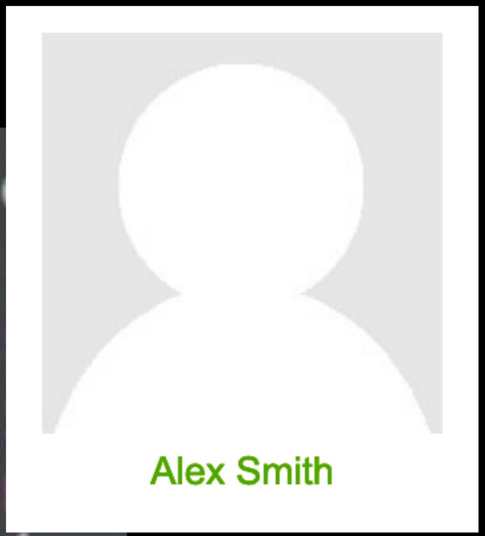
**The precursor to graded comonads.**

$$\frac{!_{\overrightarrow{p}} \Gamma \vdash B}{!_{p * \overrightarrow{p}} \Gamma \vdash !_p B}\ P$$

$$\frac{\Gamma, A \vdash B}{\Gamma, !_1 A \vdash B}\ D$$

# Bounded Linear Logic in a Semiring

- **Data-usage annotations are from a semiring**
- **<u>Externally graded</u>: no modality, all hypothesis are give a grade**



# Bounded Linear Logic in a Semiring

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

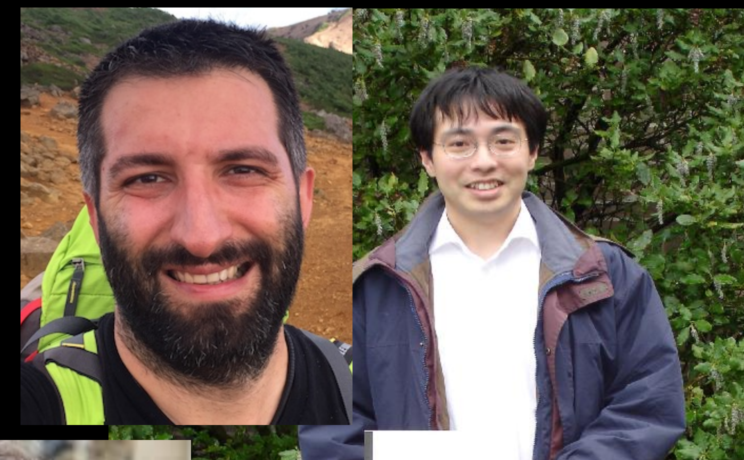
$$\frac{\Gamma_1, \Gamma_2 \vdash B}{\Gamma_1, A \odot 0, \Gamma_2 \vdash B}\, W \qquad \frac{\Gamma_1, A \odot r_1, A \odot r_2, \Gamma_2 \vdash B}{\Gamma_1, A \odot (r_1 + r_2), \Gamma_2 \vdash B}\, C$$

**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

$$\frac{\Gamma_1, \Gamma_2 \vdash B}{\Gamma_1, \Box_0 A, \Gamma_2 \vdash B}\; W$$

$$\frac{\Gamma_1, \Box_{r_1} A, \Box_{r_2} A, \Gamma_2 \vdash B}{\Gamma_1, \Box_{r_1+r_2} A, \Gamma_2 \vdash B}\; C$$

$$\frac{\Gamma, A \vdash B}{\Gamma, \Box_1 A \vdash B}\; D$$

# Graded Comonads

$$\frac{\Gamma_2 \vdash \Box_r A \qquad \Gamma_1, A \odot r \vdash B}{\Gamma_1, \Gamma_2 \vdash B}\ \Box_e$$

$$\frac{\odot\Gamma \vdash B}{p * \Gamma \vdash \Box_p B}\ P$$


See you in the future.

# Category-Graded Monads

# Parameterised Monads

**Monads parameterised by pre and post conditions:**

$$\eta : A \to P(I, I)A$$

$$\mu : P(I, J)P(J, K)A \to P(I, K)A$$

**Can graded monads and parameterised monads be unified?**

# Category-Graded Monads

**Grades are morphisms in a category:**

$$\eta : A \to \square_{\mathsf{id}_I} A$$

$$\mu : \square_f \square_g A \to \square_{f;g} A$$

# Category-Graded Monads

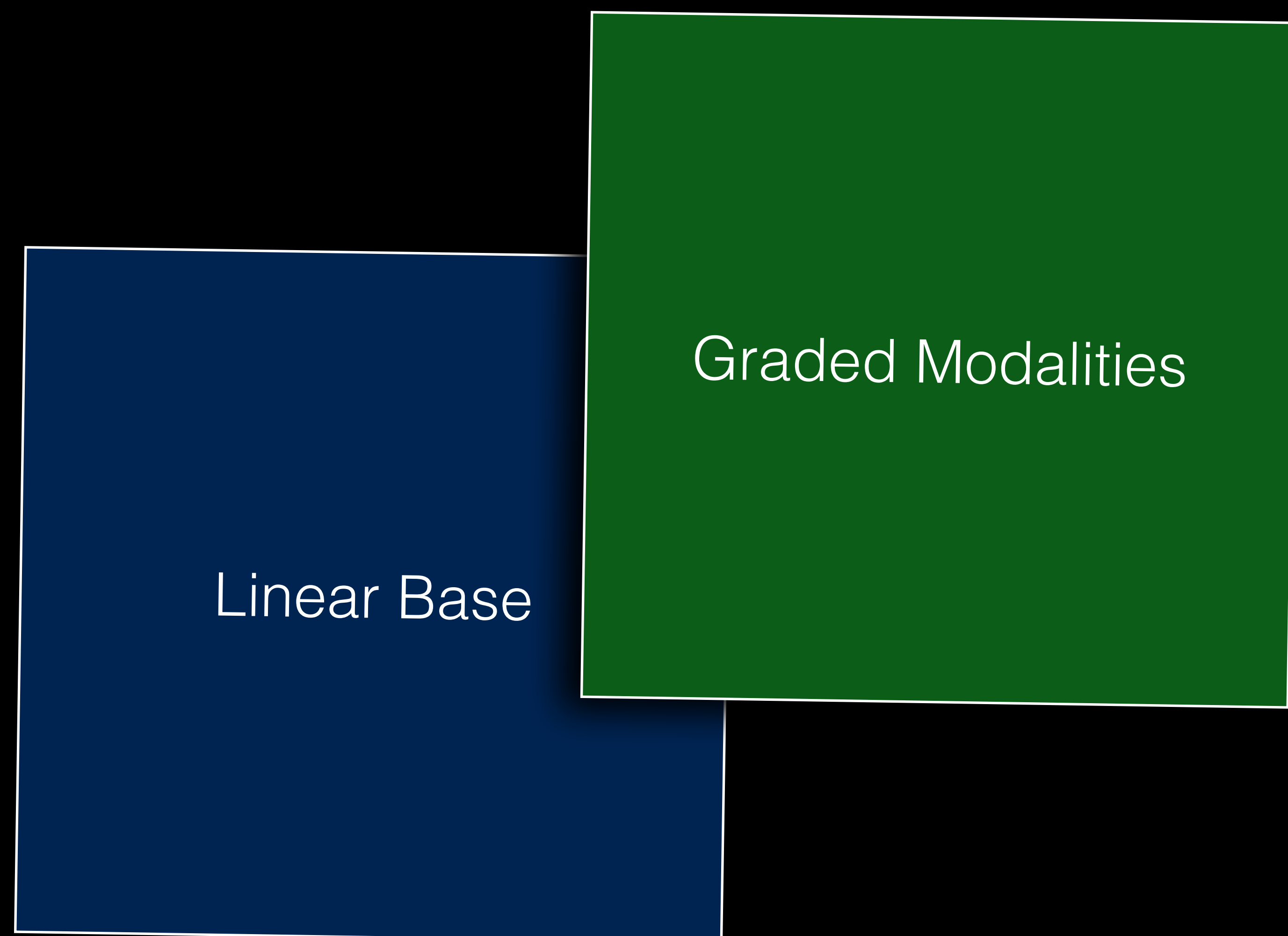
**Subsume both graded monads and parameterised monads.**

**D. Orchard, P. Wadler, H. Eades III. "Unifying graded and parameterised monads". Under review MSFP 2020.**

**Preprint: https://arxiv.org/abs/2001.10274**

# Graded Type Theory

# Graded Modal Types



# Graded Type Theory

Graded Modalities

Dependent
Linear Base

# 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)
```

# Why Dependent Types?

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

# Graded Type Theory

Graded Modalities

Dependent
Linear Base

# Linear Dependent Types

## **Long standing open problem!**

Dependent
Linear Base

# Linear Dependent Types

**Non-Linear Dependent Type Theory:**

**In types**

$$\Gamma_1, x : A, \Gamma_2 \vdash t : B$$

**In the subject**

# How should inputs be managed?

**If** $\Gamma_3 \vdash B : Type_0$ **then**

**In types ?**

$$\Gamma_1, x : A, \Gamma_2 \vdash t : B$$

**Linear in the subject**

# How should inputs be managed?

**If** $\Gamma_3 \vdash B : Type_l$ **and** $l > 0$ **then**

It depends on who you talk to!

**In types ?**

$$\Gamma_1, x : A, \Gamma_2 \vdash t : B$$

**In the subject ?**

# How should inputs be managed?

- (McBride & Atkey) Quantitative Type Theory (QTT):
  - Specificational free [redacted]non-linear
  - Comput[redacted]ables are linear
- (Luo & Zh[redacted]g) A Linear Dependent Type Theory
  - Use a weaker notion of linearity, but not fully non-linear

Why?

# 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 $x \in \mathsf{FV}(\Gamma)$ or $x \in \mathsf{FV}(t)$ or $x \in \mathsf{FV}(B)$.

**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 $x \in \mathsf{FV}(\Gamma)$, then $x \notin \mathsf{FV}(t)$
- If $x \in \mathsf{FV}(t)$, then $x \notin \mathsf{FV}(\Gamma)$

# 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.

# Linear Everywhere Dependent Type Theory (LEDTT)

**Trivialization:**

If $\varnothing \vdash t : A$, then $t$ is $\mathsf{Type}_{l_1}$ and $A$ is $\mathsf{Type}_{l_2}$ 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

$$\Gamma_1, x :_s^{\Delta} A, \Gamma_2 \vdash t : B$$

**where** $\Delta : \mathsf{Vars}_{\bullet} \to \mathscr{R}$ **is called a usage map.**

# Key idea: Double the grades

$$\Gamma_1, x :^{\Delta}_{7} A, y : B(x), z : C(x) \vdash t : D(x)$$

**where**

$$\Delta := \{y \mapsto 4, z \mapsto 42, \bullet \mapsto 2\}$$

# Example : Polymorphic Identity Function

$$\varnothing \vdash \lambda a \,.\, \lambda x \,.\, x : (a : \mathsf{Type}) \multimap (x : a) \multimap a$$

# Example : Polymorphic Identity Function

$$\varnothing \vdash \lambda[a] \,.\, \lambda[x] \,.\, x : (a :_0^2 \mathsf{Type}) \multimap (x :_1^0 a) \multimap a$$

# Graded Type Theory (GrDTT)

$\boxed{\textbf{GrTT = LEDTT + Graded Types}}$

**H. Eades III, B. Moon, and D. Orchard. "Graded Type Theory." Under review at LICS 2020.**

# Demo Time!

# Granule Design and Meta-theory

**D. Orchard, V. Liepelt, H. Eades III.**
**"Quantitative Program Reasoning with Graded Modal Types."**
**In ICFP 2019.**
**PDF: http://metatheorem.org/includes/pubs/ICFP19.pdf**

# Thank you!

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

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

**Download & Install Granule**

# Backup Slides