Granule: A language for fine-grained reasoning via graded modal types

Dominic Orchard

http://dorchard.co.uk

http://github.com/dorchard/granule

University of Kent

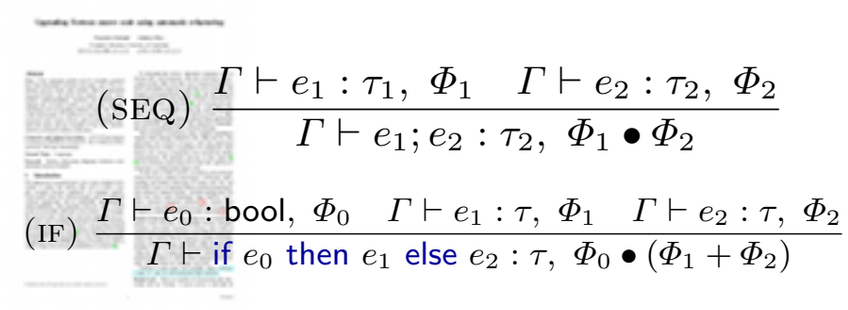
Modern programming challenges

- Manage sensitive private information
- Obey stateful protocols
- React to and understand the effect of uncertainty

+ concurrent & distributed

The types-for-verification paradigm

Undesirable program behaviour



$$(\text{SEQ})\ \frac{\Gamma \vdash e_1 : \tau_1,\ \Phi_1 \quad \Gamma \vdash e_2 : \tau_2,\ \Phi_2}{\Gamma \vdash e_1; e_2 : \tau_2,\ \Phi_1 \bullet \Phi_2}$$

$$(\text{IF})\ \frac{\Gamma \vdash e_0 : \mathsf{bool},\ \Phi_0 \quad \Gamma \vdash e_1 : \tau,\ \Phi_1 \quad \Gamma \vdash e_2 : \tau,\ \Phi_2}{\Gamma \vdash \mathsf{if}\ e_0\ \mathsf{then}\ e_1\ \mathsf{else}\ e_2 : \tau,\ \Phi_0 \bullet (\Phi_1 + \Phi_2)}$$

New type system

rarely

New language / extension

- Bespoke
- Domain-specific
- Highly coupled

e.g. ownership types 

effect types Idris

Granule - fine-grained program reasoning

Track information
for quantitative reasoning

Goal: extensible

Be strict about resources

Precision

Goal: full dependent types

Graded
modal
types

SMT
solver
(e.g. Z3)

Linear
types

Indexed
types

Granule demo 1

linear basis

□ modality — use any number of times (or !)

linear types — use exactly once

Modal logic - possibility & necessity

$\Box A$ = A is "necessarily" true, A true in all worlds

Axioms: $\Box A \to A$

$\Box A \to \Box\Box A$

$\Box(A \to B) \to \Box A \to \Box B$

Rule: $\dfrac{\vdash A}{\vdash \Box A}$

$\Diamond A$ = A "possibly" true, A true in some worlds

Axioms: $A \to \Diamond A$

$\Diamond\Diamond A \to \Diamond A$

$(A \to B) \to \Diamond A \to \Diamond B$

Graded modalities (informally)

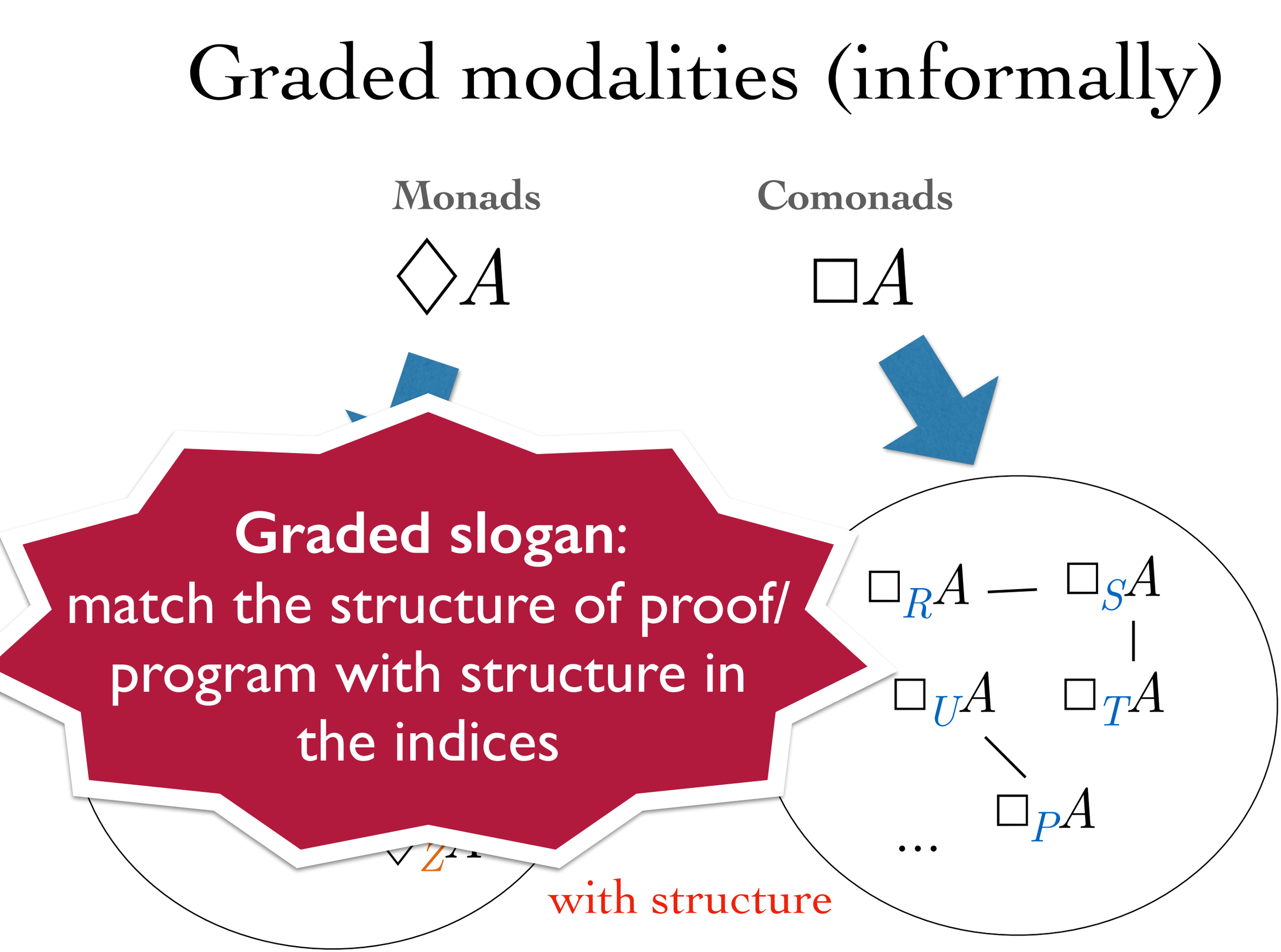
Monads

$\Diamond A$

Comonads

$\Box A$

Graded slogan:
match the structure of proof/ program with structure in the indices



with structure

Graded monads

Graded comonads

□ modality — use any number of times (or !)

linear types — use exactly once

□∞ modality — use any number of times

□n modality — use at most n number of times

linear types — use exactly once

Bounded Linear Logic

(!n in Girard et al. '92)

Family: $\Box n\, A$ *where* $n \in \mathbb{N}$ — A can be used n times

Structure: $(\mathbb{N}, *, 1, +, 0)$

Axioms:

$$\Box_{r * s}\, A \to \Box_r \Box_s A$$

$$\Box_1 A \to A$$

$$(A \to B) \to \Box_r A \to \Box_r B$$

$$\Box_0 A \to 1$$

$$\Box_{r+s}\, A \to \Box_r A \times \Box_s A$$

$$\Box_s A \to \Box_r A \quad \textit{where } r \leq s$$

Granule demo 2

bounded linearity

$\Box_n A$ written in Granule as type `A |n|`

and indexed types

Bounded Linear Logic

(!n in Girard et al. '92)

Family: $\Box\, n\, A$ *where* $n \in \mathbb{N}$ — A can be used n times

Structure: $(\mathbb{N}, *, 1, +, 0)$

Axioms:

$$\Box_{r * s}\ A \to \Box_r \Box_s A$$

$$\Box_1 A \to A$$

$$(A \to B) \to \Box_r A \to \Box_r B$$

$$\Box_0 A \to 1$$

$$\Box_{r+s}\ A \to \Box_r A \times \Box_s A$$

$$\Box_s A \to \Box_r A \quad \textit{where } r \leq s$$

Semiring-graded necessity

Family: $\Box\, n\ A$

Structure: $(R,\ *,\ 1,\ +,\ 0,\ \leq)$ **... (ordered) semiring**

Axioms:

$$\Box_{r * s}\ A \to \Box_r\, \Box_s\, A$$

$$\Box_1 A \to A$$

$$(A \to B) \to \Box_r\, A \to \Box_r\, B$$

$$\Box_0 A \to 1$$

$$\Box_{r+s} A \to \Box_r A \times \Box_s A$$

$$\Box_s A \to \Box_r A \quad where\ r \leq s$$

also known as "coeffects"
modelled by graded comonads

Granule demo 3

security coeffects

$(\{\text{Private}, \text{Public}\}, \wedge, \text{Private}, \vee, \text{Public})$

$\Box_{l}\, A$ written in Granule as `A |1|`

Linear types with semiring-graded necessity

$$\text{ax}\,\frac{}{x : A \vdash x : A} \qquad \text{abs}\,\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \to B} \qquad \text{app}\,\frac{\Gamma \vdash t : A \to B \quad \Delta \vdash t' : A}{\Gamma + \Delta \vdash t\,t' : B}$$

Resource accounting: $r, s \in (\mathcal{R}, *, 1, +, 0, \leq)$

Types: $A, B ::= A \to B \mid \Box_r A \mid c$

`A |r|`

Contexts: $\Gamma, \Delta ::= A, \Gamma \mid \underbrace{\boxdot_r A}, \Gamma \mid \cdot$

non-linear variable (discharged modality)

Contraction:

$(x : A, \Gamma) + (x : A, \Delta)$ *is ill-formed*

$(x : \Box_r A, \Gamma) + (x : \Box_s A, \Delta) \quad = \quad x : \Box_{r+s} A, (\Gamma + \Delta)$

Weakening: $\dfrac{\Gamma \vdash t : A}{\Gamma, \boxdot_0 \Delta \vdash t : A}$

Linear types with semiring-graded necessity

Shift linear variable to modal: **(derelection)**

$$\mathrm{der}\,\frac{\Gamma, x : A \vdash t : B}{\Gamma, x : \boxdot_1 A \vdash t : B}$$

Propagate grading: **(promotion)**

$$\mathrm{pr}\,\frac{\boxdot\Gamma \vdash t : B}{r * \boxdot\Gamma \vdash [t] : \Box_r B}$$

Composition/cut:

$$\mathrm{let}\Box\,\frac{\Gamma \vdash t_1 : \Box_r A \qquad \Delta, x : \boxdot_r A \vdash t_2 : B}{\Gamma + \Delta \vdash \mathbf{let}\ [x] = t_1\ \mathbf{in}\ t_2 : B}$$

Effects via graded possibility (graded monads)

Graded necessity:

$$\Box_{r * s}\ A \to \Box_r \Box_s A$$

$$\Box_1 A \to A$$

$$(A \to B) \to \Box_r A \to \Box_r B$$

Graded possibility:

$$\Diamond_x \Diamond_y A \to \Diamond_{x \oplus y} A$$

$$A \to \Diamond_I A$$

$$(A \to B) \to \Diamond_x A \to \Diamond_x B$$

Meta-language style: (do in Haskell**)**

$$\frac{\Gamma \vdash e : A}{\Gamma \vdash \langle e \rangle : \Diamond_I A}$$

$$\frac{\Gamma \vdash e_1 : \Diamond_x A \quad \Gamma, v : A \vdash e_2 : \Diamond_y B}{\Gamma \vdash \mathbf{let}\ \Diamond v = e_1\ \mathbf{in}\ e_2 : \Diamond_{x \oplus y} B}$$

Granule demo 4

revisiting the file handling demo

effect-graded possibility

$$(X,\ \oplus,\ I) = (\text{List}\ \{\text{R},\ \text{W},\ \text{C},\ \text{RW},\ \text{O}\},\ ++,\ [])$$

$\Diamond x\ A$ written in Granule as `A < x >`

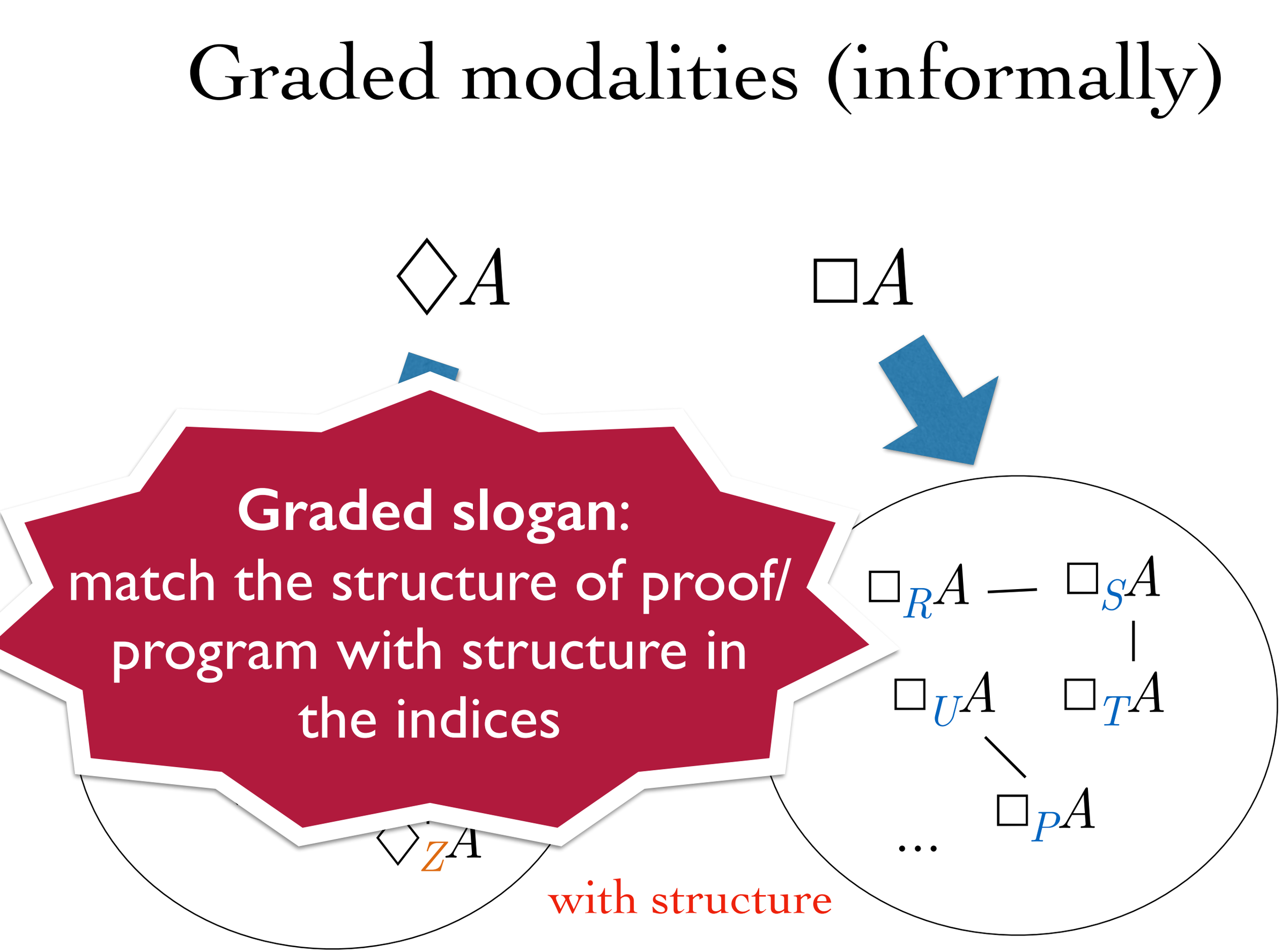
Granule demo 5

Bounded sessions

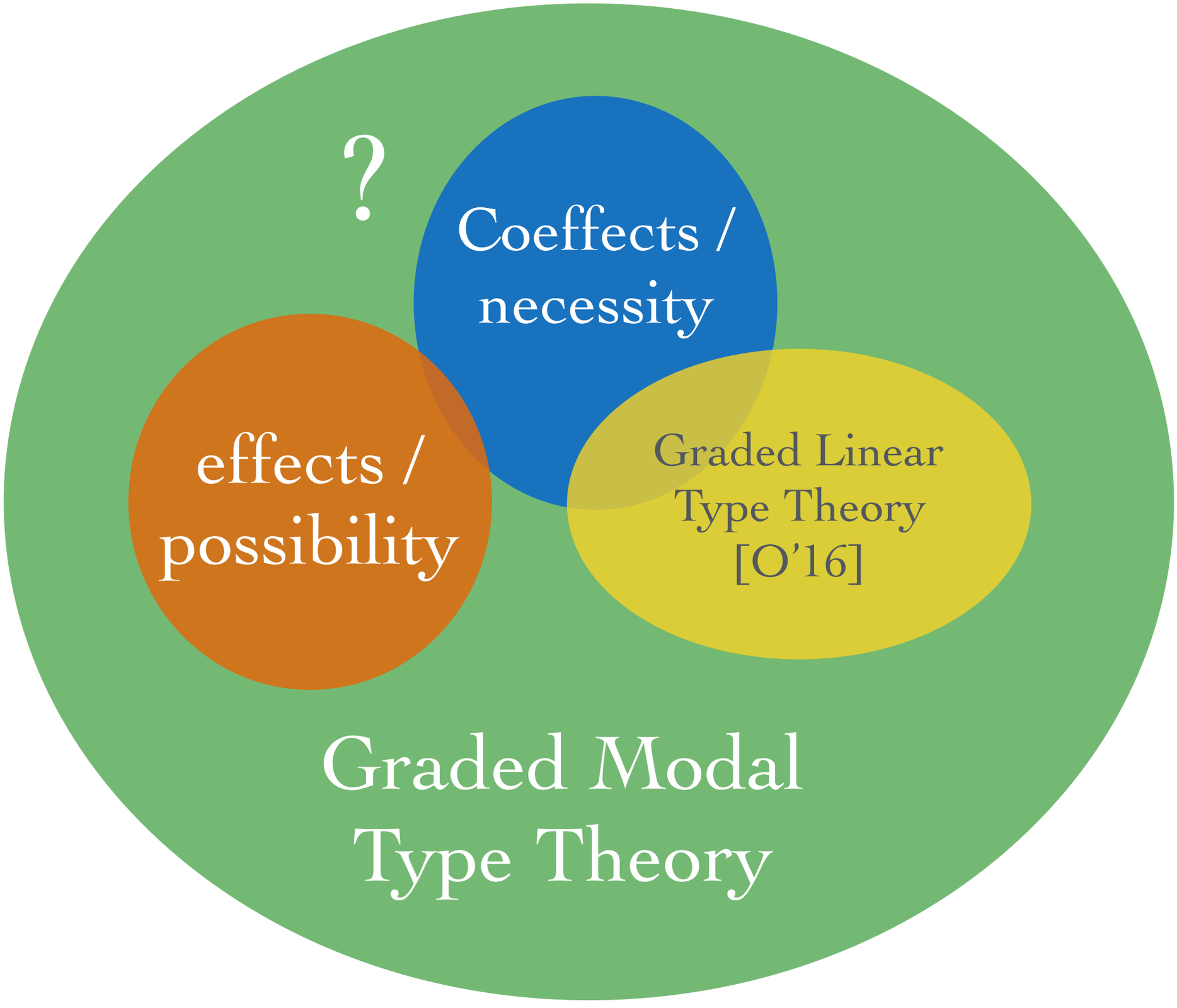
Graded modalities (informally)

$\Diamond A$ $\Box A$

Graded slogan:
match the structure of proof/ program with structure in the indices



with structure


?
Coeffects / necessity
effects / possibility
Graded Linear Type Theory [O'16]
Graded Modal Type Theory

Modern programming challenges

- Manage sensitive private information
  - ✤ Security coeffects
- Obey stateful protocols
  - ✤ Graded by pairs of pre-post conditions (see parameterised monads)
  - ✤ Or, graded by "morphisms" giving proof between pre-and-post
- React to uncertainty
  - ✤ Capture exact requirements with modality

In progress

In progress

Future

+ concurrent & distributed

- ✤ Session types + fine grained resources + effects

What next?

- **Granule with <u>extensible</u> graded modalities**
  - **Pick-your-own-gradings, and combine**
  - **User-defined (within Granule)**
  - **May need extra solver support**

- Interacting possibility and necessity modalities:
  - Combining effects and coeffects via grading (Gaboardi et al. 2016)
  - Classical data flow analyses

- Full-dependent types

- Decompose and re-express existing work

Here is what I consider one of the biggest mistakes of all in modal logic: concentration on a system with just one modal operator

Dana Scott (1968)

Download me and play!
http://github.com/dorchard/granule

Use graded modalities in *your* research!

Graded possibility:

$$\Diamond_x \Diamond_y A \to \Diamond_{x \oplus y} A$$

$$A \to \Diamond_I A$$

$$(A \to B) \to \Diamond_x A \to \Diamond_x B$$

Graded necessity:

$$\Box_{r * s} A \to \Box_r \Box_s A$$

$$\Box_1 A \to A$$

$$(A \to B) \to \Box_r A \to \Box_r B$$

Thanks!

Backup slides

Other applications / GMTTs

- Contextual Model Type Theory (Nanevski et al. '08)

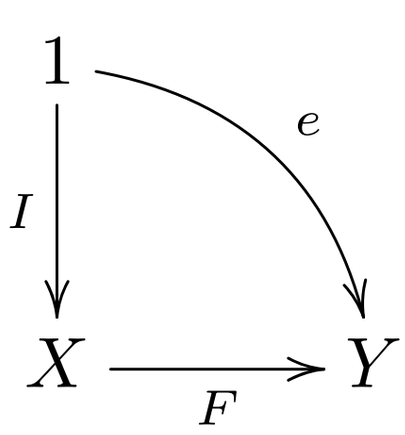
  $\Box_\Gamma A$ — A is true under closure of Γ

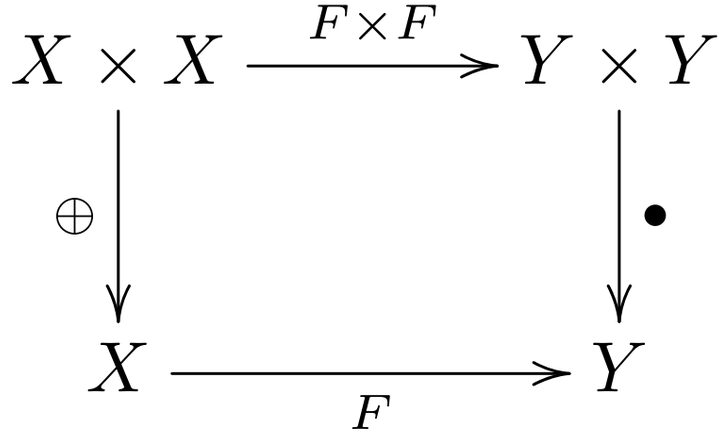
- Resources for dependent lollipop (McBride '16)
- Hardware schedules (Ghica et al. '14)
- Explicit provability logics (Artemov '95, '01)

- Multi-stage programming (generalising Pfenning & Davies)
- Costs (cf. Cicek et al. 17)
- Robustness / sensitivity (Gaboardi et al., Pierce et al.)
- Provenance
- Probabilistic programming (forwards / backwards)
- Type state (stateful protocols)

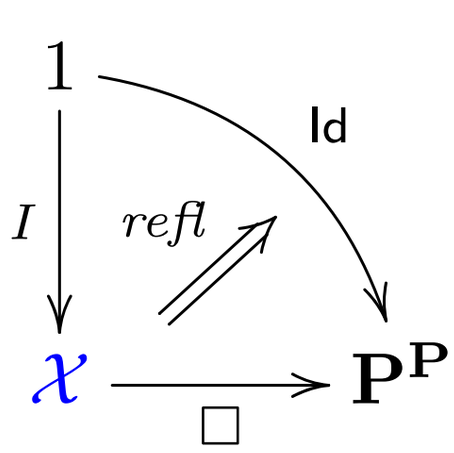
The Essence of Graded Modality

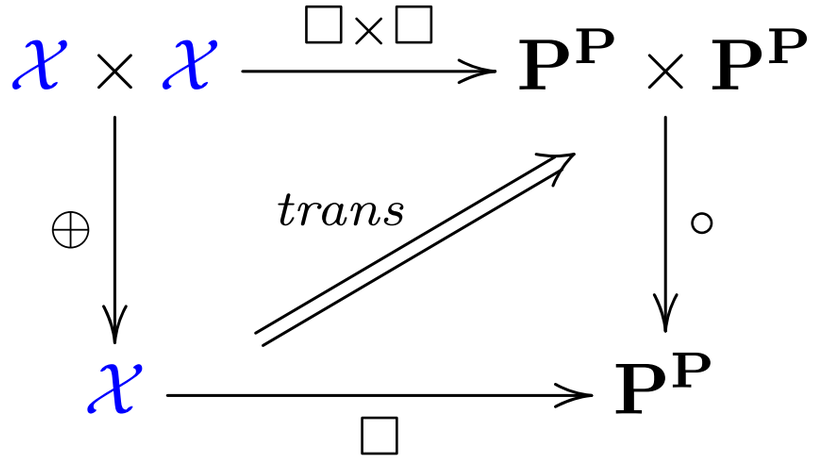
Recall, monoid homomorphism $(X, \oplus, I) \xrightarrow{F} (Y, \bullet, e)$





Graded necessity, with $(\mathcal{X}, \oplus, I)$ is a **lax** monoid homomorphism



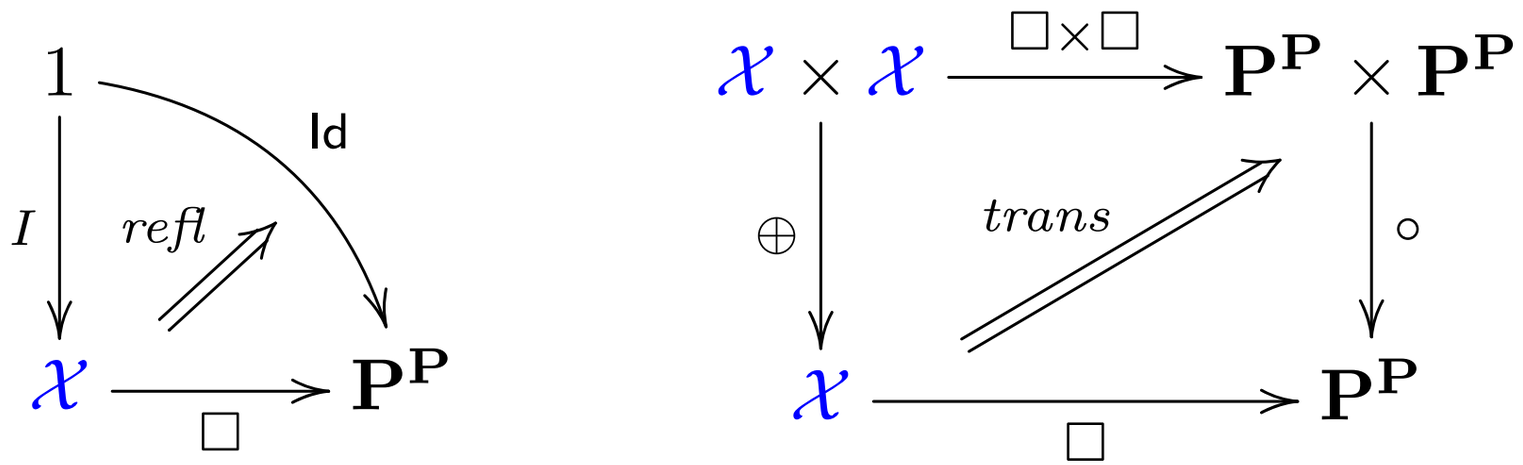


$\Box_I A \xrightarrow{refl} A$

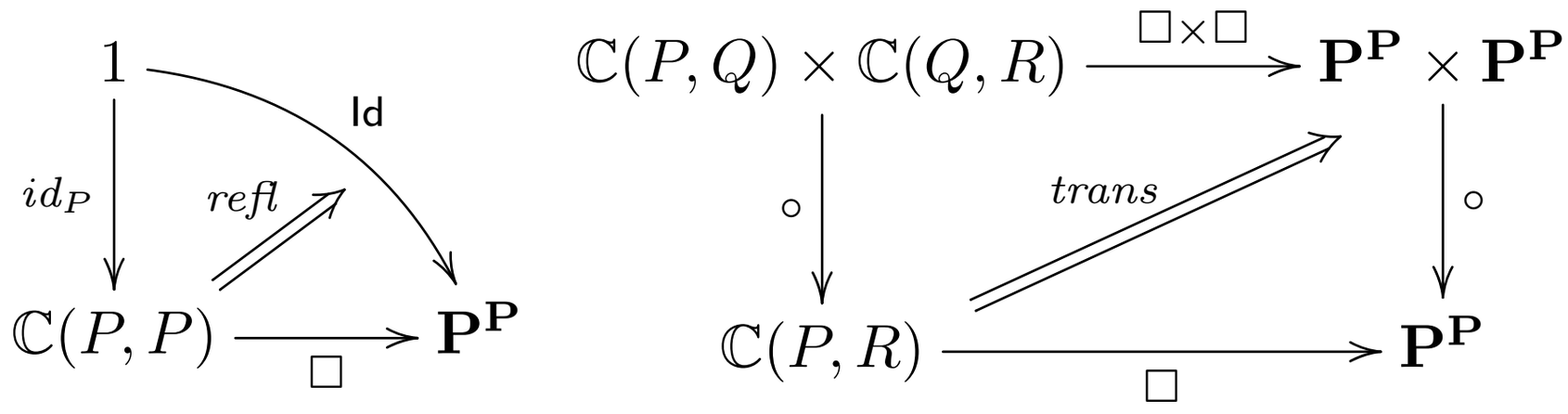
$\Box_{f \oplus g} A \xrightarrow{trans} \Box_g \Box_f A$

The Essence of Graded Modality

Graded necessity, with $(\mathcal{X}, \oplus, I)$ is a **lax** monoid homomorphism



General graded modality is a **lax** functor (category homomorphism)



$\Box_{id_P} A \to A$

$\Box_{g \circ f} A \to \Box_g \Box_f A$