Linear Exponentials as Graded Modal Types
Linear Logic

! A modality represents non-linear usage of A

Bounded Linear Logic

!_r A family of modalities where r gives an upper bound on usage

generalises to...

Graded Modal Types

□_r A family of modalities where r is drawn from a pre-ordered semiring

(\mathcal{R}, *, 1, +, 0, \sqsubseteq)
Granule

- Linear Types (data as a resource)
- Graded Modal Types (quantitative reasoning)
- Indexed Types (precision)
Demonstration
The Problem

In Granule... (and Linear Haskell, and Idris 2...)

\[
push : \forall \{ a, b : \text{Type}, s : \text{Semiring}, r : s \} . (a, b) [r] \to (a [r], b [r])
push [(x, y)] = ([x], [y])
\]

but in linear logic...

\[
push : ! (A \otimes B) \to ! A \otimes ! B
\]

is not derivable!
The Solution

No semiring that can represent the ! modality...

...so we must augment our semiring with an additional operation.

\[ \exists : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R} \]

(pronounced hsup)

Not too difficult to apply to other graded type systems!
The \( \{0, 1, \omega\} \) Semiring

- 0 must be discarded
- 1 must be used linearly
- \( \omega \) permits unconstrained use

We want to represent \(!A\) as \(\Box_{\omega} A\).

In Granule:

- Zero
- One
- Many
Pattern Matching

\[ ?r \vdash p : A \triangleright \Delta \]

before...

\[
\begin{align*}
\frac{r \vdash p_1 : A \triangleright \Gamma_1 \quad s \vdash p_2 : B \triangleright \Gamma_2}{r \uplus s \vdash (p_1,p_2) : A \otimes B \triangleright \Gamma_1, \Gamma_2} \quad \text{[PPROD]} \\
\frac{r \times s \vdash (p_1,p_2) : A \otimes B \triangleright \Gamma_1, \Gamma_2}{r \vdash p_1 : A \triangleright \Gamma_1 \quad s \vdash p_2 : B \triangleright \Gamma_2} \quad \text{[PPROD]}
\end{align*}
\]

...after

In semirings where we are okay with allowing push, just set \( \times = \uplus \)!
The $\triangleleft$ Operation

$$r \triangleleft s = \begin{cases} 1 & r = 1 \land s = 1 \\ \bot & \text{otherwise} \end{cases}$$

okay to push if both grades are 1!

$$(A \otimes B) \rightarrow A \otimes B$$

otherwise, pushing is forbidden.

$$\text{push}_! : ! (A \otimes B) \rightarrow ! A \otimes ! B$$
Representing!

We want to represent $!A$ as $\square_\omega A$.

So in Granule...

\[
\text{push : forall } \{a \ b : \text{Type}\}. \ (a, \ b) \ [\text{Many}] \rightarrow (a \ [\text{Many}], \ b \ [\text{Many}])
\]
\[
\text{push } [(x, \ y)] = ([x], \ [y])
\]

...should be disallowed.
Demonstration

The main theorem:
The adjusted Granule core calculus, for the \( \{0, 1, \omega\} \) semiring with \(!A = \Box_\omega A\), has the same expressive power as IMELL.
References

For more on Granule and graded types...

*Quantitative program reasoning with graded modal types*
Dominic Orchard, Vilem Benjamin-Liepelt, Harley Eades III (ICFP 2019)

For more on distributive laws...

*Deriving distributive laws for graded linear types*
Jack Hughes, Michael Vollmer, Dominic Orchard (LINEARITY/TLLA 2020)

And read the rest of this paper for more details!

*Linear exponentials as graded modal types*
Jack Hughes, Daniel Marshall, James Wood, Dominic Orchard (TLLA 2021)
Thank you!

Publications, implementation and more at: granule-project.github.io

Find us on Twitter at: @granulelang

Find me on Twitter at: @starsandspirals ⭐🌀