Thesis proposal: Limits of graded modalities

Unité de recherche. LIPN

Discipline. Informatique

Directrice de thèse. Micaela Mayero

Co-encadrant de thèse. Flavien Breuvart

Contact. `mayero@lipn.univ-paris13.fr`, `breuvart@lipn.univ-paris13.fr`

Domaine de recherche. Théorie de la démonstration

Mots clés. Analyse statique, types d'ordre supérieurs, dépendance, resources quantitatives, sémentique dénotationelle, réalisabilité, théorie des catégorie, inférence de types, programation fonctionelle, logique.

Candidat. Thomas Laure

Context: Graded modalities

Type systems are generally constraining a programming language so that meaningless programs are not allowed by the compiler. But the notion can be widened to include computation of structural invariants, such as the size of the expected value, the efficiency of a program or the assertion of complex propositions. This thesis explore, from a theoretical perspective, the possible ways to recover these invariants and the limits of such an approach.

Graded modal types [12, 18] (GM) are type systems enriched with additional annotated information insides types (and restricted to some modalities). Those annotations are able to accumulate information, thus computing invariants that are more refined and more extensional than usual analyses via type systems; that are only able to capture intentional structure of the program. GM are especially interesting for the diversity of the analyses they allow via a modularity with respect to an algebraic structure, while preserving some degree of automation (type inference).

GM have gained some popularity in the last decade [3, 2, 18, 8, 16, 10, 14, 5, 1] due to an approachable presentation backed by a strong mathematical concept. The idea is that modalities can be graded by algebraic structures, the different modal operations act on. In practice, this means that one can perform static analysis by simply refining types with modalities that are annotated with algebraic information; inferring those types and recovering the most general (or just a good enough) information can give important insights on the terms akin to those from abstract interpretation analysis.

Among those modalities, literature has been mostly focused on grading (strong) monads [19, 17, 15], exponentials (from linear logic) [7, 13] and their interactions [12]. A graded monad is the choice of an ordered monoid $(\mathbb{M}, 1, \cdot, \leq)$ and, for each $e \in \mathbb{M}$ and each type A, a type $M_e A$ of types "which effect or result verify the property e", and graded unit and bind

$$\eta : A \to M_1 A \qquad \texttt{bind} : M_e A \times (A \to M_f B) \to M_{e \cdot f} B\ .$$

Where, by (side) effect, we mean anything that comes from the interaction of the calculation with the real world, from a system calls or non-deterministic choices to time consumption.

Of course, the sentence "which effect or result verify the property e" is very informal, and any formalisation would be tautological without considering specific cases such as sets of possible exceptions, or a bound to the number of sequential access to a in/output buffer, or even a bound on the probability of correction. The important point is that the type discipline forces the sequencialization of the information flow formed by the functional compositions, but, contrary to CPS transformations, some operators may be non-sequencializable provided that they are mirrored in the monoid by an additional structure; for example, one can consider semirings/duoids to represent parallelisation, or Kleene algebra to interpret recursive functions.

A graded exponential is similar, but the unit η and bind are replaced by opposite linear logic operators $d :!_{1}A \to A$ and $\delta :!_{e \cdot f}A \to !_{e}!_{f}A$, together with monadic operators requiring a semiring structure on indexes, $w :!_{0}A \to \mathbb{1}$ and $c :!_{e+f}A \to !_{e}A \times !_{f}A$. The paradigmatic graded exponential is the one graded by $(\mathbb{N}, 1, \cdot, 0, +, \leq)$ which represents bounds on resource usage in the sense that a program typed by $!_{3}A \to B$ is a program using its argument at most three times; a similar example is that of $(\texttt{bool}, \mathtt{tt}, \wedge, \mathtt{ff}, \vee, \Rightarrow)$ which is behind the implementation of linear Haskell [3].[1] In general, graded exponential are characterising "coeffects" rather than effects, which are usage constraints on resources.

These modalities are thus presented as super-structures on a preexisting programming language, in which we have refined control. But denotational semantics told us that any programming language always performs into an ambient monad (which allows effects such as printing, communicating or even just non-termination) and an ambient exponential (which allows duplication and erasure).[2] Thus one can also grade those ambient modalities which corresponds to performing a static analysis on the given program. Surprisingly, one can also see inferring graded monads as performing some forward analysis, while inferring comonad is performing some backward analysis.

Principle: Successes and Shortcomings of graded modalities

GM have been quite successful in attracting both theoretic and practical application. With, notably, a small language developed by the Granule project [18]. Nonetheless, most concrete applications, such as the DFuzz language [11] had to drop the formalism at some point, going for standard dependently typed program without type-inference via Rocq (previously Coq), Idris, or Agda. This indicates that the approach has limitations.

Even though those many of these limitations have been intuitively known, they were never formalised nor proven, let alone exhaustively studied. **The long term objective of the thesis** will be to identify them precisely and to form a reflection on how to go beyond those with limited cost.

Axis 1. Among those limitations, the applicant will first focus on one that is well known from linear logician: the fact that the graded exponentials, much like their non-graded counterpart, are inherently commutative. In terms of gradations, this implies that the ordered semiring has a commutative addition. For static analysis, this is quite a drawback, as this means that there is no way to know in which order event will happen. This specific problem will be the starting point of the thesis.

Axis 2. Another shortcoming is the isomorphism of resources. In Granule, and in some other papers, types can be quantified over gradations. However, no analysis of the expressive power of such extension has been made. The meaning of a quantifier over gradation is, in fact, a concept that has a multitude of variations, with vastly different expressivity. To understand this, one have to see that, looking for the most general type mean looking for a unification algorithm, which, itself, means resolving systems of equations (or constraint entailment problems), but solving such system in a given algebra $\mathbb{M}$ can be extremely complex, or even undecidable, even when testing an equality is. However, solving such system in the theory of monoid is easy, and remains easy even if you add constants which multiplication you can compute, this corresponds to solving the equation in the free monoidal extension of $\mathbb{M} \uplus \texttt{Var}$. Granule is performing the last, which is absolutely reasonable, but can be refine by considering more

[1] We are simplifying a bit here, as they use a 3-valued semiring.

[2] We are, here, refering to Mogi "Evaluation" monad and the "linear" translation of LJ.

refine theories such as preordered monoids, semirings, lattice semirings, quantals etc, each with a different expressive power but also each with a different complexity of inference. Granule also use the well established let-polymorphism, but it may not be the only decidable and interesting fragment of the (probably undecidable) free-polymorphism; this part will be discussed next section.

Axis 3. Depending on how the thesis goes, other issues regarding GM could be looked at. One of those is the relation with the calling strategy: while exponentials and monads are known to fit well with call-by-name and call-by-value respectively, via a (co-)Kleisli construction, they are not so pertinent in the other strategy.[3] Their graded versions are used in both cases, though, with some deviations from the choosing strategy; such deviations are reasonable when the modalities are explicited, but not if we plan for an external analysis. Thus, a proper understanding of the link between the call strategy and those GM is required, which may be approached via an encoding of those modalities in a call-by-push-value language or a bang-calculus.

Methodology: Curry-Howard-Lambek

The applicant will approach these issues using four different perspectives:

Type theory. Much like ML-like languages, such as Ocaml or Haskell, can be used to encode many examples of monads, dependently typed ones, such as Rocq, Agda or Idris, can be use to encode variety of graded monads. They are not just a source of examples, but a perfect place to study the limit of what can be done since proofs can be internalised and since any extension written there will be sound by construction. Finally, the foundation of Martin-Löf type-theory will be of great help to consider foundations of possible extensions of graded modalities (see next section).

Category theory. The separation between the syntactic nature of types from the semantic nature of gradation is easier to understand at categorical level. In addition, category theory can be use to study, not only each instance of GM, but the whole world formed by these modalities; microcosm principle, that states that, in category theory, the whole picture tends to have a similar structure as the instances, can then be used to study the limits of our model. On this last point, one of the advisor managed to show with coauthors [6] that[4] a monad has a "most general gradation" which is basically given by the possible gradations themselves; imposing further constrains, this allows, for example, to show that "meaningful" grading of a list-monad are only the sub-monoid over the power-set of natural numbers (representing information on the size of the list only).

Proof theory. The traditional approach to defining a new type system is to do it using proof theory, using set theory and rewriting theory to define the type system as a sequent calculus with cut-elimination. Proof theory offers a variety of tools that we will be able to use for our objective, in particular realisability (and more generally logical relations) can be use to understand invariants, while encoding such as CPS, CbPV, or dialectica transformations are wonderful to transform a problem into another, widening the horizons, and, obviously, type inference methods such as focusing are crucial.

Denotational semantics. Denotational semantics do allow simpler representations of programs, in which the gradation can have very natural interpretation. Many proofs can be performed there and, even more so, it is a representation which is much more intuitive. Game semantics, in particular, is a much needed tool to approach the notion of scheduled types presented in the next section.

[3] Eilenberg-Moore construction is not accessible at type level and the double Kleisli construction (which consists in taking the Kleisli $\mathbb{C}_T$ over the monad T and then the co-Kleisli $(\mathbb{C}_T)_{T'}$ over the co-monad T' mirroring T in $\mathbb{T}_T$) is not as efficient.

[4] up-to a size-jump issue.

Entry point: Schedule types

Surprisingly, these three axes are related by a common question, that of the study of Scheduled types, which happen to extend slightly the usual notions of graded exponential and graded monad, while sublimating each of the questions above. Schedule types form a novel notion that have not been studied yet, one of the advisor merely scratch the idea years ago but the notion was never properly formalised. The results obtained never took the form of a paper, but there is enough material. Verifying, formalising them and helping in the redaction of the this preliminary paper will naturally be the internship subject of the candidate, and an introduction to the schedule-type thematic, itself an entry-point to tackle our three interrogations.

A Schedule type system is a system in which the type of a function with two arguments has the form

$$l_1 \cdot A_1 \to I \bullet (l_2 \cdot A_2 \to J \bullet B)$$

where A_1, A_2 and B are scheduled types, where l_1 and l_2 are labels, and where I and J is an abstract object called *schedule* that "represents" the extensional behaviour of the execution of the programme once given its arguments, which we represent as regular expressions over the available alphabet ($I \in \texttt{Regexp}\{l_1, e\}$ and $J \in \texttt{Regexp}\{l_1, l_2, e\}$) where e represents an effect, in this case, if $J = l_2((e + l_2)l_1)^*$ this means that the second argument is always called at the beginning, then there is an alternation of events with the first being either an effect or a call over the second argument, and where the second is a call over the first argument. We are also interested in varying over the definition of schedules, which could be more complex (or simpler), but have to respect a certain algebraic structure.

Such type systems are endowed with a complex notion of scope and composition, especially if we consider subtyping and variance (with labels restricted to positive occurrences). The associated proofs of subject reduction and realizability semantics are then extremely entangled, especially if we want to consider the coherence of additional equalities such as

$$I \bullet (l \cdot A \to JK \bullet B) \simeq IJ \bullet (l \cdot A \to K \bullet B) \tag{1}$$

whenever l do not appear in J. These scope issues are also crucial for foundations of type theory, thus using and studying a dependently typed proof assistant (such as Rocq) is expected to be a rewarding approach.

In fact, an arrow $l \cdot A \to J \bullet B$ could consider as a type $\forall l. M_l(A) \to M_J(B)$ in Granule, i.e., as an an arrow type in a graded monadic call-by-name encoding where the left gradation is always universally quantified. Restricting Granule to such a fragment seem to be particularly interesting since it allows to **bypass the limitations of Axis 2** while preserving soundness. Indeed, the sum and Kleene star do not seem to be representable in Granule by default, because it implies equalities which are not "structural" on the resource variables, such as $(le)^* = \epsilon + l(el)^*e$, but their inference should not be more difficult. In addition, considering such arrows only constitute a constraint on the polymorphism that is vastly different to that of let polymorphism, but still interesting and probably inferenceable.

In the proposed scheduled types, the constant "e", that represent an arbitrary effect, has an unusual status as **explained in Axis 3**. Indeed, effects are rarely captured in call-by-name setting as their sheer meaning is not completely clear (even more so when considering that implementations are generally using call-by-need). For example, a program of type $e \bullet (l \cdot A \to e' \bullet B)$ will yield effect e once it is "used" and effect e' once it is applied and its result is used, but in CbN, one cannot happen without the other since the application is only performed whenever the final result us put to use. That is why Equation (1) seems to be reasonable to ask in call-by-name.

Equation (1) is also of importance for **the study of Axis 1**. Indeed, if we change the notion of schedules, from regexps over labels and effects, toward that of finite multiset of labels (without effects), then this equation not only seems to be sound, but the obtained type system seem to be equivalent to that of the call-by-name encoding of a system with an exponential graded by natural numbers, with the following encoding

$$[\![!_n A \to B]\!] = l \cdot A \to [l^n] \bullet B$$

However, is the same equation sound for the regexp case, which multiplication is not commutative ? First investigations seem to show that it is, but the proof is non-trivial, and the relation with the graded exponents is even less.

Work Context

Hosting team. The **LoCal** team (from the split of LoVe team) is especially dynamic, with 4 CNRS researchers and 11 faculty members, as shown by the regular seminar that has been held weekly for more than a decade without more than a month of discontinuity (summer Holiday excluded). The general interest for the present subject is also shared by other members of LoCal team, both permanent [7, 5] and PhD [5], thus insuring the inclusion of the candidate.

Supervision. It will be primary carried out by Flavien Breuvart, who have already supervised a PhD student (funded by an ANRJCJC) who defended in 2024 and has no current student. However, Micaela Mayero, HDR for several years ago, who will be the official supervisor, will not be a simple figurehead, but will provide a high level supervising as well as her expertise in dependant types, formalisations in Rocq(for example [9, 4]) and lead of projects (CerPAN and MILC.

Previous internships. The candidate has already spend a 4-month internship working with F.Breuvart on a completely different subject: the objective was to shade a new light on categorical semantics of separation logic. Another 5-month internship, one Scheduled types this time, will start in March.

International collaborations. Scheduled types resulted from an old collaboration between Flavien Breuvart and Dan Ghica that has been stopped due to the pandemic as well as other responsibilities of both parties. They have reconnected recently and are willing to continue working on this together, with old unpublished results that can be reused to constitute a first article for the applicant in short timing. Another possible international collaboration would be with Tarmo Uustalu, in Reykjavik, with whom F.Breuvart has a current collaboration on related subjects [6] and who is also willing to invite the applicant.

[1] Ignacio Bellas Acosta and Yde Venema. "Counting to infinity: Graded Modal Logic with an infinity Diamond". In: *Rev. Symb. Log.* (2024).

[2] Robert Atkey. "Syntax and Semantics of Quantitative Type Theory". In: *LICS*. 2018.

[3] Jean-Philippe Bernardy et al. "Linear Haskell: practical linearity in a higher-order polymorphic language". In: POPL (2018).

[4] Sylvie Boldo et al. "A Coq Formalization of Lebesgue Integration of Nonnegative Functions". In: *J. Autom. Reasoning* (2022).

[5] Flavien Breuvart, Marie Kerjean, and Simon Mirwasser. "Unifying Graded Linear Logic and Differential Operators". In: *FSCD*. 2023.

[6] Flavien Breuvart, Dylan McDermott, and Tarmo Uustalu. "Canonical Gradings of Monads". In: *ACT*. 2022.

[7] Aloïs Brunel et al. "A Core Quantitative Coeffect Calculus". In: *ESOP, Part of ETAPS*. 2014.

[8] Pritam Choudhury et al. "A graded dependent type system with a usage-aware semantics". In: POPL (2021).

[9] Hugo Férée et al. "Formal proof of polynomial-time complexity with quasi-interpretations". In: *CPP*. 2018.

[10] Chase Ford. "Presentations of Graded Coalgebraic Semantics". PhD thesis. University of Erlangen-Nuremberg, 2023.

[11] Marco Gaboardi et al. "Linear dependent types for differential privacy". In: *POPL*. 2013.

[12] Marco Gaboardi et al. "Combining effects and coeffects via grading". In: *ICFP*. 2016.

[13] Dan R. Ghica and Alex I. Smith. "Bounded Linear Types in a Resource Semiring". In: *ESOP, Part of ETAPS*. 2014.

[14] Peter Hanukaev and Harley Eades III. "Combining Dependency, Grades, and Adjoint Logic". In: *TyDe*. 2023.

[15] Shin-ya Katsumata. "Parametric effect monads and semantics of effect systems". In: *POPL*. 2014.

[16] Shin-ya Katsumata et al. "Flexible presentations of graded monads". In: *ACM* ICFP (2022).

[17] Paul-André Melliès. "The parametric continuation monad". In: *MSCS* (2017).

[18] Dominic Orchard, Vilem-Benjamin Liepelt, and Harley Eades III. "Quantitative program reasoning with graded modal types". In: ICFP (2019).

[19] Alexander Smirnov. "Graded Monads and Rings of Polynomials". In: *J. Math. Sci.* 151 (2008), pp. 3032–3051.