GEDISTRIBUEERDE EN VEILIGE SOFTWARE (DISTRINET)
CELESTIJNENLAAN 200A, BOX 2402
3001 LEUVEN, BELGIUM

PhD/Researcher position:
Implementation of Multimodal Type Theory for Agda

Are you experienced or deeply interested in topics such as functional programming, machine-checked formal proving, programming language design and implementation, category theory and logic? We are looking for a PhD researcher to work on the implementation of modal type theory in Agda, in order to make the recent theoretical advancements in this topic accessible to a wider audience, thus facilitating adoption, application and further development of modal language features in functional programming languages and proof assistants. In this role, you will work with us, the Agda development community and our international research connections to structure, unify, justify and generalize current ad hoc implementations of modal features for Agda based on general theoretical foundations. In the process, we will encounter both foreseeable and unforeseen gaps between theory and practice that require more analytical research to be bridged.

Research Context

Dependent Type Theories (DTTs) form the theoretical basis of proof assistants like Agda, Coq, Lean and Idris. Such systems automatically type-check proofs, assisting the user and increasing trustworthiness of proofs, both in mathematics and in the verification of software correctness and security properties. Modal type theories additionally support modal logics, e.g. logics for possibility and necessity [PD01], variance of functors [Abe08, LH11], type-checking time and runtime irrelevance/erasure [Pfe01, AS12, AVW17, ND18, ADE23], parametricity [NVD17, ND18], guarded recursion [Nak00, BGM17, MM18], and constructing internal universes for homotopy and directed type theory [LOPS18, ND23].

While historically, the metatheory of different modal type theories has typically been studied separately, together with our colleagues from Aarhus University, we have contributed Multimodal Type Theory (MTT) [GKNB21]. MTT is parametrized by a 2-category called the mode theory, that specifies the number of modalities and their interactions. MTT can be studied and implemented once and for all, under general assumptions about the mode theory, while specific modal type theories can be recovered by instantiating the mode theory. MTT has been well-received by the community, but adoption by both fellow type theorists and more applied researchers remains difficult by lack of good proof assistant support.

Agda is a proof assistant that is widely-used by both type-theorists and more applied researchers, as it strikes a good balance between maturity and adaptability. Its current code-base contains a number of parallel hardcoded mode theories for runtime erasure [ADE23], irrelevance and shape-irrelevance [AVW17], and the flat modality (♭) for 'external' quantification [LOPS18], and we have a pull request for functor variance [PECAN23] and a fork for parametricity [NVD17]. These are currently only partially supported by theoretical work. Moreover, the Agda development community is currently lacking a person who has (or will gain) the relevant theoretical and practical expertise and can dedicate the time to maintain these modal features and address the questions and issues surrounding them.

Central goals of the envisioned research are to unify these ad hoc implementations and to bring them up to date with today's *generic* state of the art, and to extend the theoretical knowledge on MTT to cover also the interaction of modalities with practical language features such as definitions, record types, inductive types, modules, type parameters and indices. This work will produce benefits on several fronts: (i) formalize the implemented modal type systems, rectify existing mistakes, and strengthen confidence in them and their interactions, (ii) harmonize and share the design and implementation of different modal systems, (iii) facilitate the dissemination of MTT by providing an interactive tool usable for instructive experimentation and (iv) facilitate further research – perhaps more complex or more application-oriented – on these and additional modal systems.

Requirements

- You have an academic Master's degree (or equivalent) relevant to research on the boundary of mathematical logic and theoretical computer science. If your prior education has been overwhelmingly on one side of the boundary, you can demonstrate interest in and skill with the other side.
- You are familiar with aspects of some of the following subjects, and have a strong willingness to learn more: functional programming (in particular Haskell, in which Agda is implemented), machine-checked formal proving and type theory (in particular Agda), dependently typed programming language design, implementation and semantics, category theory, and logic.
- You are a goal-oriented, analytical and creative thinker.

In addition, you have:

- Fluent English written and oral communication skills,
- A willingness to write about your work and to present your learnings to others,
- Openness to collaborate with our international co-workers and the Agda development community,
- The ability to independently plan, execute and report on your work on a regular basis.

Offer

We offer a full-time position as researcher in a PhD trajectory in computer science:

- You will join a small, supportive research team on type theory which has about a decade of expertise in modal type theory and longer in Agda implementation work,
- You will be part of the larger division of DistriNet, where you can never lose sight of the ultimate applicability of your research thanks to the strong and broad focus on cybersecurity, covering in particular subjects such as secure languages, secure compilation and software and hardware verification,
- There is project funding available for research visits, conference attendance and in order to attend Agda Implementors' Meetings,
- You will have the opportunity to publish and present your work at top international academic conferences,
- You can make impactful contributions to a leading, mature and flexible, free and open source proof assistant,
- You will work in a multicultural working environment at KU Leuven, in the vibrant city of Leuven. (More information on working conditions.)

KU LEUVEN

GEDISTRIBUEERDE EN VEILIGE SOFTWARE (DISTRINET)
CELESTIJNENLAAN 200A, BOX 2402
3001 LEUVEN, BELGIUM

Interested?

Send both of us your CV, short motivation letter, and Bachelor and Master diploma grade transcripts by e-mail, on **Nov 15 at the latest.**
In case of questions, you are welcome to mail us as well.

Andreas Nuyts
Dominique Devriese
firstname.lastname@kuleuven.be

References

[Abe08] Andreas Abel. Polarised subtyping for sized types. Mathematical Structures in Computer Science, 18(5):797–822, 2008. doi: 10.1017/S0960129508006853.
[ADE23] Andreas Abel, Nils Anders Danielsson, and Oskar Eriksson. A graded modal dependent type theory with a universe and erasure, formalized. Proc. ACM Program. Lang., 7(ICFP), aug 2023. doi:10.1145/3607862.
[AS12] Andreas Abel and Gabriel Scherer. On irrelevance and algorithmic equality in predicative type theory. lmcs, 8(1):1–36, 2012. TYPES'10 special issue. doi:http://dx.doi.org/10.2168/LMCS-8(1:29)2012.
[AVW17] Andreas Abel, Andrea Vezzosi, and Theo Winterhalter. Normalization by evaluation for sized dependent types. Proc. ACM Program. Lang., 1(ICFP):33:1–33:30, August 2017. URL: http://doi.acm.org/10.1145/3110277, doi:10.1145/3110277.
[BGM17] Patrick Bahr, Hans Bugge Grathwohl, and Rasmus Ejlers Møgelberg. The clocks are ticking: No more delays! In 2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), pages 1–12, 2017. doi:10.1109/LICS.2017.8005097.
[GKNB21] Daniel Gratzer, G. A. Kavvos, Andreas Nuyts, and Lars Birkedal. 2021. Multimodal Dependent Type Theory. Logical Methods in Computer Science Volume 17, Issue 3, (July 2021). DOI:https://doi.org/10.46298/lmcs-17(3:11)2021
[LH11] Daniel R. Licata and Robert Harper. 2-dimensional directed type theory. Electr. Notes Theor. Comput. Sci., 276:263–289, 2011. doi:10.1016/j.entcs.2011.09.026.
[LOPS18] Daniel R. Licata, Ian Orton, Andrew M. Pitts, and Bas Spitters. Internal universes in models of homotopy type theory. In FSCD, LIPIcs, pages 22:1–22:17, 2018. doi:10.4230/LIPIcs.FSCD.2018.22.
[MM18] Bassel Mannaa and Rasmus Ejlers Møgelberg. The Clocks They Are Adjunctions Denotational Semantics for Clocked Type Theory. In FSCD, volume 108 of LIPIcs, pages 23:1–23:17, 2018. doi:10.4230/LIPIcs.FSCD.2018.23.
[Nak00] Hiroshi Nakano. A modality for recursion. In 15th Annual IEEE Symposium on Logic in Computer Science, Santa Barbara, California, USA, June 26-29, 2000, pages 255–266. IEEE Computer Society, 2000. doi:10.1109/LICS.2000.855774.
[ND18] Andreas Nuyts and Dominique Devriese. Degrees of relatedness: A unified framework for parametricity, irrelevance, ad hoc polymorphism, intersections, unions and algebra in dependent type theory. In Logic in Computer Science (LICS) 2018, Oxford, UK, July 09-12, 2018, pages 779–788, 2018. doi:10.1145/3209108.3209119.
[ND23] Andreas Nuyts and Dominique Devriese. 2024. Transpension: The Right Adjoint to the Pi-type. Logical Methods in Computer Science Volume 20, Issue 2, (June 2024). DOI:https://doi.org/10.46298/lmcs-20(2:16)2024
[NVD17] Andreas Nuyts, Andrea Vezzosi, and Dominique Devriese. Parametric quantifiers for dependent type theory. PACMPL, 1(ICFP):32:1–32:29, 2017. URL: http://doi.acm.org/10.1145/3110276, doi:10.1145/3110276.
[PD01] Frank Pfenning and Rowan Davies. A judgmental reconstruction of modal logic. Math. Struct. Comput. Sci., 11(4):511–540, 2001. doi:10.1017/S0960129501003322.
[PECAN23] Josselin Poiret, Lucas Escot, Joris Ceulemans, Malin Altenmüller, and Andreas Nuyts. Read the mode and stay positive. In TYPES, 6 2023. URL: https://lirias.kuleuven.be/retrieve/720869.
[Pfe01] Frank Pfenning. Intensionality, extensionality, and proof irrelevance in modal type theory. In LICS '01, pages 221–230, 2001. doi:10.1109/LICS.2001.932499.
[SGB23] Philipp Stassen, Daniel Gratzer, and Lars Birkedal. {mitten}: A Flexible Multimodal Proof Assistant. In International Conference on Types for Proofs and Programs (TYPES 2022), volume 269 of LIPIcs, pages 6:1–6:23, 2023. doi:10.4230/LIPIcs.TYPES.2022.6.