Barbarossa's research state of art, September 2020

My research work started with Giulio (my supervisor), relating fundamental theorems about the *mathematical theory* of (untyped) λ-calculus with resource approximation and Taylor expansion. This led to our POPL 2020 paper "Taylor subsumes Scott, Berry, Kahn and Plotkin" (`https://lipn.univ-paris13.fr/~barbarossa/papers/BarbarossaM20.pdf`). If I were to summarize it by a slogan, I would say that *Taylor expansion provides the best theory of approximation for λ-calculus* (rather than Böhm trees). I want to stress the fact that λ-calculus is indeed no different from every other mathematical branch: the notion of "approximation" (whatever that means) plays a central role and allows to prove the crucial results of the discipline.
That paper raised a natural question: for what other interesting programming languages can one reproduce the work we did for λ-calculus? There are many natural candidates (probabilistic, algebraic, cbV, !-calculus etc), but maybe the most interesting one (at least from my point of view) is $\lambda\mu$-calculus: if λ-calculus is the (Turing-complete) programming language behind *intuitionistic* logic, $\lambda\mu$-calculus is the (Turing-complete) programming language behind *classical* logic. Now, my PhD is a "cotutelle" with the university of Roma Tre, under the supervision (in Rome) of Lorenzo Tortora de Falco. That's why last year I have spent a lot of time with Lorenzo (modulo lockdown). There, after trying to understand if I could transfer to other languages parts of the work done with Giulio (something which included looking all of them as well as their Taylor expansion, which took a long time!), I understood that $\lambda\mu$-calculus was the best suited. Now I have more or less 50 still-not-well-written pages, which I am looking forward to submit both as a LICS 2021 version, as well as a long journal one (but I still don't know which journal). For the seek of teasing, let me say that I should be able to provide a natural Taylor expansion for $\lambda\mu$-terms, show that it behaves well and finally that it allows to lift some advanced properties (e.g. Stability and Perpendicular lines Property, with non existence of parallel-or as a corollary). I believe this provides a starting point for the study of a *mathematical theory* of $\lambda\mu$-calculus (just like we have for λ-calculus) and raises a lot of other interesting questions (Krivine's realizability, Böhm trees, Saurin's $\Lambda\mu$-calculus, connectedness in proof-nets etc).

In parallel to that, there are some other interesting directions. The difficulty being that, as Tito says, rumours are that I have to write a thesis this year. Also, a lot of time is taken by my teachings this semester.

With Giulio, Flavien, Emma and maybe Thomas Ehrhard, we'd like to study hyperchoerences to approach the problem of observational equivalences in λ-calculus (on the style of Morris' ones). The idea is: how can we characterize - with some typing system probably coming from hyperchoerences - the equivalence (which is very natural) in λ-calculus obtained by $M \simeq N$ iff for all context $C(\!|\cdot|\!)$ one has that $\mathrm{nf}(C(\!|M|\!)) = S$ exactly when $\mathrm{nf}(C(\!|N|\!)) = S$?

Also, studying hyperchoerences I realized that what Ehrhard calls "positive hyporchoerences" is exactly (word by word) what in algebraic topology one calls "abstract simplicial complexes", which are the basic objects of (co-)homology theory. I find that being able to talk about the dimension, or the homology, of a proof/program is amazing and so I talked to Ehrhard, which told me the he had noticed the same too. If we have the occasion, we would like to investigate this direction which is so tempting.

Finally, as I had the occasion to mention to Tito and Boris, I have always been convinced about the deep relevance of the philosophical side of the transcendental syntax program proposed by Girard. I consider writing, in my thesis, some serious considerations on the subject.