Extra Credit - The Edge of Knowledge

David M Kahn

Summer 2022

This summer there will be an extra credit opportunity worth 0.5%. The basic idea is that you will read a paper on functional programming. This document details how the extra credit assignment works.

1 The Point

The point of this assignment is to give motivated students the opportunity to see beyond the course. Functional programming has so much more to it than could possibly fit into the course, and its body of knowledge is actively growing with new research. By participating in this assignment, you will see what the edge of functional programming knowledge looks like, and how people like you or me add to it. This isn't meant to be a hard assigment, but rather an opportunity for a new experience: a mental field trip to the edge of knowledge.

2 What To Do

To complete this assignment, here is what you need to do:

1. Select a paper to read in the realm of functional progamming, either from my suggestions or on your own.
2. Email me at davidkah@andrew.cmu.edu [1]with your paper choice. Please provide either a pdf or a link to the paper. I will either approve it or ask that you select something else. Complete this step by July 22 11:59pm.
3. Read the paper.
4. Write a document (the deliverable) on the paper and email it to me. Complete this step by August 5 11:59pm.

The assignment may be completed *once* throughout the summer – no double extra credit.

[1]Yes, my email leaves off the last letter of my last name.

3 The Deliverable

3.1 Basics

The document that you send to me is mainly to confirm that you read and thought about the paper. Nonetheless, I do expect to see two main components, which should be clearly identifiable in your submission:

- summary – Please include a short summary of the paper. This should briefly mention the methodology and novel contributions, and should include any key ideas and/or results.
- your thoughts – This can include criticisms of the paper's writing or content, new things you learned, things you couldn't understand after chewing them over, your experience reading the paper, comparisons with related material you know of, improvements you might suggest, things you just found neat, or anything else that comes to mind when digesting the paper. It does not need to include all of these points; only what is relevant to you.

With these two components, the deliverable is pretty close to a paper review that a peer reviewer would write when deciding whether or not a paper gets published. The only piece it's really missing is whether or not you think it should be published!

Like a paper review, I also expect your document to be short. A typical thorough review is usually no longer than a single page, and often closer to half a page. Thus, I reserve the right to stop reading after 2 pages[2]. You could probably get full credit with like 3 substantial paragraphs.

3.2 Grading

Grading will be binary – either I believe you really thought about the paper, or I do not. Whether you read through the paper should be apparent from your summary. Whether you put effort into reflecting on the paper should be apparent from the thoughts you write down. If I feel both these points are met, I will grant the 0.5%. And again, I reserve the right to only read up to the first 2 pages.

To reiterate, this is not meant to be hard. You should not need a lot of writing to prove that you read and reflected on the paper. You do not need to write down any proofs or deep technical explanations. I do not expect you to fully understand every last little detail. Reading the paper will likely take longer than writing the submission. The submission is merely to prove to me that you engaged with the work, like a slightly-more-involved way of taking attendance.

[2] at a normal formatting and font size, for the tricky among you

4 Paper Suggestions

Here are some paper suggestions by topic to help you out.

- resource analysis – What if types could also reason about work or other resources? [1]

  `https://www.cs.cmu.edu/~janh/assets/pdf/HoffmannH10.pdf`

- flow typing – What if we allowed casing on types (not just values), and used set-theoretic types? [2]

  `https://hal.archives-ouvertes.fr/hal-03426711/document`

- parallel programming – What if we really paid attention to how parallel functional programming uses memory? [3]

  `https://dl.acm.org/doi/pdf/10.1145/3371115`

- coinductive typing – What if values like looped and infinite lists really were allowed? [4]

  `https://discovery.ucl.ac.uk/id/eprint/1523431/1/Silva_CoCaml.pdf`

- dependent typing – What if we used types to encode a lot more about the allowed values (like an integer being positive), but did it in a lightweight way? [5]

  `https://link.springer.com/content/pdf/10.1007/978-3-642-35873-9_19.pdf`

If you want to look for other papers that interest you more, feel free to do so. The International Conference on Functional Programming (ICFP) is an easy place to find functional programming papers, as are more general programming language conferences:

- Symposium on Principles of Programming Languages (POPL)
- Conference on Programming Language Implementation and Design (PLDI)
- European Joint Conferences on Theory and Practice of Software (ETAPS)
- various others

Wherever you find what you want, just make sure to have the paper approved with me!

References

[1] J. Hoffmann and M. Hofmann, "Amortized resource analysis with polynomial potential," in *European Symposium on Programming*. Springer, 2010, pp. 287–306.

[2] G. Castagna, M. Laurent, K. Nguyen, and M. Lutze, "On type-cases, union elimination, and occurrence typing," *Proceedings of the ACM on Programming Languages*, vol. 6, no. POPL, p. 75, 2022.

[3] S. Westrick, R. Yadav, M. Fluet, and U. A. Acar, "Disentanglement in nested-parallel programs," *Proceedings of the ACM on Programming Languages*, vol. 4, no. POPL, pp. 1–32, 2019.

[4] J.-B. Jeannin, D. Kozen, and A. Silva, "Cocaml: Functional programming with regular coinductive types," *Fundamenta Informaticae*, vol. 150, no. 3-4, pp. 347–377, 2017.

[5] H. Zhu and S. Jagannathan, "Compositional and lightweight dependent type inference for ml," in *International Workshop on Verification, Model Checking, and Abstract Interpretation*. Springer, 2013, pp. 295–314.