Full Title[*]

Subtitle[†]

ANONYMOUS AUTHOR(S)

Write this last. State the problem. Say why it's an interesting problem. Say what your solution achieves. Say what follows from your solution.

Additional Key Words and Phrases: keyword1, keyword2, keyword3

1 INTRODUCTION

One page[1]. This is a template for your (mine?, first?) research paper. The benefits of this template are:

- It has a bulleted list of contributions. This is the list you have to rewrite first. It will drive the entire paper.
- The repository, this template is in, has a `Makefile`. Using `make` will save your from setting up *the* build system for your paper. So you can start writing immediately. You'll need to install `lhs2tex`[2] though.[3]
- We present few examples of `lhs2tex`, a tool to make your (not only Haskell) code pretty type set. It can also be (ab)used for typing more than code blocks (Section 3).
- The template is prefilled with *Seven simple, actionable suggestions* by Jones [2015], that will make your papers better (Appendix A).
- We also mentions other seven actionable principles by Dreyer [2016]. Again, to make your papers better (Appendix B).

2 MAIN IDEAS

2–3 pages.

Figure 1 looks impressive…but sends readers to sleep and/or makes them feel stupid. Explain it as if you were speaking to someone using a whiteboard. Conveying the intuition is primary, not secondary. Once your readers have the intuition, they can follow the details (but not vice versa).

3 THE DETAILS

More sections. 5 pages. This is where things like Figure 1 belong.

We use this section to demonstrate `lhs2tex`. You could use examples to introduce the problem, they are easy to write using `lhs2tex`.

```
zipWith :: (a → b → c) → [a] → [b] → [c]
zipWith k [ ]      [ ]      = [ ]
zipWith k (x : xs) (y : ys) = k x y : zipWith k xs ys
zipWith k _        _        = error "lengths don't match"
```

[*] Title note
[†] Subtitle note
[1] The page counts are for a denser two-column format. Scale appropriately
[2] http://hackage.haskell.org/package/lhs2tex
[3] The irony is that I setup this template, so I can avoid writing

2018. 2475-1421/2018/1-ART1 $15.00
https://doi.org/

$$\frac{}{x : A \vdash x : A}\ \textsc{Var}$$

$$\frac{\Gamma \vdash e : C}{\Gamma, x : A \vdash e : C}\ \textsc{Weaken} \qquad \frac{\Gamma, x_1 : A, x_2 : A \vdash e : C}{\Gamma, x : A \vdash e : C}\ \textsc{Contract} \qquad \frac{\Gamma, \Delta \vdash e : C}{\Delta, \Gamma \vdash e : C}\ \textsc{Exchange}$$

$$\frac{\Gamma \vdash f : A \rightarrow B \qquad \Delta \vdash x : A}{\Gamma, \Delta \vdash f\ x : B}\ \textsc{App} \qquad \frac{\Gamma, x : A \vdash e : B}{\Gamma \vdash \lambda x \rightarrow e : A \rightarrow B}\ \textsc{Abs}$$

Fig. 1. Simply typed lambda calculus with explicit structural rules

But even it isn't related, you can (ab)use `lhs2tex`, to write maths like it was Haskell. For example we can define a category to consist of following data [Awodey 2010] (your paper doesn't need to be about category theory):

- *Objects*: A, B, C
- *Arrows*: f, g, h
- For each arrow f there are given objects: $\mathit{dom}\ f$ and $\mathit{cod}\ f$ called the *domain* and *codomain* of f.
- ...

These data are required to satisfy following laws

- Associativity:
$$f \circ (g \circ h) = (f \circ g) \circ h \tag{1}$$
for all $f : A \rightarrow B$, $g : B \rightarrow C$, $h : C \rightarrow D$.
- Unit:
$$f \circ 1_A = f = 1_B \circ f \tag{2}$$
for all $f : A \rightarrow B$.

For further example, we can present associativity law as a diagram. The diagram below is made with a *MetaPost*[4]. You might consider using `diagrams` package[5]. Whatever tool you decide to use, reserve proper time to make your diagrams. "A picture is worth a thousand words" holds for information density for production time.

$$\begin{tikzcd}
A \arrow[dr, "f"] \arrow[rr, bend left, "g \circ f"] \arrow[drrr, bend left=40, "h \circ (g \circ f)"] \arrow[drrr, bend right=40, "(h \circ g) \circ f"'] & & C \arrow[dr, "h"] & \\
& B \arrow[ur, "g"] \arrow[rr, bend right, "h \circ g"'] & & D
\end{tikzcd}$$

Next we'll test that `cleveref` works: Lemma 3.1 and Example 3.2.

Lemma 3.1 (Yoneda). *Let* $\mathbf{C}$ *be locally small. For any object* $C \in \mathbf{C}$ *and functor* $F \in \mathbf{Sets}^{\mathbf{C}^{\mathrm{op}}}$ *there is an isomorphism*

$$\mathit{Hom}(yC, F) \cong FC$$

which, moreover, is natural in both F *and* C.

Proof. Omitted. □

[4] https://tug.org/metapost.html
[5] https://archives.haskell.org/projects.haskell.org/diagrams/

Example 3.2. We can use Lemma 3.1 to derive Profunctor Optics [Boisseau and Gibbons 2018].

4 RELATED WORK

1–2 pages. There are various resources.

- ACM SIGPLAN Author Information http://www.sigplan.org/Resources/Author/ has a short *Writing* section.
- Simon Peyton Jones has a longer list on https://www.microsoft.com/en-us/research/academic-program/write-great-research-paper/#!other-resources.

5 CONCLUSIONS AND FURTHER WORK

Half a page.

REFERENCES

Steve Awodey. 2010. *Category Theory* (2nd ed.). Oxford University Press, Inc., New York, NY, USA.

Guillaume Boisseau and Jeremy Gibbons. 2018. What You Needa Know about Yoneda: Profunctor Optics and the Yoneda Lemma (Functional Pearl). *Proc. ACM Program. Lang.* 1, ICFP, Article 84 (Aug. 2018), 27 pages.

Derek Dreyer. 2016. How to write papers so that people can read them. https://www.mpi-sws.org/~dreyer/talks/talk-plmw16.pdf. [Online; accessed 20-July-2018].

Simon Peyton Jones. 2015. How to write a great research paper. https://www.microsoft.com/en-us/research/academic-program/write-great-research-paper/, https://www.cis.upenn.edu/~sweirich/icfp-plmw15/slides/peyton-jones.pdf. [Online; accessed 20-July-2018].

A SIMON'S SUGGESTIONS

We used some of *Seven simple, actionable suggestions* by Jones [2015], in this template. The all suggestions are:

(1) **Don't wait: write**. Writing papers is a primary mechanism for doing research (not just for reporting it)
(2) **Identify your key idea**. Your goal s to convey a useful, re-usable, clear and sharp idea.
(3) **Tell a story**. Imagine you are explaining at a whiteboard.
(4) **Nail your contributions**. Do not leave the reader to guess what your contributions are!
(5) **Related work: later**. Problems with too early related work: The reader knows nothing about the problem yet; so your description of various technical tradeoffs is absolutely incomprehensible. Describing alternative approaches gets between the reader and your idea
(6) **Put your readers first (examples)**. Introduce the problem, and your idea, using *examples* and only then present the general case.
(7) **Listen to your readers**. Get your paper read by as many friendly guinea pigs as possible

There are various recodings of the presentation on YouTube, for example https://www.youtube.com/watch?v=WP-FkUaOcOM.

B DEREK'S PRINCIPLES

Dreyer [2016] gives seven conrete suggestions, which are different from Simon's.

(1) **Old to new**. Begin sentences with old info. End sentences with new info.
(2) **One paragraph, one point**. A paragraph should have one main point, expressed in a single *point sentence*.
(3) **Name your baby**. Give unique names to things and use them consistently.
(4) **Just in time**. Give information precisely when it is needed, not before

(5) **CGI model for abstract/intro**. *Context*: Set the stage, motivate the general topic. *Gap*: Explain your specifc problem and why existing work does not adequately solve it. *Innovation*: State what you've done that is new, and explain how it helps fill the gap.
(6) **Have a "main ideas" section**. Use *concrete illustrative examples* and high-level intuition. Do *not* have to show the general solution.
(7) **Compare with related work at the end**. It goes at the end of the paper. Give real comparisons, not a "laundry list"!

There is a recording of the talk on YouTube: https://www.youtube.com/watch?v=PM1Atui30qU.