Research Methodology in Networking

2021-2022

Class 3

Writing - Part 1/2

---

papers

---

papers

~

internship report, note to boss, project proposal, start-up fund raising doc, etc.

3 exercises during class

Writing papers is a skill

- Many papers are badly written

- Good writing is a skill you can learn

- It's a skill that is worth learning

  - You will get more brownie points (more papers accepted, etc.)

  - Your ideas will have more impact

  - You will have better ideas

Writing papers: model 1

Idea → Do research → Write paper

Writing papers: model 2

Idea → Do research → Write paper

Idea → Write paper → Do research
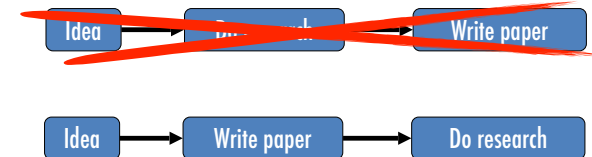
- Forces us to be clear, focused

- Crystallizes what we don't understand

- Opens the way to dialogue with others: reality check, critique, and collaboration

Do not be intimidated

FALLACY

You need to have a fantastic idea before you can write a paper. (Everyone else seems to.)

Write a paper, and give a talk, about **any idea**, no matter how weedy and insignificant it may seem to you

Do not be intimidated

Write a paper, and give a talk, about **any idea**, no matter how weedy and insignificant it may seem to you

- Writing the paper is how you develop the idea in the first place

- It usually turns out to be more interesting and challenging that it seemed at first

The purpose of your paper

Why bother?

FALLACY

We write papers and give talks mainly to impress others, gain recognition, and get promoted

Good papers and talks are a fundamental part of research excellence

Papers communicate ideas

- Your goal: to infect the mind of your reader with **your idea**, like a virus

- Papers are far more durable than programs (think Mozart)

> The greatest ideas are (literally) worthless if you keep them to yourself

13

The idea

> **Idea**
> A re-usable insight, useful to the reader

- Figure out what your idea is

- Make certain that the reader is in no doubt what the idea is. Be 100% explicit:

  - "The main idea of this paper is...."

  - "In this section we present the main contributions of the paper."

- Many papers contain good ideas, but do not distil what they are.

14

One ping

- Your paper should have just one "ping": one clear, sharp idea

- Read your paper again: can you hear the "ping"?

- You may not know exactly what the ping is when you start writing; but you must know when you finish

- If you have lots of ideas, write lots of papers
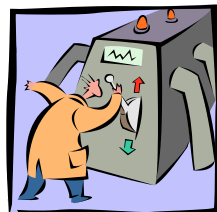
Thanks to Joe Touch for "one ping"

15

1

What is the main idea (ping) of you project?

16

The purpose of your paper is not…

… to describe the WizWoz system!

- Your reader does not have a WizWoz

- She is primarily interested in re-usable brain-stuff, not executable artefacts

17

Golden Rule: Storytelling

- Every paper tells a story
  - Not the chronology of your research

- What's the big deal? The main idea?
  - What is the problem? Why is it hard? Why is your solution interesting, significant? Why should the reader care?

- Note: Your story is not a mystery novel
  - Write top-down!

- Note: Nobody is as interested in this topic as you
  - Make it interesting!

18

Papers are not novels!

- Many aspects of writing style that are appropriate in literature are ineffective (or annoying!) in technical papers

- Tips for simplicity
  - One idea/topic per logical unit
  - Simple organization
  - Short words, simple structure

19

Economize

- "I am sorry I have had to write you such a long letter, but I did not have time to write you a short one" — Blaise Pascal

- The length of a paper should be correlated with its content

20

The structure of your paper

Your narrative flow

I wish I knew how to solve that!

I see how that works. Ingenious!

- Here is a problem

- It's an interesting problem

- It's an unsolved problem

- **Here is my idea**

- My idea works (details, data)

- Here's how my idea compares to other people's approaches

From: S. Peyton-Jones, "How to write a good research paper"
https://www.microsoft.com/en-us/research/academic-program/write-great-research-paper

---
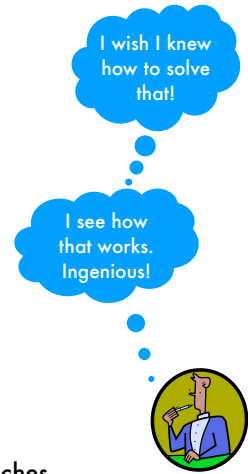
Structure (example for 12 pages)

- Title (1000 readers)

- Abstract (4 sentences, 100 readers)

- Introduction (1 page, 100 readers)

- The problem (1 page, 10 readers)

- My idea (2 pages, 10 readers)

- The details (5 pages, 3 readers)

- Related work (1-2 pages, 10 readers)

- Conclusion and further work (0.5 page)

From: S. Peyton-Jones, "How to write a good research paper"
https://www.microsoft.com/en-us/research/academic-program/write-great-research-paper

---

Structure

- Title

- **Abstract (a few sentences)**

- Introduction (1 page)

- The problem (1 page)

- My idea (2 pages)

- The details (5 pages)

- Related work (1-2 pages)

- Conclusion and further work (0.5 page)

From: S. Peyton-Jones, "How to write a good research paper"
https://www.microsoft.com/en-us/research/academic-program/write-great-research-paper

The abstract

- I usually write the abstract last

- Used by program committee members to decide which papers to read

- Four must-have parts [Kent Beck]

  - State the problem

  - Say why it's an interesting problem

  - Say what your solution achieves

  - Say what follows from your solution

---

nature

How to construct a *Nature* summary paragraph

Annotated example taken from *Nature* **435,** 114–118 (5 May 2005).

One or two sentences providing a **basic introduction** to the field, comprehensible to a scientist in any discipline.

Two to three sentences of **more detailed background**, comprehensible to scientists in related disciplines.

One sentence clearly stating the **general problem** being addressed by this particular study.

One sentence summarizing the main result (with the words "**here we show**" or their equivalent).

Two or three sentences explaining what the **main result** reveals in direct comparison to what was thought to be the case previously, or how the main result adds to previous knowledge.

One or two sentences to put the results into a more **general context**.

Two or three sentences to provide a **broader perspective**, readily comprehensible to a scientist in any discipline, may be included in the first paragraph if the editor considers that the accessibility of the paper is significantly enhanced by their inclusion. Under these circumstances, the length of the paragraph can be up to 300 words. (This example is 190 words without the final section, and 250 words with it).

During cell division, mitotic spindles are assembled by microtubule-based motor proteins[1,2]. The bipolar organization of spindles is essential for proper segregation of chromosomes, and requires plus-end-directed homotetrameric motor proteins of the widely conserved kinesin-5 (BimC) family[3]. Hypotheses for bipolar spindle formation include the 'push–pull mitotic muscle' model, in which kinesin-5 and opposing motor proteins act between overlapping microtubules[2,4,5]. However, the precise roles of kinesin-5 during this process are unknown. Here we show that the vertebrate kinesin-5 Eg5 drives the sliding of microtubules depending on their relative orientation. We found in controlled *in vitro* assays that Eg5 has the remarkable capability of simultaneously moving at ~20 nm s^{-1} towards the plus-ends of each of the two microtubules it crosslinks. For anti-parallel microtubules, this results in relative sliding at ~40 nm s^{-1}, comparable to spindle pole separation rates *in vivo*[6]. Furthermore, we found that Eg5 can tether microtubule plus-ends, suggesting an additional microtubule-binding mode for Eg5. Our results demonstrate how members of the kinesin-5 family are likely to function in mitosis, pushing apart interpolar microtubules as well as recruiting microtubules into bundles that are subsequently polarized by relative sliding. We anticipate our assay to be a starting point for more sophisticated *in vitro* models of mitotic spindles. For example, the individual and combined action of multiple mitotic motors could be tested, including minus-end-directed motors opposing Eg5 motility. Furthermore, Eg5 inhibition is a major target of anti-cancer drug development, and a well-defined and quantitative assay for motor function will be relevant for such developments.

2

Write an abstract for your project

Structure

- Title

- Abstract (a few sentences)

- **Introduction (1 page)**

- The problem (1 page)

- My idea (2 pages)

- The details (5 pages)

- Related work (1-2 pages)

- Conclusion and further work (0.5 page)

Introduction

- Summarizes the whole story

  - The most important part of the paper!

  - If people don't understand your problem, approach, importance by the end of the intro, you're out of luck

- Two schools of thought

  - Write it first: make certain the story is clear

  - Write it last: story becomes clear at the end

- Advice: do both

Why to start the intro early

- It's important to be able to concisely summarize your key contributions

  - In as little as a single paragraph

- If you cannot do this, it's quite possible that your thinking is not clear

  - Working on the story can improve your thinking

  - It may also become clear that you don't have a paper!

- **Muddled writing reflects muddled thinking**

The importance of first impressions

- Many readers make up their minds within the first few paragraphs

- The first few paragraphs should state the paper's purpose *with context*

  - Beware "This paper concerns"

- The beginning should be intelligible to any reader

Introduction: J. Kurose's Formula

- Paragraph 1: Context

- Paragraph 2: Problem area

- Paragraph 3: "This paper…"

- Paragraphs 4-5: Challenges / Solutions

- Paragraph 6: Summary of results

- Paragraph 7: Outline (we will discuss about this point later)

The introduction

- Describe the problem

- State your contributions

  - …and that is the core of the introduction

Describe the problem

1   Introduction

There are two basic ways to implement function application in a higher-order language, when the function is unknown: the *push/enter* model or the *eval/apply* model [11]. To illustrate the difference, consider the higher-order function `zipWith`, which zips together two lists, using a function **k** to combine corresponding list elements:

```
zipWith :: (a->b->c) -> [a] -> [b] -> [c]
zipWith k []     []     = []
zipWith k (x:xs) (y:ys) = k x y : zipWith xs ys
```

Here **k** is an *unknown function*, passed as an argument; global flow analysis aside, the compiler does not know what function **k** is bound to. How should the compiler deal with the call **k x y** in the body of `zipWith`? It can't blithely apply **k** to two arguments, because **k** might in reality take just one argument and compute for a while before returning a function that consumes the next argument; or **k** might take three arguments, so that the result of the `zipWith` is a list of functions.

> Use an example to introduce the problem

State your contributions

- Write the list of contributions first

- The list of contributions drives the entire paper: the paper substantiates the claims you have made

- Reader thinks "gosh, if they can really deliver this, that's be exciting; I'd better read on"

State your contributions

Which of the two is best in practice? The trouble is that the evaluation model has a pervasive effect on the implementation, so it is too much work to implement both and pick the best. Historically, compilers for strict languages (using call-by-value) have tended to use eval/apply, while those for lazy languages (using call-by-need) have often used push/enter, but this is 90% historical accident — either approach will work in both settings. In practice, implementors choose one of the two approaches based on a qualitative assessment of the trade-offs. In this paper we put the choice on a firmer basis:

- We explain precisely what the two models are, in a common notational framework (Section 4). Surprisingly, this has not been done before.
- The choice of evaluation model affects many other design choices in subtle but pervasive ways. We identify and discuss these effects in Sections 5 and 6, and contrast them in Section 7. There are lots of nitty-gritty details here, for which we make no apology — they were far from obvious to us, and articulating these details is one of our main contributions.

In terms of its impact on compiler and run-time system complexity, eval/apply seems decisively superior, principally because push/enter requires a stack like no other: stack-walking

Bulleted list of contributions

Do not leave the reader to guess what your contributions are!

Contributions should be refutable

NO!	YES!
We describe the WizWoz system. It is really cool.	We give the syntax and semantics of a language that supports concurrent processes (Section 3). Its innovative features are…
We study its properties	We prove that the type system is sound, and that type checking is decidable (Section 4)
We have used WizWoz in practice	We have built a GUI toolkit in WizWoz, and used it to implement a text editor (Section 5). The result is half the length of the Java version.

Think twice about "rest of this paper is…"

- Some authors suggest not to use:

"The rest of this paper is structured as follows. Section 2 introduces the problem. Section 3 … Finally, Section 8 concludes".

- Instead, one possibility is to use forward references from the narrative in the introduction.

- The introduction (including the contributions) should survey the whole paper, and therefore forward reference every important part.

3

Comment on a colleague's abstract

Homework 3

- Summarize the content of Class 3 (writing 1/2) in one PDF page

  - Produce a document that could be used by others as a reference (similar to what Jim Kurose did in his "formula" for writing good introductions

  - Make sure to list at least 5 important aspects of a good paper

- Use the template available on the METHOD webpage

- Deadline: October 13, 2021

Research Methodology in Networking

2021-2022