Notizen zu dem Video "PLMW ICFP21 - How to Write Papers So People Can Read Them".

- Curse of knowledge: When you know everything about a topic, it's hard to imagine, what it's like to know nothing.

Sentences & paragraphs (Low level structures)

In der Meinung vom Vortragshalter ist Flow und Coherence am **wichtigsten**.

Flow

- It should be clear, how each sentence and paragraph relates to the adjacent ones.
- Old to new::
  - Begin sentences with old info.
  - Creates link to earlier text.
  - End sentences with new info.
  - Create link to the text that follows.
  - Also places new info in position of emphasis.
  - Kannst Diagramme über neue/alte Infos machen, um dir zu verdeutlichen, wie, und ob, die Sätze miteinander verbinden.
- But flow is not enough! It still needs **Coherence**.

Coherence

- It should be clear, how each sentence and paragraph relates to the big picture.
- One paragraph, one point:
  - A paragraph should have one main point, expressed in a singel **point sentence**.
  - **Typically** the point sentence should appear **at or near the beginning of the paragraph**. (Especially for beginner.)
  - You can think of it, as the beginning of the paragraph, up to and including the point sentence, is kind of like the theorem of the paragraph. And the rest of the paragraph gives the proof for that theorem.
- The **point sentence** summarizes the paragraph. Or more accurat, summarizes the point, the paragraph makes.

- More of a conceptual problem. Harder to correct.
- Need to rewrite the whole paragraph most of the time.
- Beispiel, wie so etwas aussieht:

> **There appears to be a negative correlation between the charisma of a species and its ability to survive.** Lions and tigers, for instance, are among the most majestic creatures in the animal kingdom, yet they are currently facing extinction. In contrast, the house cat is evolutionarily quite successful, even though it is mostly known for stupid pet tricks.

Three small principles

- Name your baby: Give unique names to things and use them consistently.
  - Wenn du etwas mehrfach im Paper ansprichst, solltest du ihm auch einen einzigartigen Namen geben.
  - Sehr praktisch.
  - Zu viele Acronyme macht es verwirrend.
- Just in time: Give information precisely when it is needed, not before.
  - Leser können sich nicht an super viele Sachen erinnern.
  - Große Abschnitte voller Hintergrundinformationen werden schnell vergessen.
  - Also gibst du nötige Hintergrundinformationen erst, wenn du sie auch brauchst.
- Short subjects: Subject of sentence should be at most 8 words long.

Structure of a research paper (High level structures)

Overachring Principle #1: TOP-DOWN

- Explain your work at multiple levels of abstraction, starting at a high level (accessible to non-experts) and getting progressively more detailed.

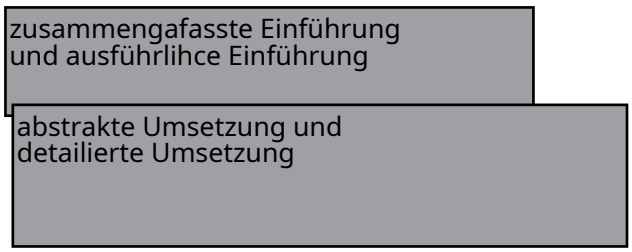
Overarching Principle #2: Tell them what they want to know

- How is your work important?
  - Why is this relevant to anybody?

    - What problem is this solving?
    - Why should I care?
- How is your work novel?
    - What is the new thing?
    - How is this different to prior work?
- How is your work interesting?
- ~~How was your work challenging?~~
    - Not important.
    - Some papers found answers, that weren't very complex at all. Complexity isn't the goal anyway.
- This questions should be answered as soon as possible.
- This structure doesn't always work, but if you can structure your paper in this way, it should help with your success.

A structure that works

- Abstract (1-2 paragraphs, 1000 readers)
- Intro (2-4 pages, 100 readers)
- Key ideas (4-6 pages, 50 readers)
- Technical meat (8-12 pages, 5 readers)
- Related works (1-3 pages, 100 readers)

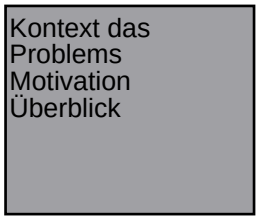


Abstract

- Like a condensed version of the Intro.
- Recommends the CGI model.
- CGI model:
    - Do one after the other.
    - Context: Set the stage, motivate the general topic. (Should take around a paragraph, or two.)
    - Gap: Explain your specific 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.

Intro

- Like an expanded version of the abstract.
- Alternative approach (SPJ): Same as CGI, but without Context.



    - Start with a concrete example, e.g. "Consider this Haskell code..."
    - If this works, it can be effective, but I find it often doesn't work.
    - It assumes reader already knows context.

Key ideas

- Bridge between Intro and the technical details.
- Use concrete illustrative examples and high-level intuition.
- You do **not** have to show the general solution (that's what the technical section is for).
- You want to give the high level intuition of your solution.
- Why have a "key ideas" section at all?
    1. Forces you to have a takeaway, i.e. something interesting.
    2. Many readers only care about the takeaway, not the technical details.
    3. For those who want the technical details, the key ideas are still useful as "scaffolding".

Related works

- It goes at the end of the paper. You can only properly compare to related work once you've explained your own.
- Give real comparisons, not a "laundry list"! Explain in detail how your work fills the Gap in a way that related work doesn't.
- You wanna be kind to related works. Don't be mean! Stay factual and don't overplay your own contribution.

Q&A am Ende

- Finding a good title is **very** important.
- Da Abstract die kleine Version von Intro ist, und sich die 'Story' des Papers meistens viel ändert, schreibt er das Abstract meistens erst am Ende. (Zu viel Arbeit, es sonst up to date zu halten.)
- Wenn er zu viele Seiten geschrieben hat, kürzt der Präsentator eher die technischen Details. Ihm erscheinen die Erklärungen von Ideen und Beispiele um einiges wichtiger.
- Im Generellen scheint es dem Präsentator wichtiger zu sein, Verständnis über die Ideen/Techniken hinter dem Paper zu schaffen. Genaue Implementationsdetails und Mathematik dahinter ist ihm weniger wichtig. (Lesen die meisten Leute eh nicht.)