Fordypning: teorimodul

TDT09

"System and runtime software interaction with modern hardware"

"System- og sanntidsprogramvareinteraksjon med moderne maskinvare"

2020-08-27

Michael Engel

Overview

- Organizational things
- Why are we actually doing this?
- Critical thinking
- Topic areas and overview
- The topics in detail
- Finally, some links

.org

- **When?**
  - Start working after distribution of topics
  - My proposal: weekly or bi-weekly (every 14 days) Zoom meetings to discuss your progress and talk about problems
- **Who?**
  - In general: you alone
  - …but exchange of ideas and discussions are welcome
- **What?**
  - Deliverables:
    - Talk (25 min.) + presentation slides (as PDF file)
    - Oral exam: Questions after your talk (another 15-20 min.)
    - ***No*** written report
- **How much work is it overall?**
  - …

Why are we doing this?

- "It's part of the rules"… yes, but what benefits are in it for you?

- Introduction to scientific working
- Introduction to (hopefully) interesting current research questions in the topic area

- Preparation for master thesis
  - …and for a possible PhD candidate position (if all this doesn't frustrate you too much ;-))

- But also:
  - Practice in OS & Co., not just theory from a single course
  - Every good systems paper has an implementation component (even those about formal proofs)

I want you to think *critically*

- Operating systems, runtimes and languages suffer from many problems
- One of the biggest: "But we've always done it this way"
  (= "we need to stay compatible at all cost")
- All of the topics in TDT09 are **open ended**
  - I **don't** want a **simple summary** of the papers I propose for a topic (that's boring – in fact, I read most of the papers already ;-))
    – you should find additional papers, ideas, concepts, etc.
  - Papers should serve as an inspiration and a background to help you structure the topic, show different angles to solve a problem and give an insight into often quite contrary views
    - This also means you don't have to read every one from front to end! (see "how to ***read*** a paper" in the links)
  - I would love to hear about **your** judgement and ideas on the topic
    - Try to abstract from the various papers (without ignoring them) and get a view of the overall problem area and solution approaches

Thinking critical by *critical reading*

- Reading a paper should give you an insight into the authors' work
  - …but you have to read between the lines!
- Don't trust the papers (and their authors) blindly!
  - Pressure on publishing good looking results is unfortunately high
  - If you suspect imprecisions/errors in papers you read, take notes
  - Were the concepts of the paper actually implemented?
    - ***Extremely important*** for a systems paper
- *"Beware of bugs in the above code; I have only proved it correct, not tried it"* – Donald Knuth
  - Were they implemented on a simulator or on real hardware? (or both?), were any simplifying assumptions made?
  - Which sorts of benchmarks were used?
    - Micro- vs. macro benchmarks, what was measured?
- *"lies, damned lies and benchmarks…"*

Thinking critical by *doing*

- Reading papers gives you an impression of the research effort
  - …but you have no experience actually using the thing/concept described
  - "Everything looks better on paper" (especially if details are missing in the paper)
- If possible, try out things
  - There are many emulators/VMs that help running older code
  - An amazing amount of code is out there (but certainly not all)
  - It's a great experience trying to get things to run
- Implementing this is ***a lot of work***
  - …sometimes very frustrating: "I wonder how the authors ever got this to work" (sometimes, they actually didn't!)
  - A paper usually doesn't tell the nitty-gritty details
- It's not a requirement for TDT09 to get to run something!

Topic areas

- We try to cover some typical problems of OS developers:
  - Resource management
  - Memory management
  - Transparency and modifiability
  - Security
  - New, complex hardware components & architectures
  - Software complexity, composability and portability
  - OS implementation approaches

The topics themselves

1. Do we even need an operating system?
2. Introspection/reflection on language/OS level - get to know your OS better
3. "Everything is a file, except when not"
4. Avoiding memory loss - persistent main memories
5. Protect yourself! Different approaches for security
6. Software composition - the evil of libraries
7. Massively parallel, completely different
8. Monolithic, micro-, exokernels - one size fits all?
9. Fun with abstractions - but what do they cost us?
10. New approaches to virtual memory - when a page is too big
11. C is not for everyone! OS implementation in higher-level languages
12. I don't care what CPU you use - binary translation

Links 1

- Finding scientific papers
  - ACM, IEEE Access via NTNU (VPN or being on campus required), Springer, others not
  - Homepages of authors, researchgate, academics…
  - Send an email to one of the authors – often, there is a "corresponding author" mentioned
  - Ask me!
  - Some website of a really friendly PhD student in Kazakhstan…
- Reading scientific papers – lots of advice online, e.g.
  - https://web.stanford.edu/class/ee384m/Handouts/HowtoReadPaper.pdf
  - http://www2.cs.uregina.ca/~pwlfong/CS499/reading-paper.pdf
  - https://people.cs.pitt.edu/~litman/courses/cs2710/papers/howtoreadacspaper.pdf
- Keith Downing's "Advice to Masters Students" has a number of good tips (except for his focus on AI topics ;-)): https://folk.idi.ntnu.no/keithd/advice/masters-students.html

Links 2

- Creating scientific presentations (content)
  - Good slide set by Andreas Zeller: https://www.st.cs.uni-saarland.de/edu/masterseminar/slides/the-perfect-talk.pdf
- Creating scientific presentations (slide formatting etc.)
  - LaTeX is great for papers
  - for presentation slides ("beamer" package): it depends
    - beamer users seem to create very text-heavy slides
  - Powerpoint/LibreOffice/Keynote/etc. are perfectly fine
  - NTNU templates: https://innsida.ntnu.no/wiki/-/wiki/English/Create+NTNU+presentations
- Organization
  - Save a copy of all papers you read and build an index
  - There are tools for this, e.g. to manage bibtex entries

Topic assignment

- Not right now – but I'd like to know who has already made up her/his mind and has a (or more) prefered topic(s)
- If two students definitely want to work on the same topic:
  - Fine with me!
  - Take opposing standpoints and collaborate, present your standpoints separately
  - Example: Papers on microkernel vs. hypervisor design
- So…
  - make a priority list of topics you are interested in
  - propose your very own topic if you have one
  - enter all in a joint document (is Google docs ok?)
  - until Sunday evening, if possible :)