How to do systems research

Peter Druschel

Max
Planck
Institute
for
Software Systems

Disclaimer and Credits

There is no recipe for doing (systems) research

- as far as I know
- this makes our job so exciting
- this makes it hard to explain how to do it

Following are some thoughts and suggestions, which I learned mostly from colleagues, collaborators, and advisers

The 3 most important things in systems research

1. Problem selection
2. Problem selection
3. Problem selection

How to find a good problem

- Not in future work sections
- Ask advisers, faculty, colleagues
  - see what they're working on, get feedback
- Do as systems researchers do: read, think, build, measure, iterate
- Look at
  - emerging technologies
  - new applications
  - boundaries of different subfields
- Do an internship
- Think big! (Your adviser will help ground you)

Is the problem important?

- Can you state it concisely?
  - elevator pitch
  - do others find it compelling?
- Are there broader principles / conclusions that can be drawn?
- What applications will benefit?
- Will it be of interest 5, 10 years from now?

Is the work novel?

- Do you understand the related work?
- Can you state concisely how your work
  - compares?
  - advances the state-of-the-art?

Who is your competition?

- Avoid crowded fields
  - unless you created it
  - or you have a truly game-changing idea
- Aim to write the
  - first,
  - best, or
  - last

  paper on a topic

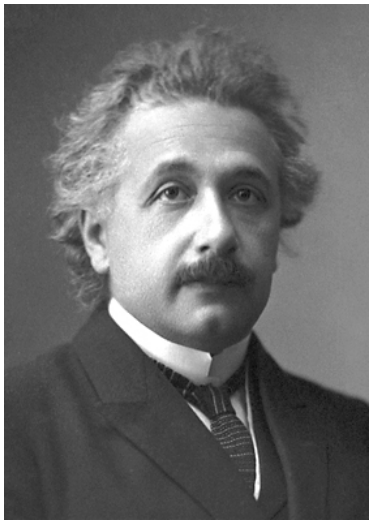
Do you have the tools?

- Does the topic play to your strengths?
- Do you / your advisers and colleagues have the expertise?
- Do you have access to the data, benchmarks, infrastructure required for a compelling evaluation?
- Can you build a first prototype in a few months?

Which community / conference will appreciate the work?

- Are the ideas appropriate, and considered significant enough, for a paper in that community?
  - look at past proceedings
  - talk to people who have served on the PC
- Will the work get the best visibility there?
  - who attends the conference, reads the proceedings?

Design



"Everything should be made as simple as possible, but not simpler."

-- Albert Einstein

See also Willy Zwaenpoel's
"*P2P, DSM, and Other Products from the Complexity Factory*"

Implemention

- Experimental apparatus designed to validate a hypothesis
- Know your hypothesis
- Clearly justify your choices
- Design, build, experiment, iterate
- Document your work meticulously
- If you perform a user study or network measurement, consider the ethics
  - Do you need IRB approval?

Evaluation

- Reproducibility
- Clearly state limitations of
  prototype, data, benchmarks, experiments
- Isolate the effect of your contributions
- Show *why* it works, not just *that* it works
- Avoid benchmarking crimes (see Gernot Heiser's list)
  - change one variable at a time; expose hidden costs
  - appropriate baseline; don't rely on micro benchmarks
- Let the data speak
- Draw appropriate conclusions

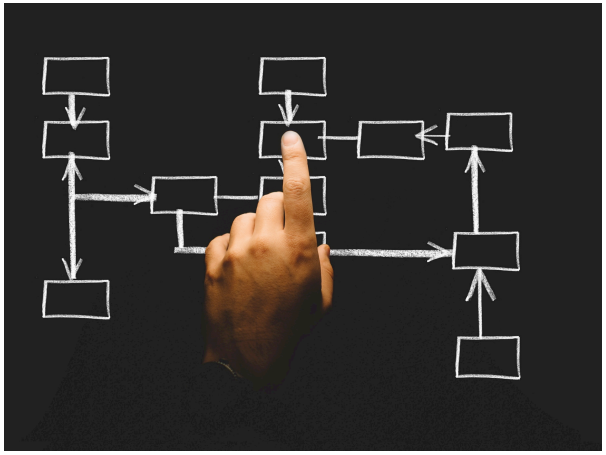
Writing

"Only two types of papers are accepted:
Good papers that are well written and
mediocre papers that are well written"

-- Larry Peterson

- Be aware of your audience / community
- Take your time to write, seek feedback, iterate

Organization



- Structure should reflect contributions
- Break work into components that can be understood individually

1. Introduction
2. Related work / background
3. System Overview
4. Subsystem 1 (first contribution)
5. Subsystem 2 (second contribution)
6. …
7. Evaluation
8. Conclusion

Introduction

- Motivate broadly, narrow down, pitch, position
  - engage the reader
- Manage expectations
  - Disclose and mitigate limitations
- Draft early, get feedback, iterate

Scott Shenker, World Champion

Best practices

- Clarity, Concision
- Teach first, impress second
- Treat related work fairly
  - clear and explicit comparisons
- Clear take-away points / conclusions
- Attention to detail
  - spelling, grammar, consistency, citations
  - sloppiness undermines readers' trust

Dealing with the Echo: Paper accepted

- Congratulations!
- Use feedback to improve your work
- Prepare a *great* talk!
- How can you maximize impact?
  - Is there a broader, longer-term agenda?
  - What can you do to encourage adoption?
  - Make the code/data available
  - Give talks, engage with practitioners
  - Consider a follow-up journal article

Dealing with the Echo: Paper rejected

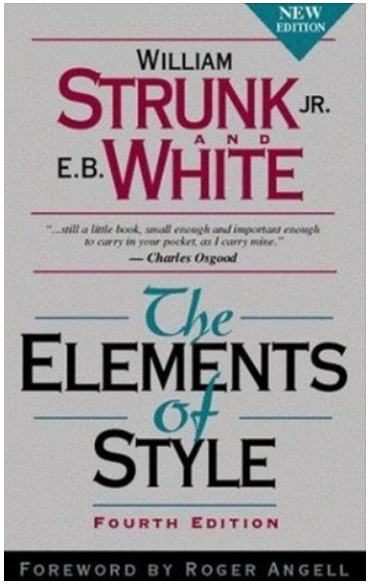
- Don’t despair, it is a part of the game
- Don’t blame the reviewers
- *Clarity:* Can you improve the writing?
- *Novelty:* Have you covered all related work, made appropriate comparisons (quantitative, qualitative)?
- *Importance:* How can you make the case stronger?
- *Contribution:* Can you generalize the solution, strengthen the evaluation?
- *Fit:* Did you submit to the right venue?

Finally: Have fun!

We’re in a privileged profession! We get to work

- on exciting problem we can choose ourselves
- with extremely smart and accomplished colleagues
- on new challenges all the time

Other resources



- *"How (and How Not) to Write a Good Systems Paper"* , Roy Levin and David D. Redell
- *"You And Your Research",* Richard Hamming
- *"Top-10 tips for writing a paper",* Jim Kurose
- *"Ten Things I Wish my Adviser Had Told Me"*, Jim Kurose
- *"How to Have a Bad Career in Research/ Academia"*, David Patterson et al.
- *"Graduate School Keys to Success"*, Remzi Arpaci-Dusseau