Research Topics in Software Engineering

Zhendong Su

Autumn Semester 2018
(based on Prof. Peter Müller’s slides)
Objectives

- Learn to present technical work
- Learn to understand & evaluate research papers
- Learn several key research directions in the area
Objectives

- Learn to present technical work
- Learn to understand & evaluate research papers
- Learn several key research directions in the area
- Have a good deal of fun doing so
Preparing a Talk

Check your presentation date

Study your paper(s)

Create draft presentation

Meet advisor, get feedback
Preparing a Talk: **Start Early**

- Preparing a good presentation takes time
- So, please start early!

1. Check your presentation date
2. Study your paper(s)
3. Create draft presentation
4. Meet advisor, get feedback
Preparing a Talk: **Study Paper**

- **3 ‘C’s of reading**
  - Carefully: look up terms, possibly read cited papers
  - Critically: find limitations, flaws
  - Creatively: think of improvements

- **Try examples** by hand
- **Try tools** if available
- **Consult with TA** if questions
Preparing a Talk: **Create Draft**

- Explain the work’s **motivation**
- Clearly present the **technical solution** and **results**
  - Use **own examples**, not the ones in the paper
  - Include a **demo** if appropriate
- Discuss **limitations** or **improvements**
- Focus on **key concepts**
  - Do not present all the details
Preparing a Talk: **Get Feedback**

- **Prepare** for the meeting
  - Schedule early
  - Send slides in advance
  - Write down questions
- **Do** address feedback
  - Take notes
- **Meeting is mandatory!**
  - At least 1 week before the talk
Grading

- Quality of your presentation
  - How well did you understand the material?
  - How well did you present it?
  - How well did you answer the questions?

- Participation
  - Did you ask good questions?
  - Did you attend all sessions?

- We will take into account
  - the difficulty of the paper
  - suggestions you received from your TA advisor
  - time you had to prepare
Feedback

- Discuss your talk’s strengths/weaknesses in-class
  - Let us know upfront if you prefer us not to

- Arrange a meeting with your TA to get feedback
Schedule

- Meet once per week with ~2 presentations each time
  - Next meeting on October 11

- Detailed schedule will be published online shortly
  - https://people.inf.ethz.ch/suz/teaching/263-2100.html
  - Including names of TA advisors
Workshop on Dependable and Secure Software Systems

Oct 19-20, 2018
Alumni Pavillon, Rämistrasse 101, ETH Zürich

ETH Zürich

- Big Data Program Analysis
- SMT Solver Tactics
- Machine Learning for Analysis
- Compilers for Analog Hardware
- Practical Verification for Systems
- Safety of Deep Learning
- Abstract Interpretation
- Security Analysis
- Blockchain Semantics

More information and registration: http://www.sri.inf.ethz.ch/workshop2018
Your Talk: **Timing**

- 30 min for talk
  - 1.5 ~ 2 min per slide
- 15 min for Q&A and discussion

- The pace of talk is important
  - Too fast, the audience cannot follow
  - Too slow, people can get bored

- Practice your talk
  - Checkpoint after ~10 minutes
Ownership Transfer in Universe Types

External Uniqueness

- Partition context into clusters
- Clusters can be unique or not
- At most one read-write reference into a unique cluster
- Arbitrary aliasing within cluster
- Any references not restricted
- Unique clusters are transferred as a whole

Related Work

- External Uniqueness
  - Types and ownership transfer
  - Destructive reads and borrowing
  - Alias Java
    - Type-safe ownership transfer
    - Lint variables break encapsulation
    - Alias burying
      - Static analysis to track temporary aliases
      - High-suspicion overload, limited by static analysis
  - Object invariants
    - Similar to enforcement of uniqueness invariant

Merging List Representations

- Refine ownership per unique field
- Use (multiple) result values

Solution

Evaluation, experiments, demo

Summary, conclusion

Meet me at the Microsoft booth
- Also to get a Specific demo
Your Talk: **Examples**

- Examples are crucial for understanding
  - Both yours and the audience’s
  - Prepare your own examples
- Try to find a running example
  - For motivation, problem, solution
  - Explain in detail (takes time)
- Reduce code example to the absolute necessary
  - Most people hate reading code
  - Use visualizations

---

Zhendong Su – Research Topics in Software Engineering
Your Talk: **Design**

---

### Ilvm bug autopsy

```c
struct tiny { char c; char d; char e; };

void foo(struct tiny x) {
    if (x.c != 1) abort();
    if (x.e != 1) abort();
}

int main() {
    struct tiny s;
    s.c = 1; s.d = 1; s.e = 1;
    foo(s);
    return 0;
}
```

- **GVN:** load struct using 32-bit load
- **SRoA:** read past the struct’s end → undefined behavior

```
$ clang -m32 -O0 test.c ; ./a.out
$ clang -m32 -O1 test.c ; ./a.out
Abort (core dumped)
```

---

**Zhendong Su – Research Topics in Software Engineering**
Powerpoint vs. Latex

- **Powerpoint**
  - Visualizations and animations are easy
  - Don’t over-do it!

- **Latex**
  - Visualizations and animations are painful
  - Don’t under-do it!
Your Talk: **Avoid Frequent Mistakes**

- Don’t try to present all details
  - Focus on the key messages: motivation, problem, main idea, main result

- Don’t stare at the screen or your laptop
  - Look at the audience

- Come prepared
  - Study paper in depth
  - Rehearse your talk
References

Markus Püschel’s small guide on giving talks


Highly recommended!
Paper Pool for Fall 2018

10. Automatically improving accuracy for floating point expressions. PLDI 2015.
27. Bringing the web up to speed with WebAssembly. PLDI 2017.