Lecture 2

ECS 289C: Seminar in Programming Languages

Plan

1. Finish activity from last time
2. Areas of PL
3. How to Read a Paper

Announcements

I'll assume enrollment is now roughly stable:

- 17 of you are registered

That means:

- Only 1 discussion lead
  - (+ presentation for your final project)

Therefore:

- Reducing discussion lead to 10%
- Adding 5% attendance, 5% for Perusall

Summary on out-of-class participation

1. Write 1 comment + 1 question on Perusall
2. Fill out a reading summary for each paper and post it on the week's Piazza thread

Announcements

Apologies, I am a bit behind on finalizing the paper list

Monday: Paper assignments

First discussion: Wednesday

Activity from last time

There may be multiple valid answers

- I’m here to help + can answer questions

Why?

- Help think about paper **scope** and **contributions**
- Active learning
- Useful beyond just PL research

Activity from last time: review

5 levels:

1. Major area (conference)
2. Subarea (conference session)
3. Research topic
4. Research problem
5. Potential solution

Additional level:

6. **Contribution:** Proof that it works

Example paper

1. Major area (conference)

This is a freebie: PL

Outside of PL: Major areas of CS are usually listed on any department website (or in graduate course offerings)

- ECS 289A, 289B, 289C, …

2. Subarea (conference session)

- Look up the conference: what session was the paper presented in?
- Is the paper in a **second** area?
  - Architecture
  - ML
  - Systems
  - Theory
  - Etc.
- Look at the **title** only – usually not the abstract

2. Subarea (conference session)

Additional hints:

- Are **keywords** listed after the abstract?
- Are there any **technical words that reoccur** in the title of a lot of different papers?
- **Conference/journal name**, if the paper was not in a major/flagship conference

3. Research Topic

Level 3 is the hardest!

Look at the **first 1-2 sentences** of the abstract (often the first sentence)

Subject to some interpretation

Look for general motivation – not the specific problem, but a general set of problems or scope of stuff being studied

4. Research Problem

Papers will tell you exactly what the problem is in the abstract

- Often the 2nd sentence

Look for:

- The problem is …
- (existing solution) cannot handle …
- (existing solution) is limited to…
- (existing solution) is difficult/hard/too expensive/too slow

5. Potential solution

Papers will tell you exactly what their solution is in the abstract

Look for:

- We introduce…
- New/Novel
- Our solution…
- Our key insight…
- Our main idea…
- We propose…

6. Contribution

How did they validate that the potential solution works?

Look for:

- We show that…
- We demonstrate that…
- We prove that…
- Experimental results
- Theoretical results (like a theorem or a proof)

Activity from last time: finishing up

PL → property-based testing (PBT), verification, validation → test input generator for type-based verification → complex data structure generated by PBT → novel interpretation of types

PL → memory safety → memory safety bugs in GPUs → existing tools are not efficient → Tool for High error detection and lower runtime overhead

PL → Types / Runtime Environments → Garbage collection safety → Region inference is complex, find a solution for higher-order programs → simple region type system

PL active research areas

PL active research areas

From PLDI 2023:

- Verification (x3)
- PL + ML
- Compilers (x2)
- Concurrency & Parallelism
- Security
- Synthesis
- Probabilistic Programming
- Program Analysis
- Testing
- Types
- Program Logics
- Parsing
- Systems

From POPL 2024:

- Synthesis (x2)
- Types (x5)
- Side effects (x2)
- Verification (x3)
- Regular Expressions & Automata
- Program Logics (x3)
- Concurrency & Parallelism (x2)
- DSLs
- Probabilistic Programming
- Quantum
- Program Analysis
- PL + ML

PL active research areas

Consolidated

- Logic + Semantics (x8)
- Verification and Testing (x7)
- Types (x6)
- Domain Specific Languages (x6)
- ML and Synthesis (x5)
- Compilers + Program Analysis (x4)
- Concurrency + Parallelism (x3)

Some additional data

Type Systems	39
Compilers	38
Language design	31
Verification	26
Testing and SE	24
Static Analysis	22
Systems and Security	22
Concurrency and Parallelism	21
Logic and Semantics	18
ML and Synthesis	17

Activity from last time: finishing up

PL → property-based testing (PBT), verification, validation

PL → memory safety

PL → Types / Runtime Environments

S. Keshav: How to Read a Paper

Pass 1

Ballpark idea

- You should be able to answer the 5 levels + contributions
- Identify keywords/areas you don’t understand

Are you interested in reading more?

- Most papers don’t make it to level 2 or level 3

Pass 2

Read **in more detail** the parts that you are interested in

Try to understand the examples! If they didn’t write one, come up with your own

Note on old papers:

- Many papers are famous for some particular idea, the rest of the paper may have fluff that didn’t turn out to be useful.

Pass 3: re-implement the idea

Only for papers very close to your research interests
(Basically, 4 of the 5 levels should match!)

Questions that people had?