Lecture 3:

Do over?

Three framing questions for the semester.

Q1. Cross-domain tracing

Desirable
Q2. Who uses traces & how?

Focus thus far: Humans for debugging.

What about other uses?

Reproducing bugs

What?
Why challenging right now?

Security
- What?
  - Audit
  - Unusual data flow
  - Why challenging now?

Scheduling, auto scaling, resource allocation
- What?
Why challenging?

Q3. Trace sizes: why are they so large?
  - Why this matters?

  - Fundamental question: what do we need to record?
    - Depends on use.
      - For reproducing bugs (proxy for human?)
        - For security


-> Other uses?

- Does reducing trace size come at a cost?

---

**Today's Papers**

- Reduce trace size by identifying & recording what is necessary.

  > Two related (?) approaches

  How do they fit into the questions?
WHAT DO WE DO WITH THE TRACES? (WHO & HOW)

WHAT DO TRACES RECORD? (WHY ARE THEY SO BIG?)

UNFORTUNATE REALITY: KNOWING HOW TRACES ARE GOING TO BE USED DOES NOT ALWAYS REDUCE TRACE SIZE.

WHY?
Challenges

- How to decompose? [API Design]

- A single process might not have enough information for its part of the query? [✘, baggage]

- Limit what a process records. [API, baggage, ✘]

How to decompose / API design.
Not restricted to one service

From \( x \) in \( Y \)

\[ \text{Join A in B on } A \rightarrow x \]

...\(^{\text{hook}}\)

- Given a query can identify
  - Where data needs to be collected
  - Using what hooks

Two related problems

- From \( A \) in \( X \)
  
  \[ \text{Join B in Y on } B \rightarrow A \]
  
  where \( B_{\text{col}} = \ldots \)

- From \( A \) in \( X \)
  
  \[ \text{Join B in Y on } B \rightarrow A \]
SELECT A.i, B.o

How this is classically implemented

Many problems

FROM A IN X
JOIN B IN Y ON B->A
WHERE B.o = ...
More Generally

Pivot Tracing Approach

What About Causality?
CHALLENGES?

Ergonomics & Adding trace points?

Sampling? Not sampling? Overheads?
Where does snicket fit in?

WHERE & WHEN DOES THIS WORK?