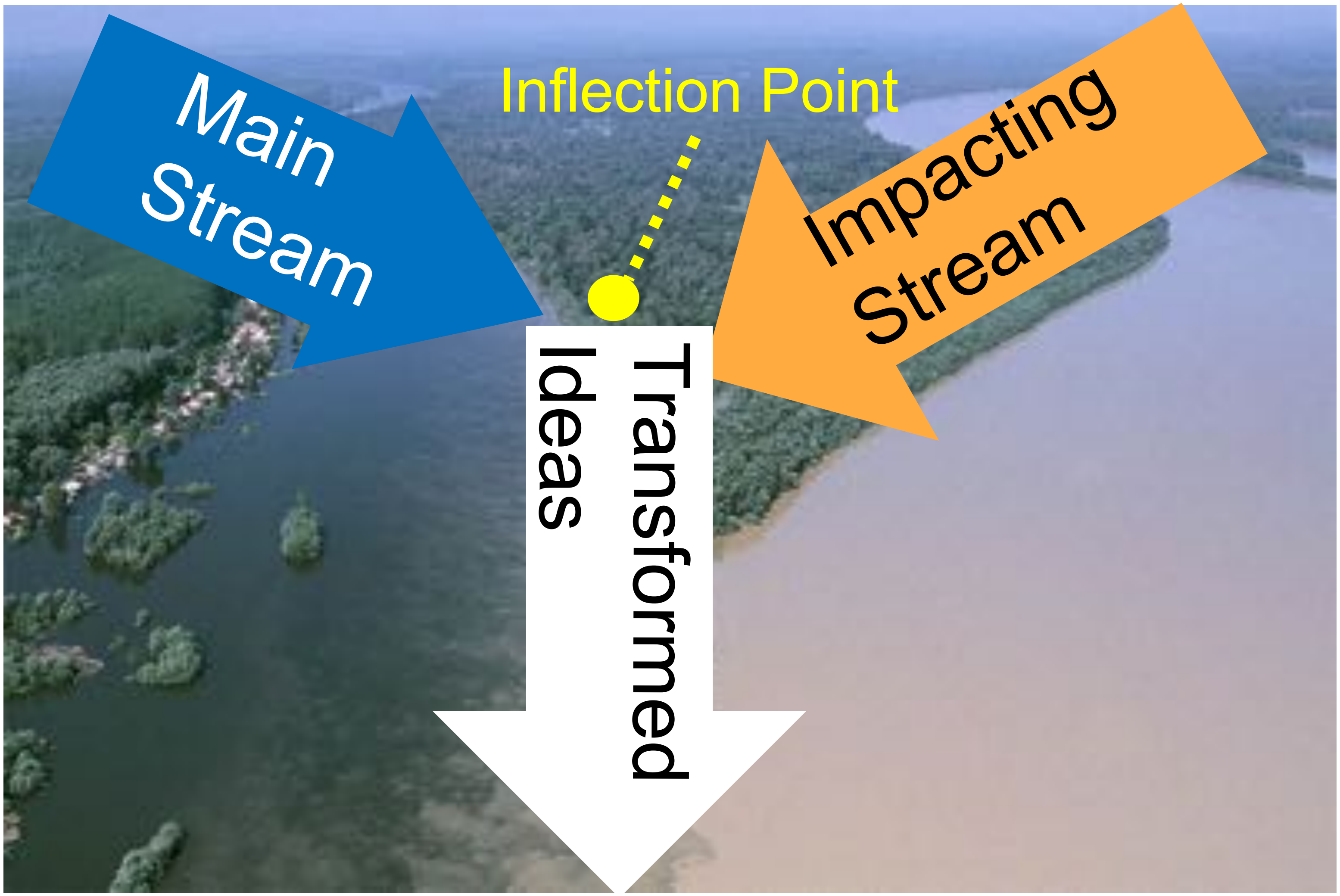
Re-Engineering Software Engineering in a Data-Centric World

Miryung Kim
University of California, Los Angeles

Confluence

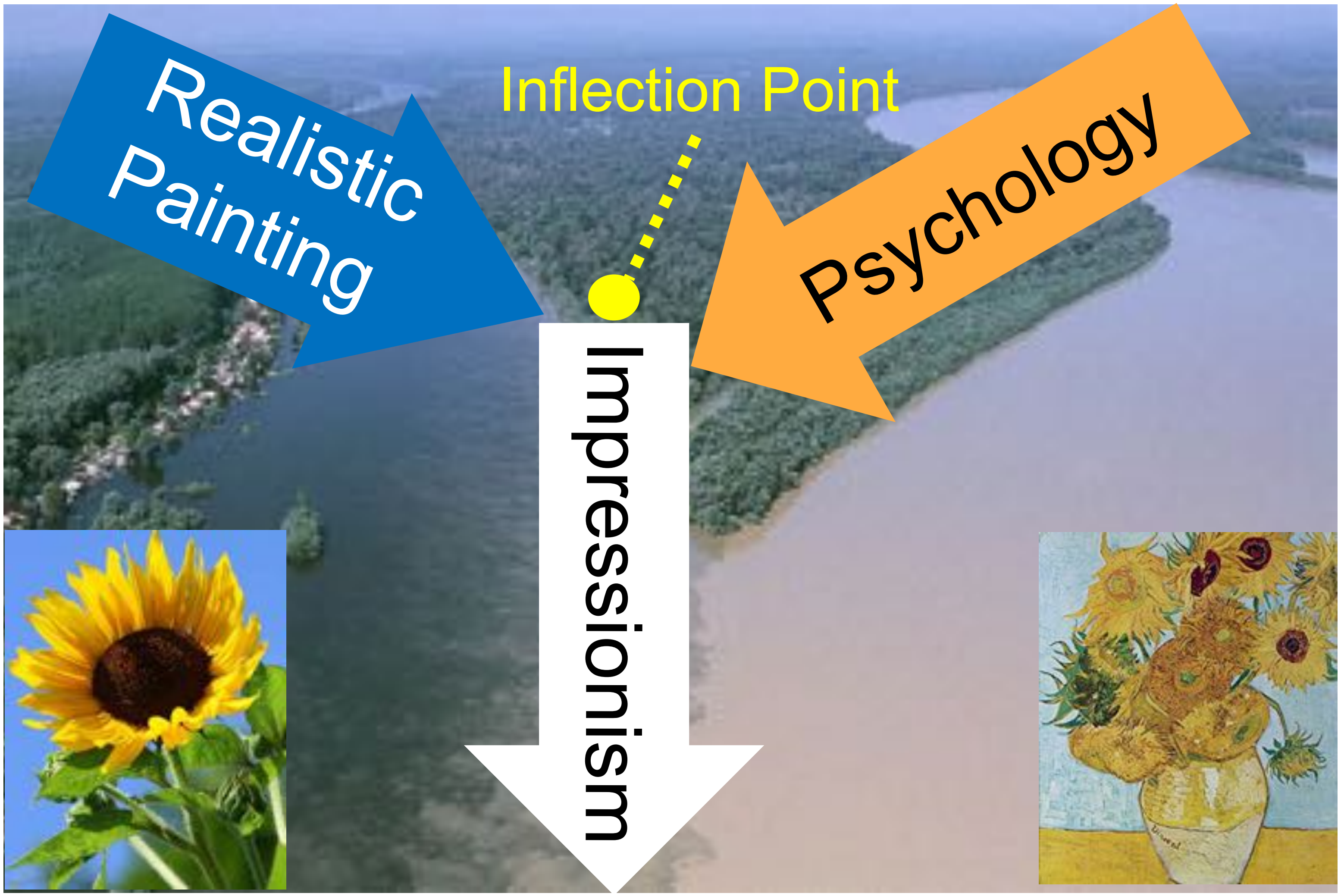
Interdisciplinary thinking via confluences, George Varghese @ SIGCOMM 2014 Keynote

Confluence: Interdisciplinary Thinking



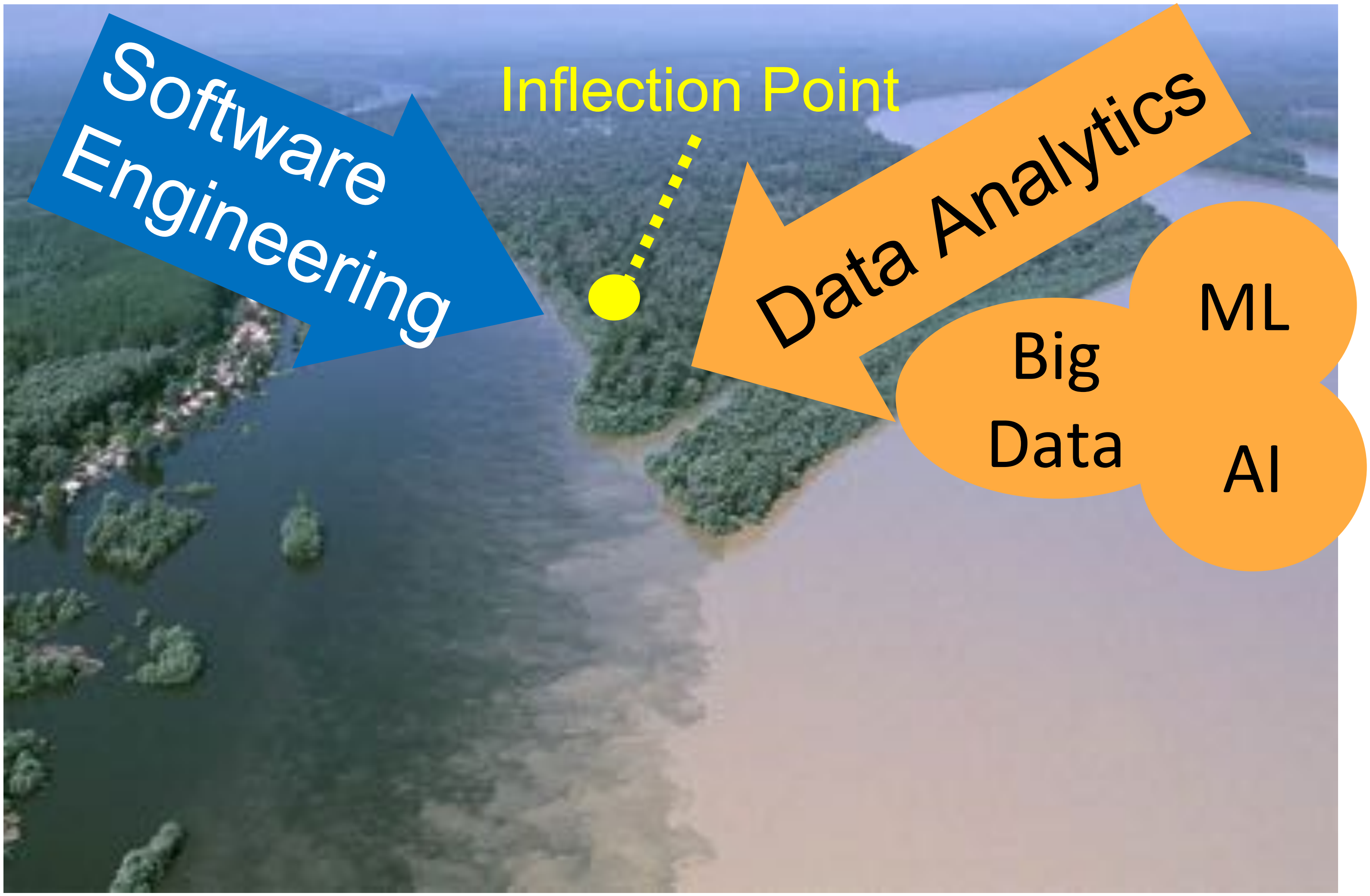
Interdisciplinary thinking via confluences, George Varghese @ SIGCOMM 2014 Keynote

Confluence: Impressionism



Interdisciplinary thinking via confluences, George Varghese @ SIGCOMM 2014 Keynote

Confluence: Data Analytics and SE



Interdisciplinary thinking via confluences, George Varghese @ SIGCOMM 2014 Keynote

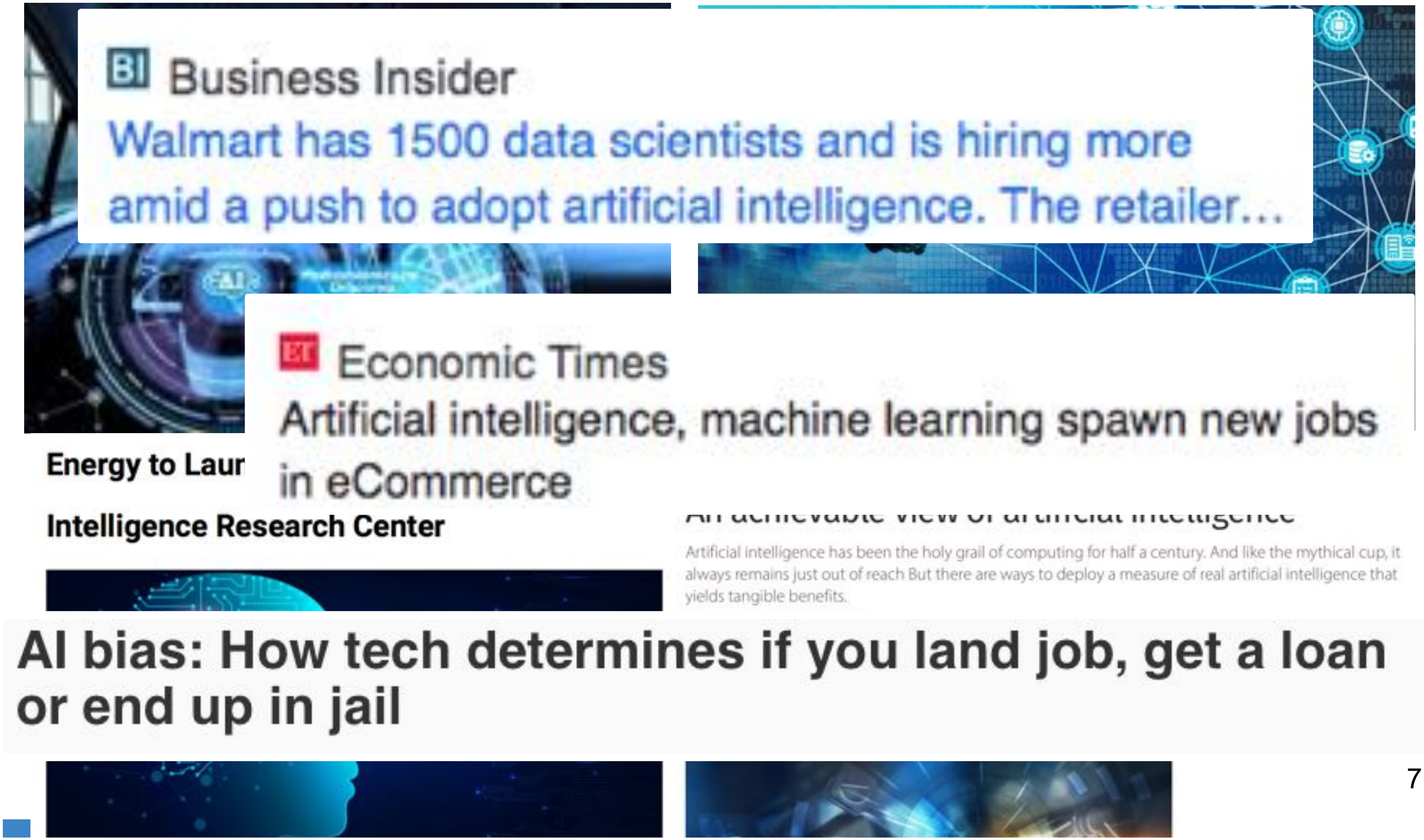
Takeaway Message: A Case for **Software Engineering for Data Analytics (SE4DA)**

Bug finding is a huge problem in data analytics.

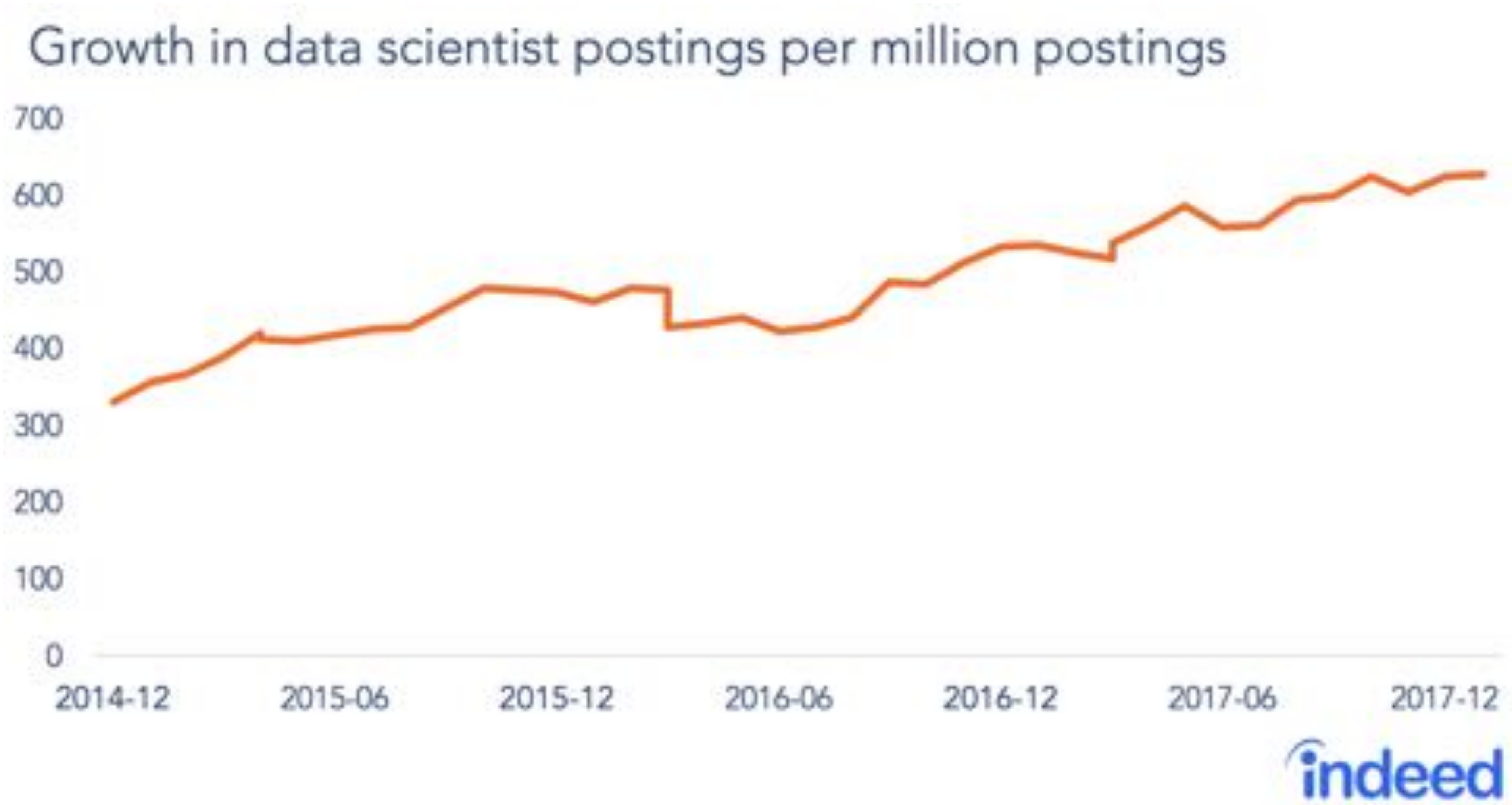
SE4DA is **underserved**; somehow people have gravitated to applying data analytics to SE.

SE4DA requires **re-thinking software engineering** techniques.

There is a huge opportunity for data analytics.



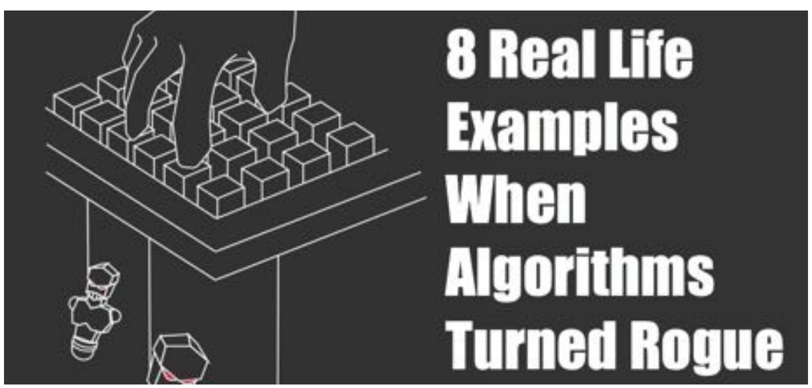
Data analytics are in high demand, yet …



Bugs are huge problems in data analytics.

Data analytics used by thousands of scientists produce **misleading** or **wrong results** [BBC News]

Predictably inaccurate: The prevalence and perils of bad big data. [Deloitte]

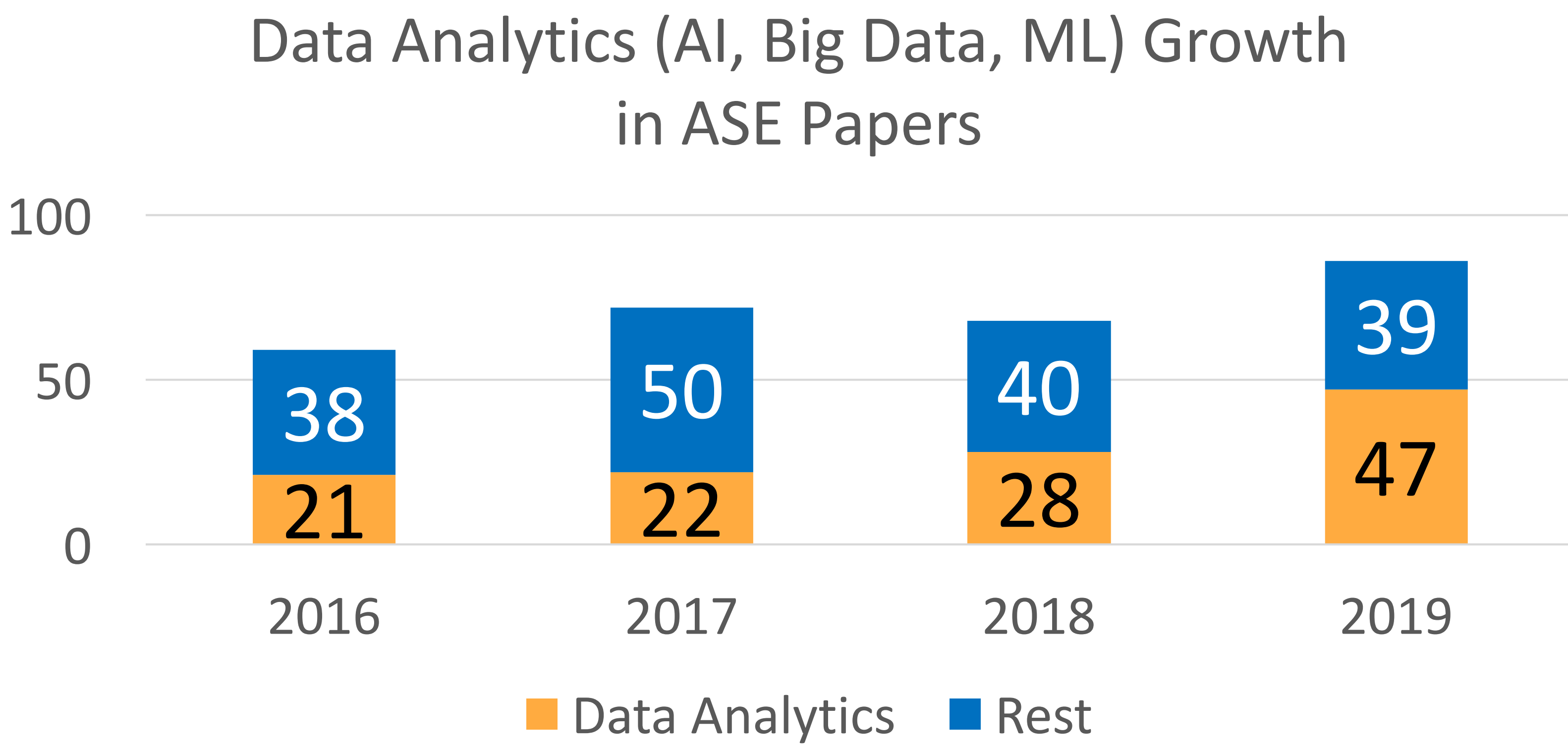


The widespread harm includes from a **wrong medical diagnosis** to **incorrect interpretation** of stock history [Dataversity]


Franken-algorithms: the deadly consequences of unpredictable code

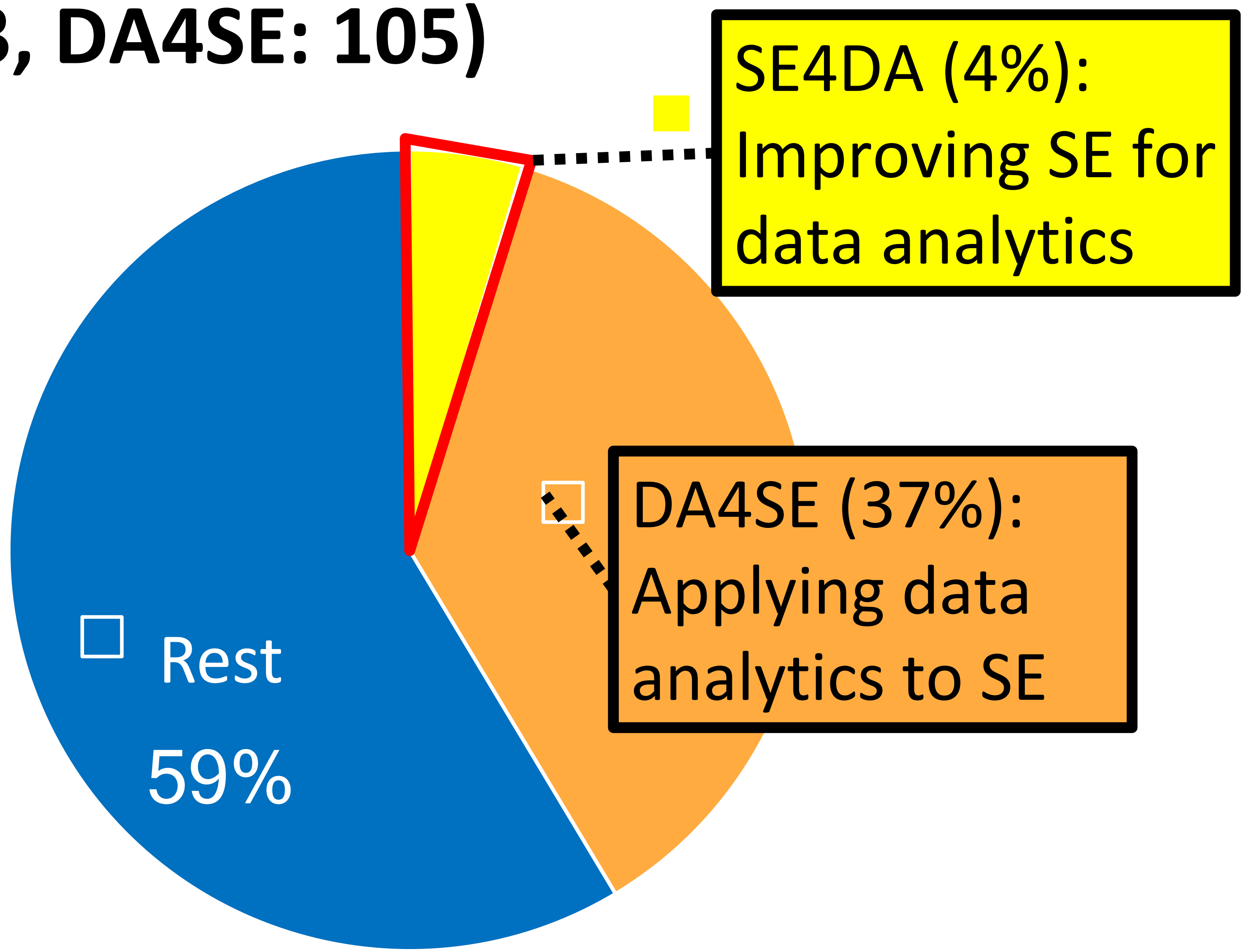
The death of a woman hit by a self-driving car highlights an unfolding technological crisis, as code piled on code creates 'a universe no one fully understands'

Growth of Data Analytics Papers in SE



SE4DA is under-investigated.

(SE4DA: 13, DA4SE: 105)



Outline: Making a Case for
Software Engineering for Data Analytics (SE4DA)

① Shift to **data-centric SW development**

② Differences between **traditional SW vs. data-centric SW dev process**

Studies: Data Scientists

③ **Debugging** & **testing** for **big data analytics**

Tools

④ **Open problems** in SE4DA

Part 1. Data Scientists in Software Teams: State of the Art and Challenges

Miryung Kim, Thomas Zimmermann, Rob DeLine, Andrew Begel

UCLA Microsoft

The Emerging Roles of Data Scientists on Software Teams

We are at a **tipping point** where there are large scale telemetry, machine, quality, and user data.

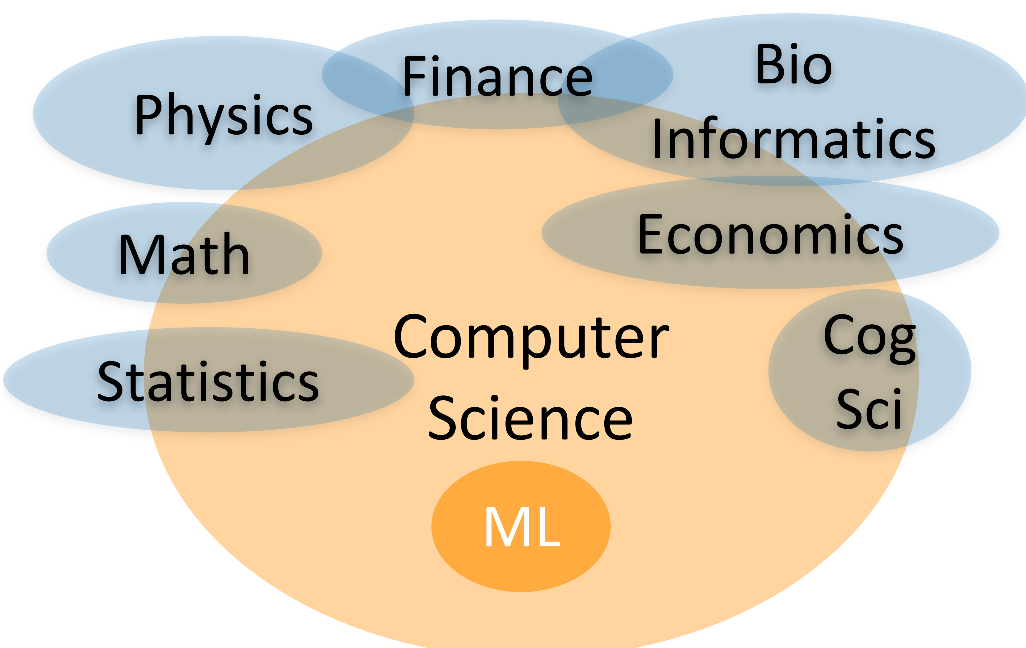
Data scientists are **emerging roles in SW teams.**

To understand **working styles** and **challenges,** we conducted the first in-depth interview study and the largest scale survey of **professional data scientists.**

Methodology for Studying "Data Scientists"

In-Depth Interviews [ICSE'16]:

- 5 women and 11 men from eight different Microsoft organizations



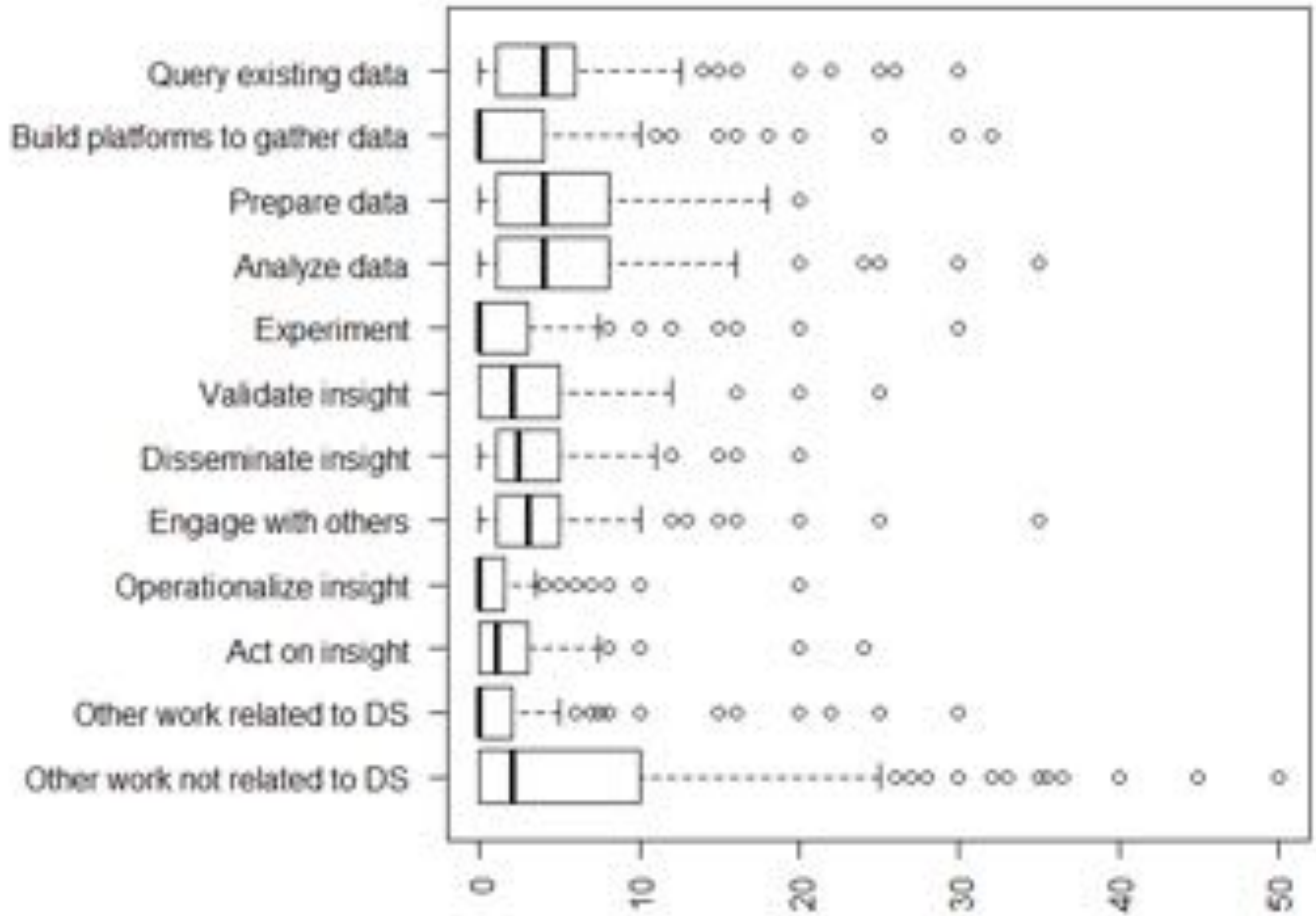
Survey [TSE 2018]

793 responses

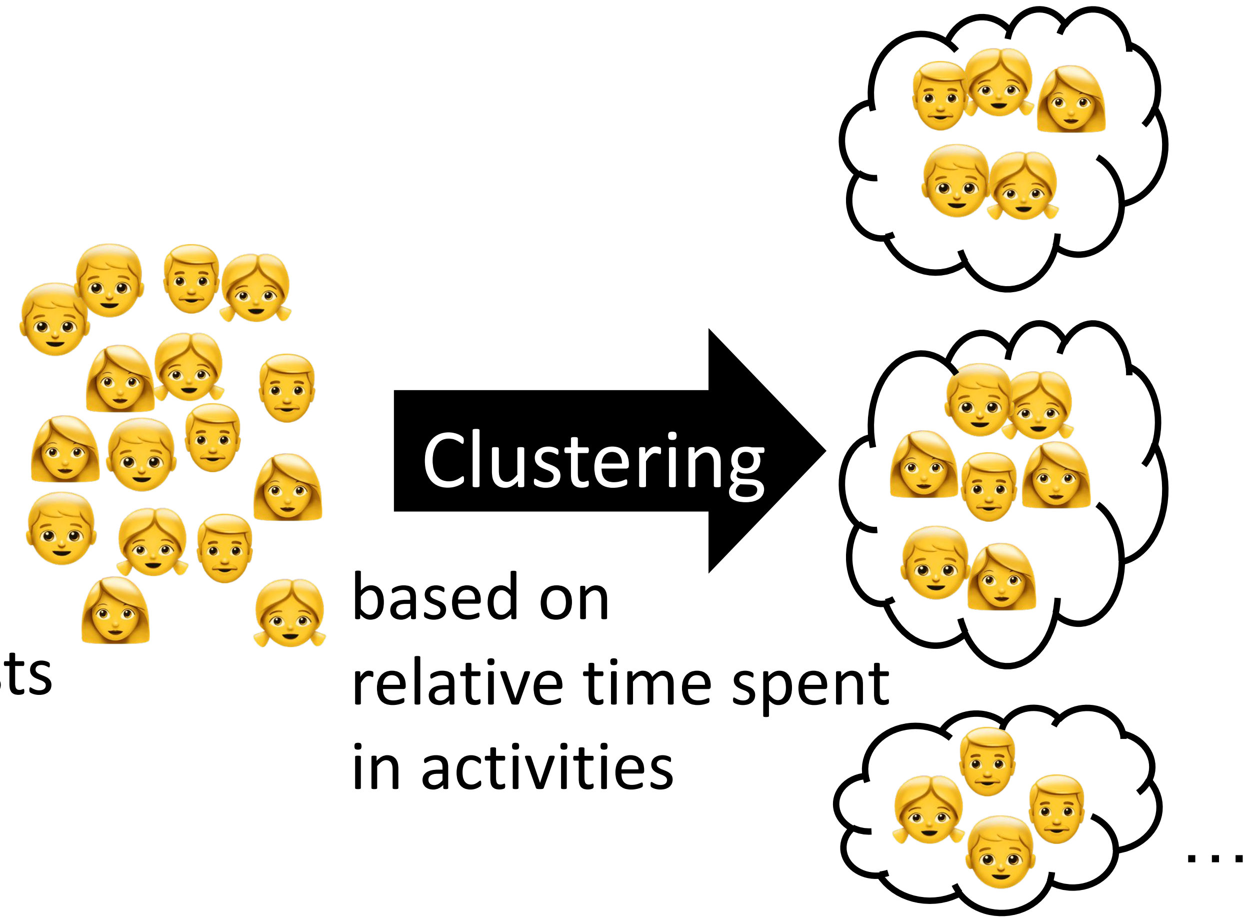
- demographics/self-perception
- skills and tool usage
- working styles
- time spent
- challenges and best practices

Time Spent on Activities

Hours spent on certain activities (self reported, survey, N=532)



What is a “Data Scientist”?



532 data scientists
at Microsoft

9 Distinct Categories

Category 1: Data Shaper

- Analyzing and preparing data
- Post-graduate degrees
- Algorithms, machine learning, and optimizations
- Less familiar with front-end programming

Category 2: Platform Builder

Instrument code to collect data

Big data and distributed systems

Back-end and front-end programming

SQL, C, C++ and C#

Category 3: Data Analyzer

Familiar with statistics

Not familiar with front-end programming

Difficulty with data transformation

R Studio or statistical analysis

Common challenges: Data scientists find it difficult to ensure "correctness"

Validation is a major challenge.

"Honestly, **we don't have a good method** for this."
"Just because the math is right, doesn't mean that the answer is right."

Explainability is important— "to gain insights, you must go one level deeper."

① Data Scientists | **② Difference** | ③ Tools | ④ Challenges

Outline: Making a Case for
Software Engineering for Data Analytics (SE4DA)

① Shift to **data-centric SW development**

Studies: Data Scientists

② Differences between **traditional SW vs. data-centric SW dev process**

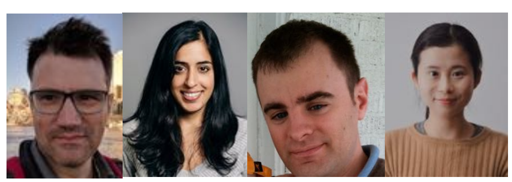
③ **Debugging** & **testing** for **big data analytics**

Tools

④ **Open problems** in SE4DA

Part 2. How is Traditional Development Different from Big Data Analytics Development?

[ICSE'16] [TSE'18]



[Interactions'12] [ICSE-SEIP'19]
[NIPS'15] [TSE'19]

Traditional vs. Big Data Analytics Development

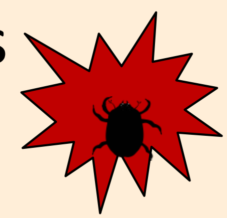
1. Develop
2. Run
3. Test
4. Debug
5. Repeat

1. Develop locally

2. Test locally with Sample Data

3. Execute the job on the cloud hoping that it would work

4. Several hours later, the job crashes or produces wrong output



5. Repeat

Traditional vs. Big Data Analytics Development

1. Develop locally

2. Test with Sample

1. Data is **huge, remote**, and **distributed**.

Traditional vs. Big Data Analytics Development

2 Test with Sample

4 The job crashes or produces wrong output

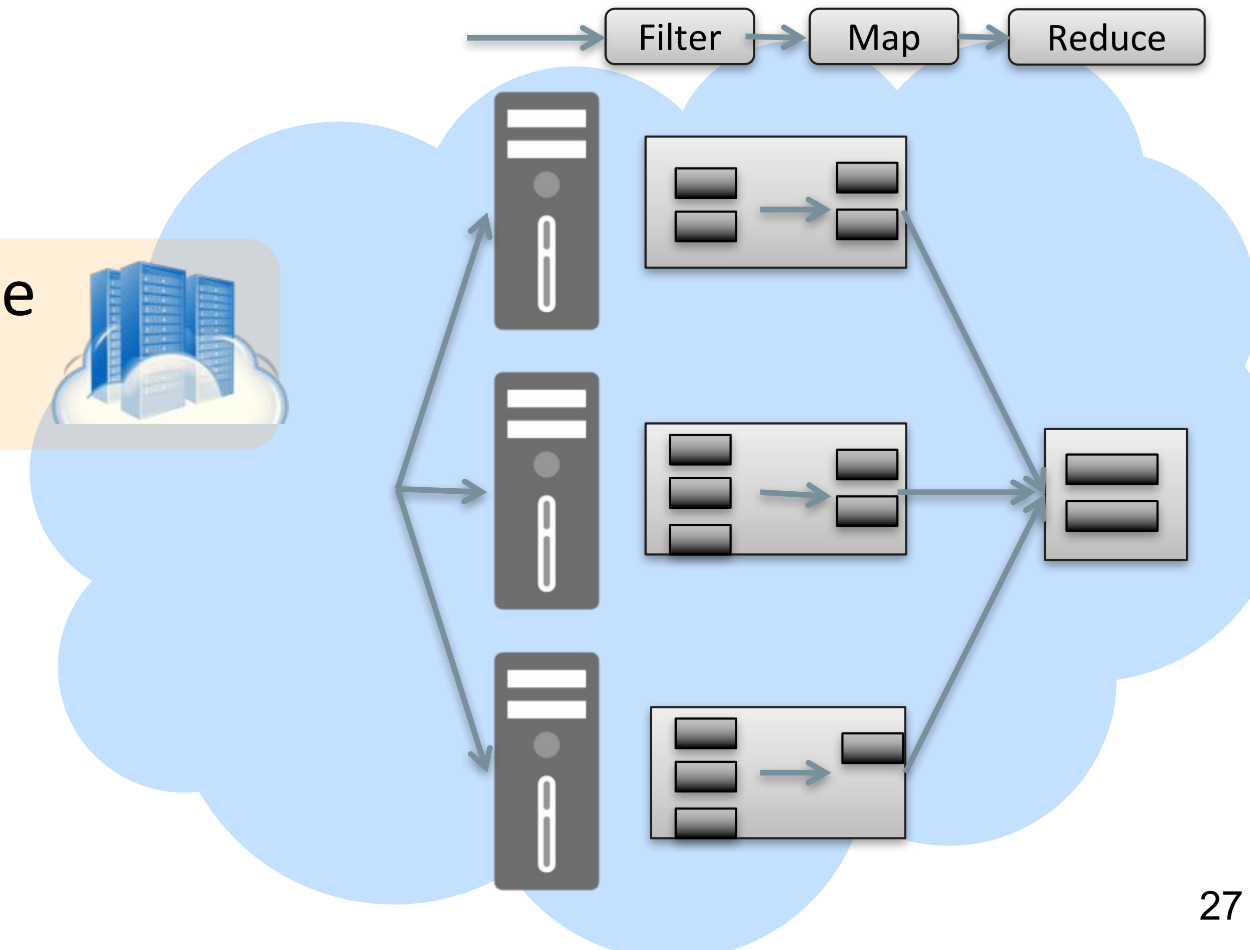
2. **Writing test** is **hard**.

Don’t even know the full input and don’t know the expected output.

3. **Failures** are **hard to define.**

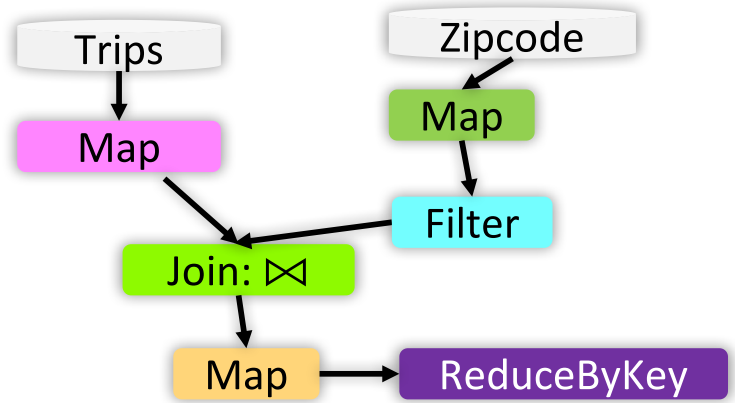
Traditional vs. Big Data Analytics Development

4. **System stack** is **complex** with **little visibility.**



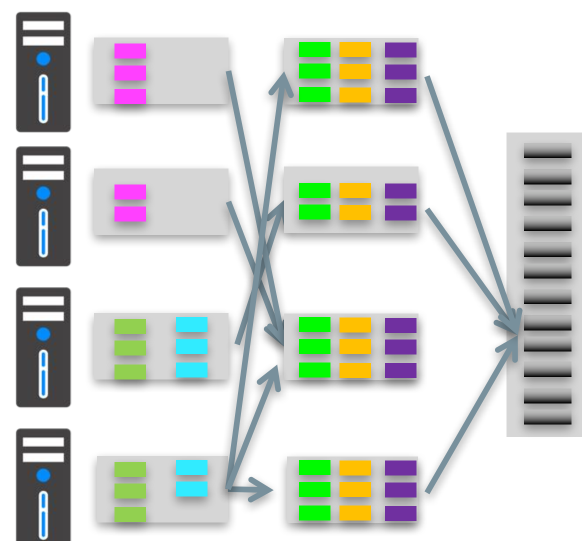
3 Execute the job on the cloud

Traditional vs. Big Data Analytics Development



3 Execute the job on the cloud

5. **Gap** between **logical** vs. **physical** execution



Traditional vs. Big Data Analytics Development

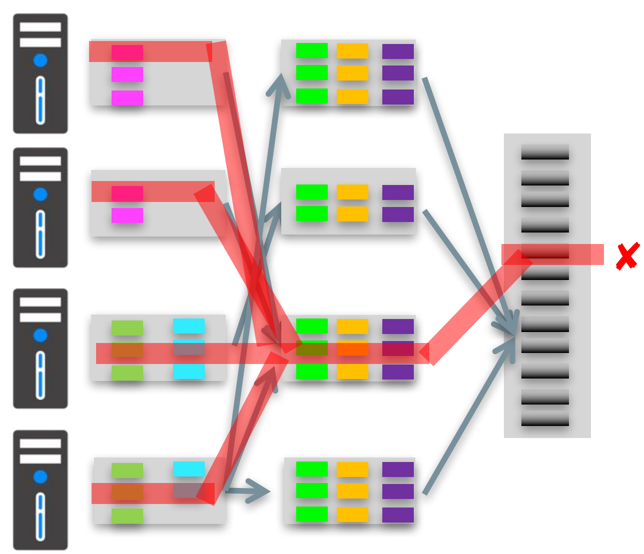
```
Task 31 failed 3 times; aborting job
ERROR Executor: Exception in task 31
in stage 0 (TID 31)
java.lang.NumberFormatException
```

3. Execute the job on the cloud

4. The job crashes or produces wrong output

5. Repeat

6. Data **tracing** is **hard.**



Outline: Making a Case for
Software Engineering for Data Analytics (SE4DA)

① Shift to **data-centric SW development**

② Differences between **traditional SW vs. data-centric SW dev process**

Studies: Data Scientists

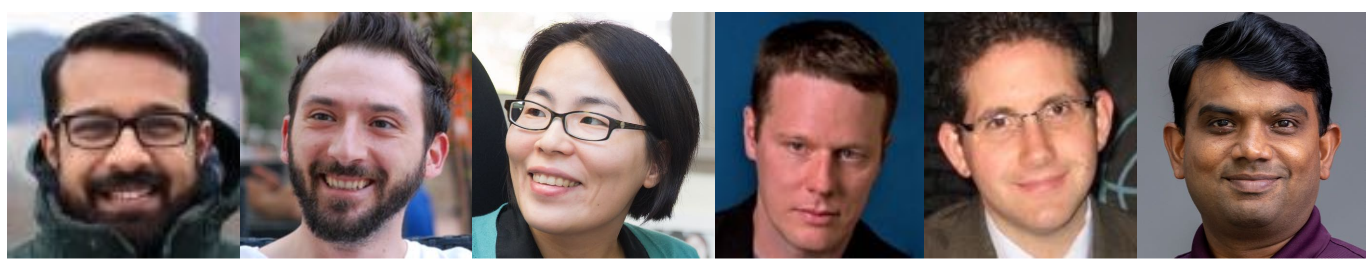
③ **Debugging** & **testing** for **big data analytics**

Tools

④ **Open problems** in SE4DA

Part 3. Debugging and Testing for Big Data Analytics

Tyson Condie, Ari Ekmekji, Muhammad Ali Gulzar, Miryung Kim, Matteo Interlandi, Shaghayegh Mardani, Todd Millstein, Madanlal Musuvathi, Kshitij Shah, Sai Deep Tetali, Seunghyun Yoo



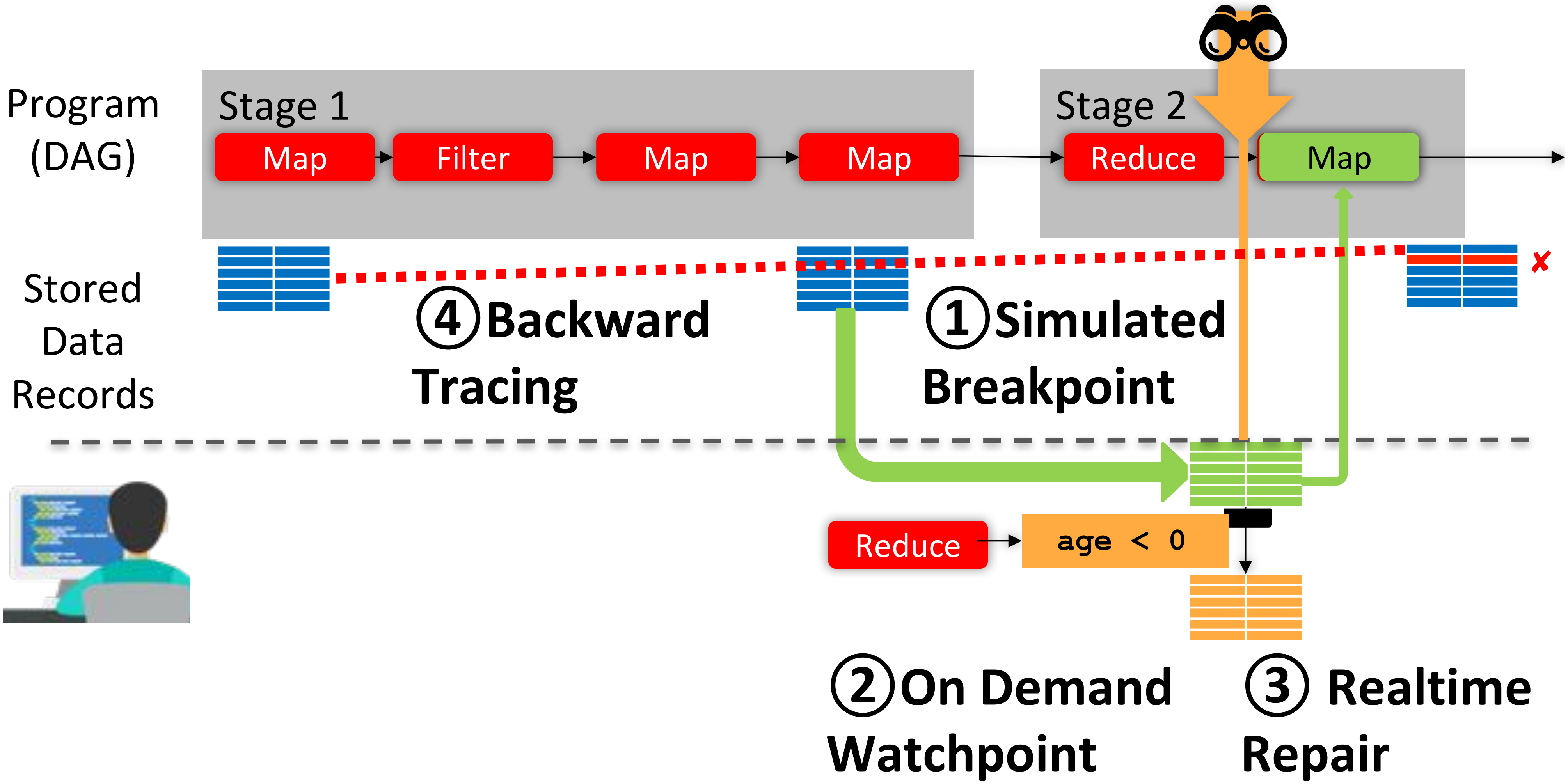
Insights from Debugging and Testing for Apache Spark

- Designing interactive debug primitives requires deep understanding of **internal execution model, job scheduling, and materialization**.
- Providing traceability requires **modifying a runtime.**
- **Abstraction** is a powerful force in simplifying program paths.

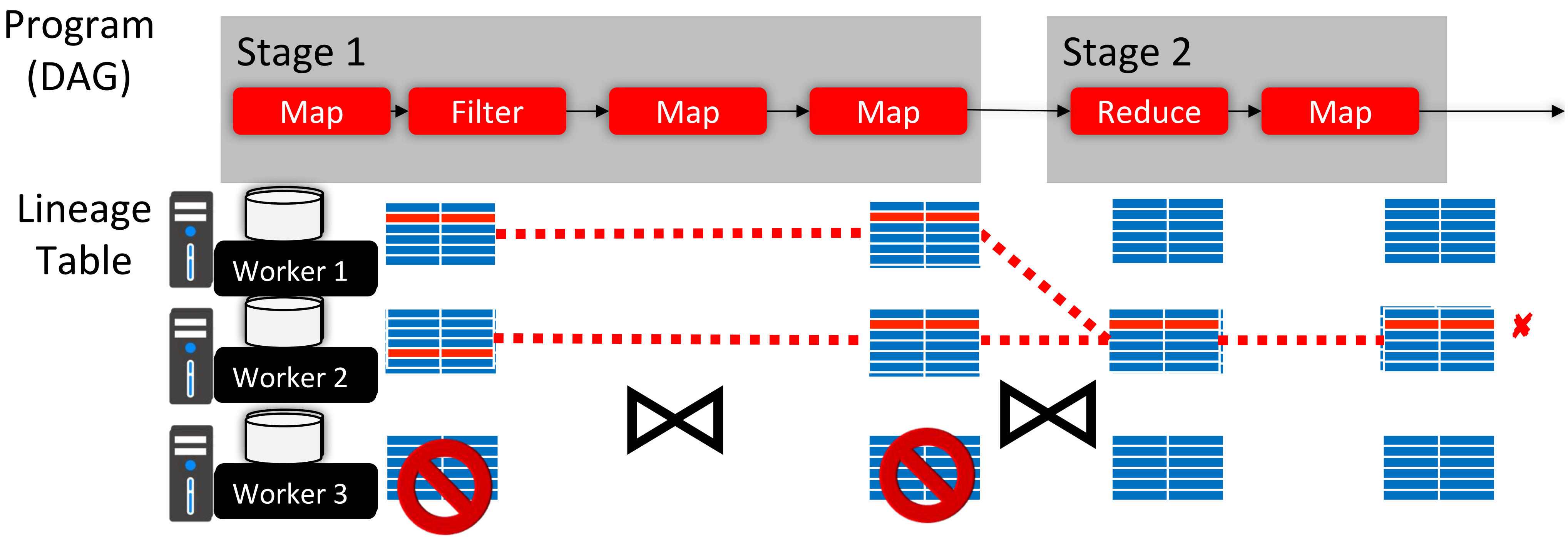
Enabling interactive debugging requires us to re-think a traditional debugger

- Pausing the entire computation on the cluster could reduce throughput
- It is clearly infeasible for a user to inspect billion of records through a regular watchpoint

BigDebug: Interactive Debug Primitives for Big Data Analytics [ICSE 2016]



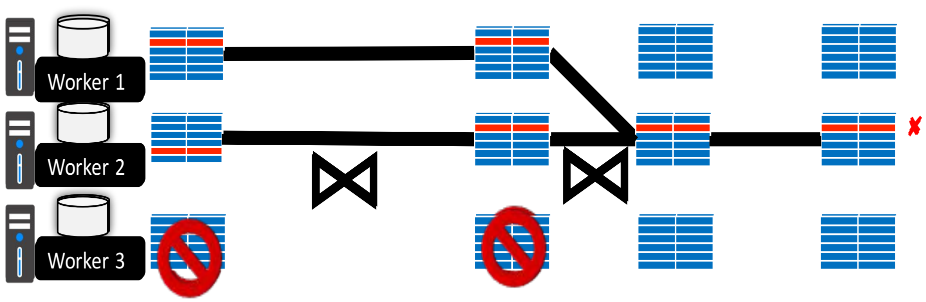
Titian: Data Provenance for Apache Spark [VLDB 2016]



BigSift: Automated Debugging of Big Data Analytics [SoCC 2017]

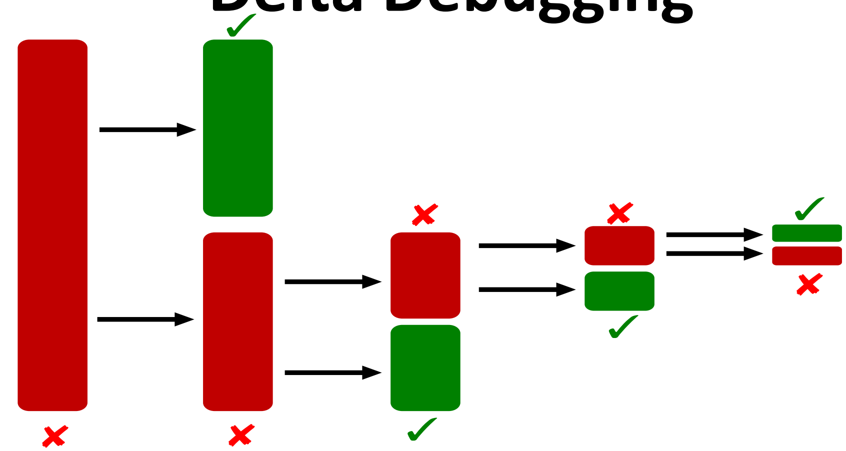
Input: A Program, A Test Function   Output: Faulty Records

Titian Data Provenance



⊕

Delta Debugging



Test Predicate Pushdown

Prioritizing Backward Traces

Bitmap based Memoization

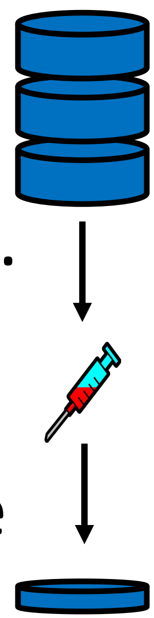
Results on Debugging of Big Data Analytics

- BigDebug enables **interactive debugging** and **repair**, while retaining the **scale-up** property. It poses at most **34% overhead** [ICSE 2016].
- Titian's **data provenance** is **orders of magnitude faster** than alternatives [VLDB 2016].
- BigSift **automatically** finds bugs **66X faster than delta debugging**. It takes 62% less time to debug than the original job's run [SoCC 2017].

Why is Testing Big Data Analytics Challenging?

Option 1: Sample Data

- random sampling,
- top n sampling
- top k% sample, etc.



Limitations:

- Low code coverage
- Or increased local **testing time**

Option 2: Traditional Testing

- 700 KLOC for Apache Spark

Limitations:

- **Symbolic execution** without abstraction would **not scale**.

BigTest: White-Box Testing of Big Data Analytics [ESEC/FSE 2019]

Relational skeleton
700 KLOC Spark

Abstract

Logical Specifications

$\texttt{JOIN}: \exists tR, tL: cR \in CR \wedge cL \in CL \wedge cR(tR) \wedge tR,key = tL,key \wedge cL(tL)$

User defined func

Extract

Symbolic Execution

Path Constraint	Effect
T.split(",").length ≥ 1 ∧…∧ V2 = "ERROR" …	"\x00", "Palms"

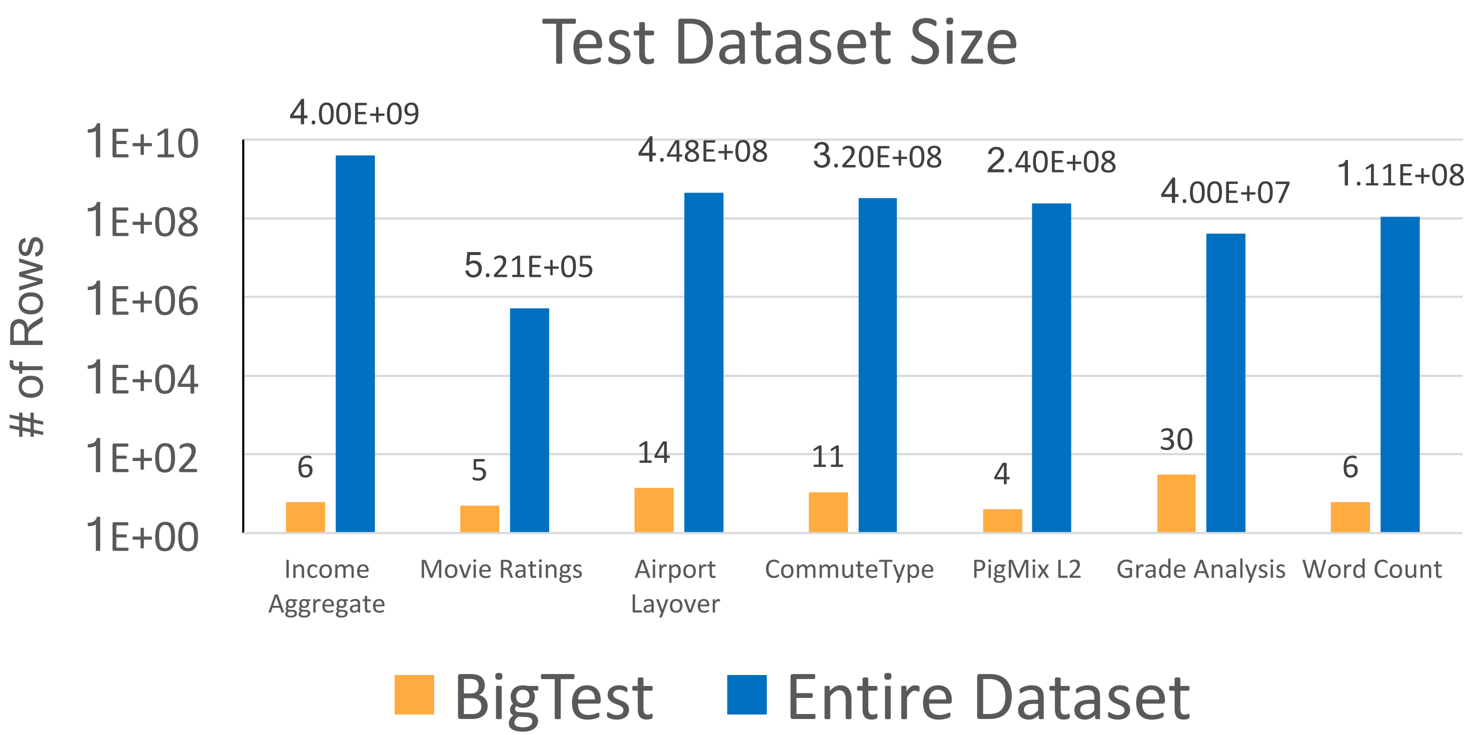
String operations

Model

String Constraints

```
Z.split(",")[1]="Palms" ∧
Z.split(",").length >1 ∧
T.split(",")[1] = Z.split(",")[0] ∧
T.split(",").length >1 ∧ …
```

Test Size Reduction



BigTest reduces tests by 10^5X to 10^8X, achieving 194X testing speed up.

Outline: Making a Case for Software Engineering for Data Analytics (SE4DA)

① Shift to **data-centric SW development** — **Studies: Data Scientists**

② Differences between **traditional SW vs. data-centric SW dev process**

③ **Debugging & testing** for **big data analytics** — **Tools**

④ **Open problems** in SE4DA

Part 4. Roadmap for Accelerating Data-Centric Development

DA4SE

SE4DA

2004 2008 2014 2019 2022 2025

GitHub

stackoverflow

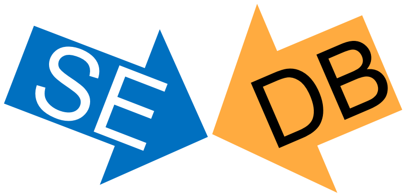
Data scientists in software teams

SE **collaborates** with Systems, ML, and DB to design tools

SE tools are **co-developed** with runtimes & frameworks

Insight 1: Debugging data analytics requires both data and code analysis.

How to **define a bug** based on the properties of **both data** and **code**?



Data X-Ray
[SIGMOD'15]

Bug Patterns
[SIGPLAN 2004], etc.

How to **repair** both code and data errors?

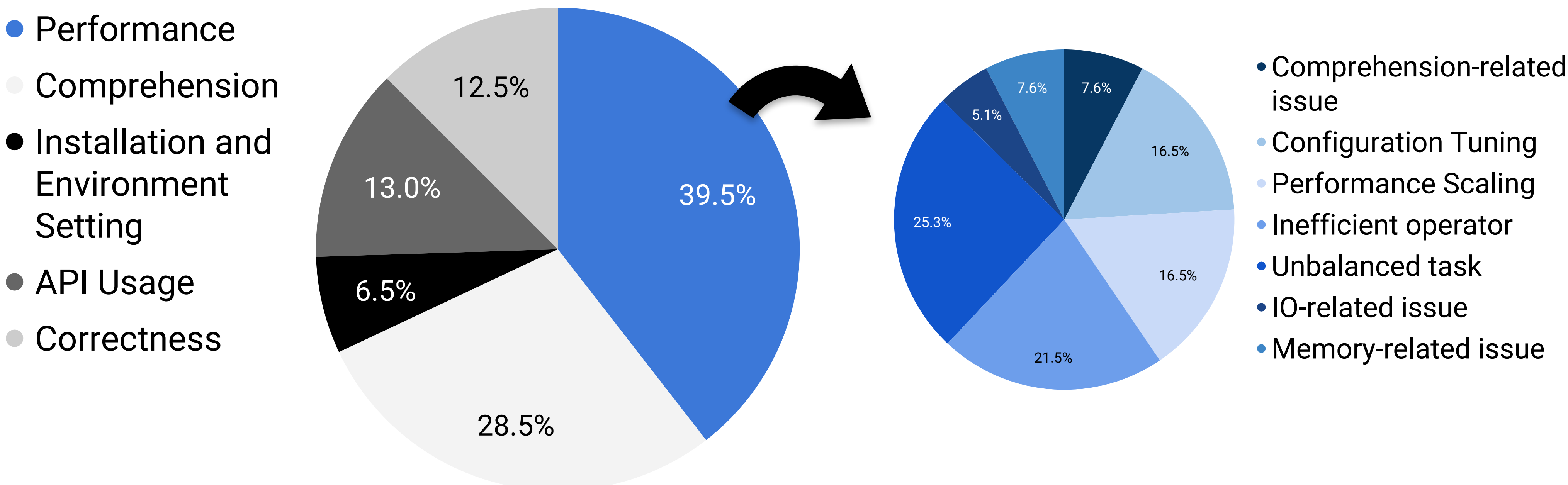
Data Cleaning
[VLDB'01] [VLDB'15] [SIGMOD '15] [SIGMOD'10]

Data Repair
[VLDB'11] [SIGMOD '14]

Data Wrangling
[CHI'11]

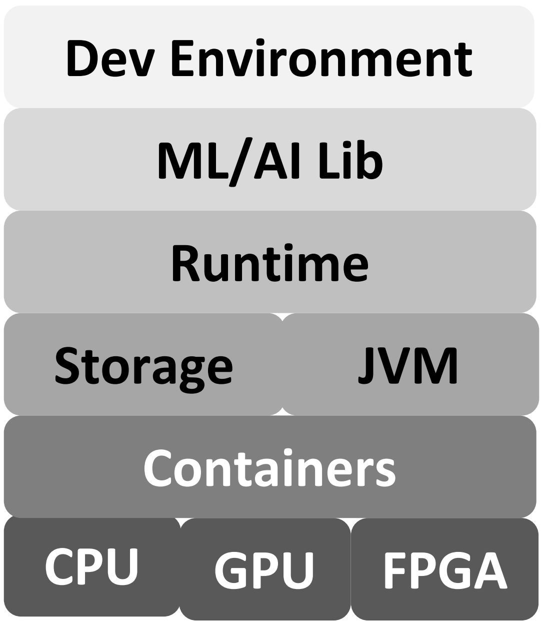
Program Repair
[ICSE'09] [ICSE'13], etc.

Insight 2: Performance debugging is a pain point.



Manual inspection of top 200 Spark related posts from Stack Overflow

Insight 2: Performance debugging requires visibility of system stack, code, and data.



How to estimate performance based on data size?

Ernest [NSDI'16]

How to optimize query performance using a cost model?

Neo [VLDB'16]

How to debug computation and data skews?

Skewtune [SIGMOD'12]

PerfDebug [SoCC'19]

How to identify the cause of bottlenecks?

Causal Profiling [SOSP'15]

Causal Monitoring [SOSP'15]

Insight 3: We must relax the strict notion of an incorrect behavior and the root cause.

How to **specify oracles** for data-centric software**?**

Metamorphic relations are simple or hard to define

Metamorphic Testing
[1998]

DeepTest
[ICSE 2018]

DeepConcolic
[ASE 2018]

DeepHunter
[ISSTA 2019]

How to **quantify importance** when debugging faulty inputs for data analytics?

LIME
[KDD'16]

Influence Function
[ICML'17]

Training Set Debugging
[AAAI'18]

MODE
[ESEC/FSE'18]

Lamp
[ESEC/FSE 2017]

Conclusion: Hope for **Software Engineering for Data Analytics (SE4DA)**

We are at an **inflection point.** SE4DA is underserved.

Progress has been made in SE4DA by **re-thinking software engineering** for big data analytics.

We can together work on **open problems in SE4DA**.

SE4DA: AI, Big Data, and ML need awesome SE tools

Diagnose

- ✓ Debugging
- ✓ Intelligent sampling and testing
- ✓ Root cause analysis

Fix

- ✓ Data cleaning

Optimize

- ✓ Performance analytics
- ✓ Code analytics

Questions?