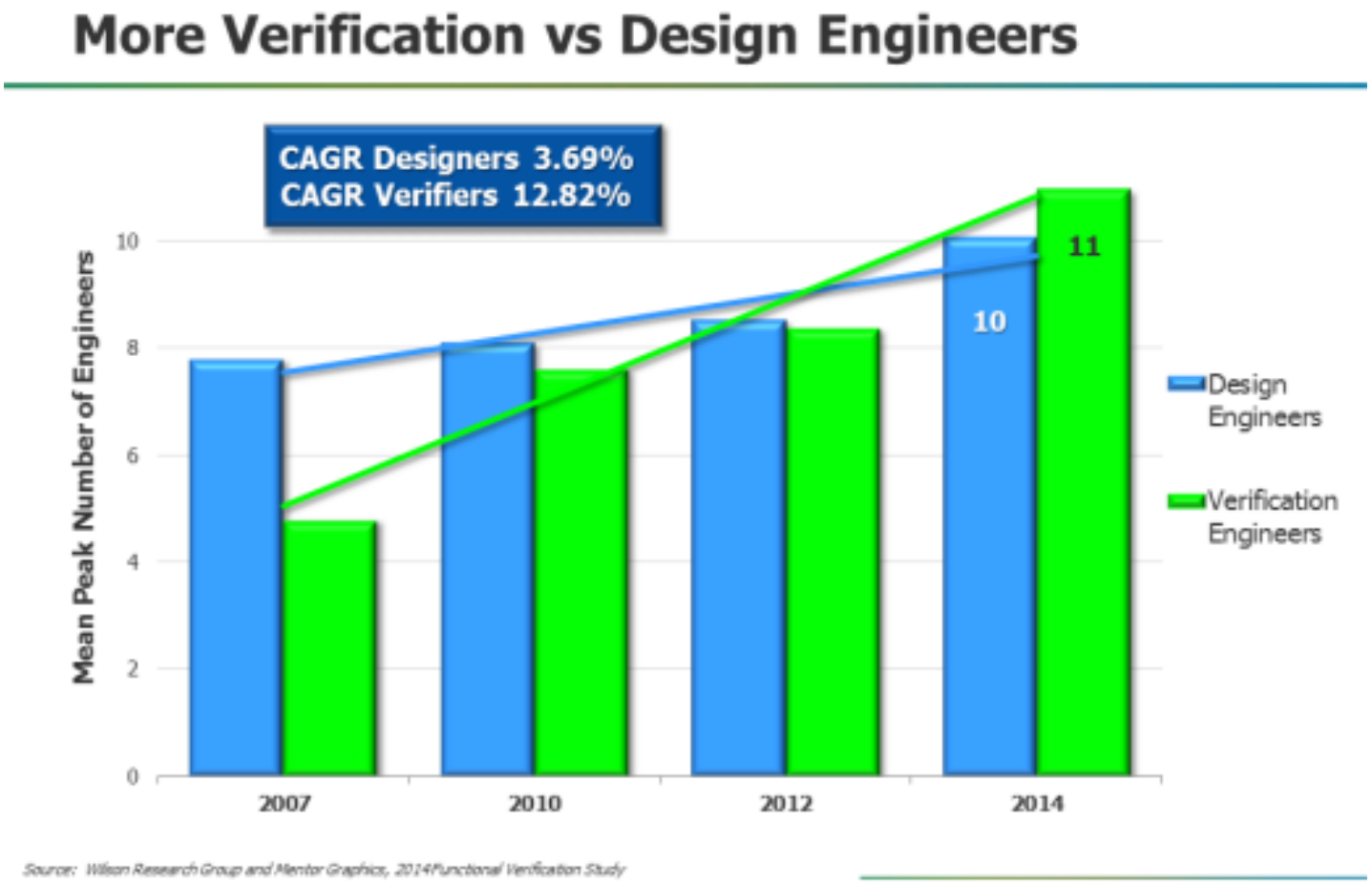
# A Topological Approach to Hardware Bug Triage

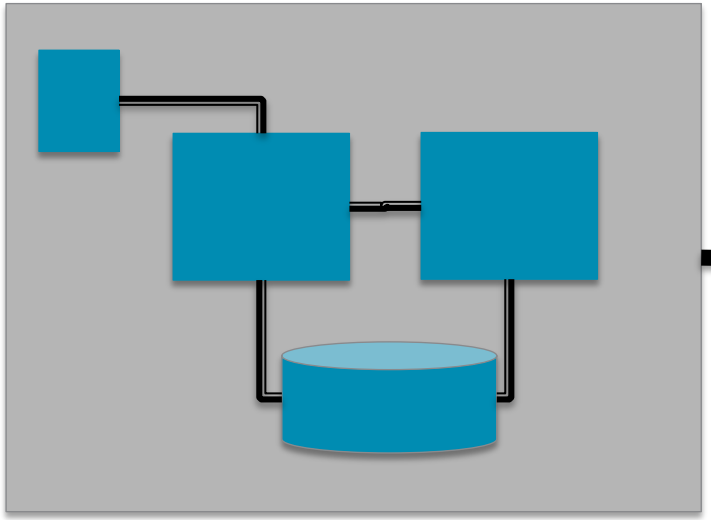
Rico Angell, Ben Oztalay, and Andrew DeOrio

University of Michigan

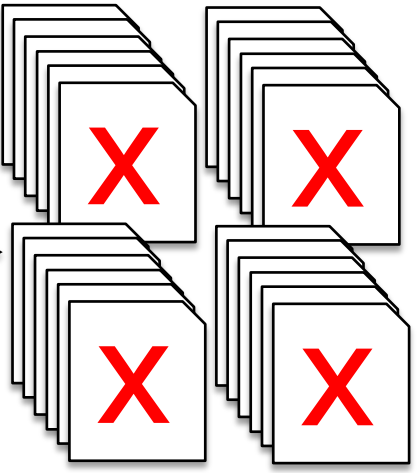
# Increasing Complexity



# Post-Silicon Validation

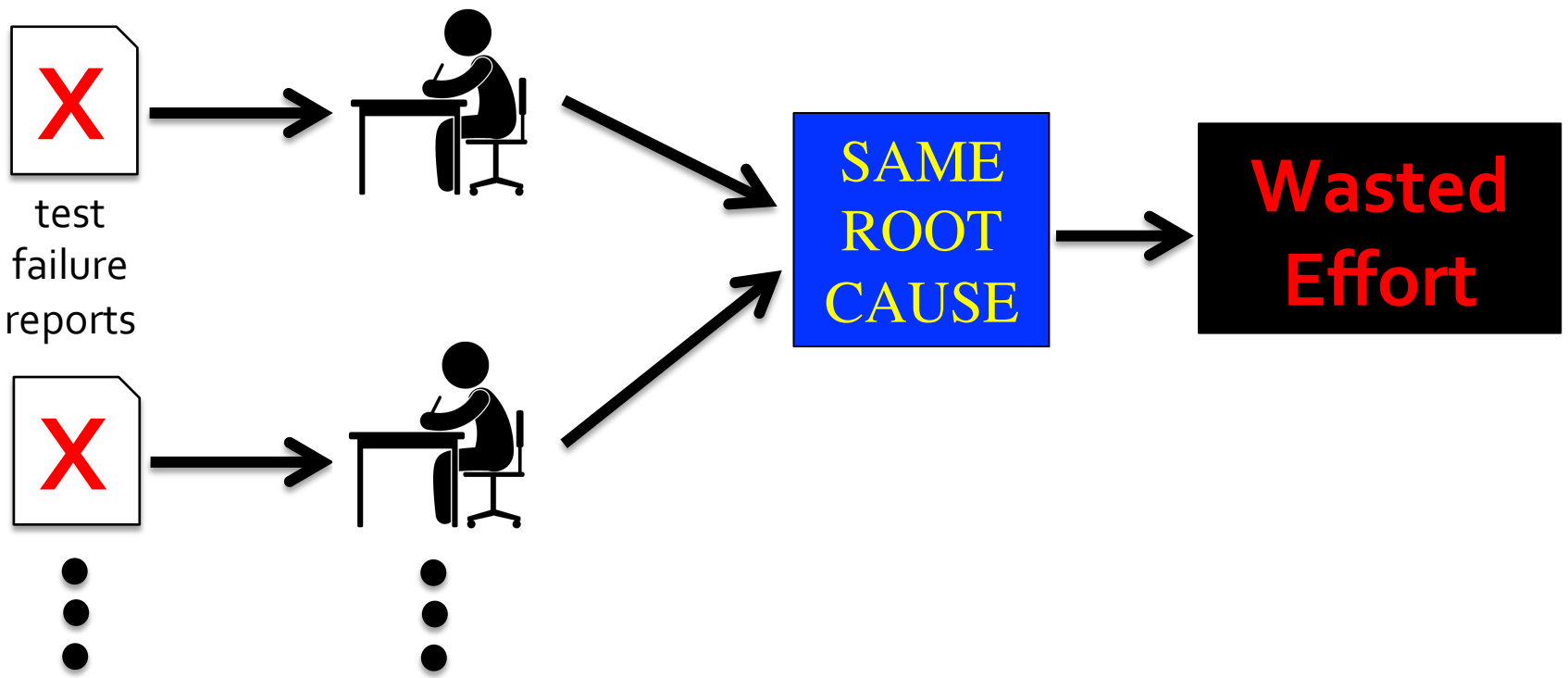


Post-silicon platform




Many failure reports
Goal: debug root causes

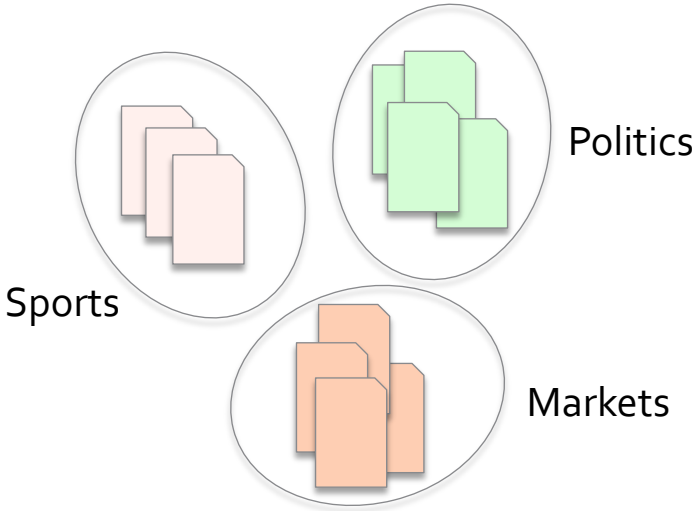
# Duplicated Effort



# Machine Learning Background

- Goal: Create a model by learning from input data
- Supervised requires labeled training data to create the model
- Unsupervised creates model from unlabeled input data

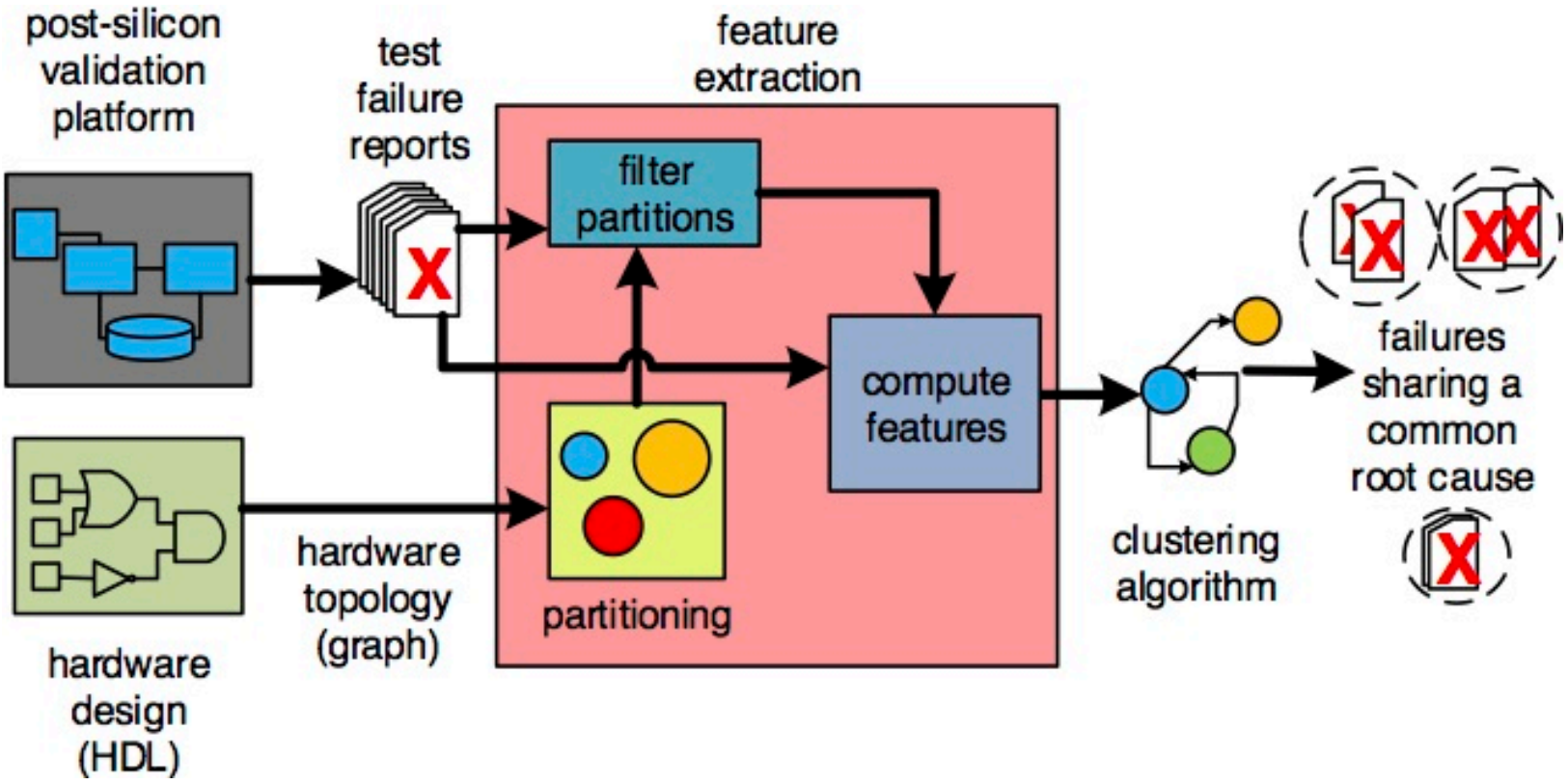

Classification of News Articles



# Contributions

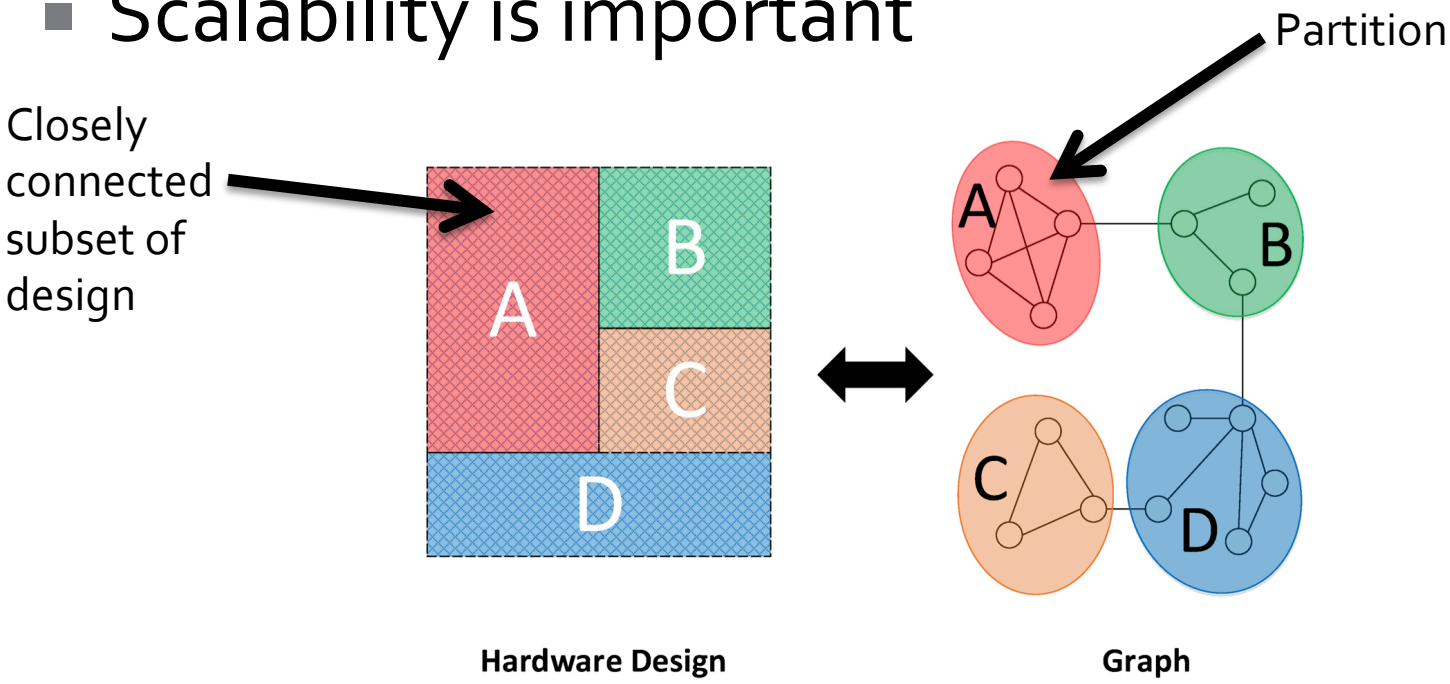
- Novel approach to choosing features for clustering failure reports
- New metric, Unique Debugging Instances (UDI), for measuring verification efficiency with respect to triaging failure reports

# Our solution



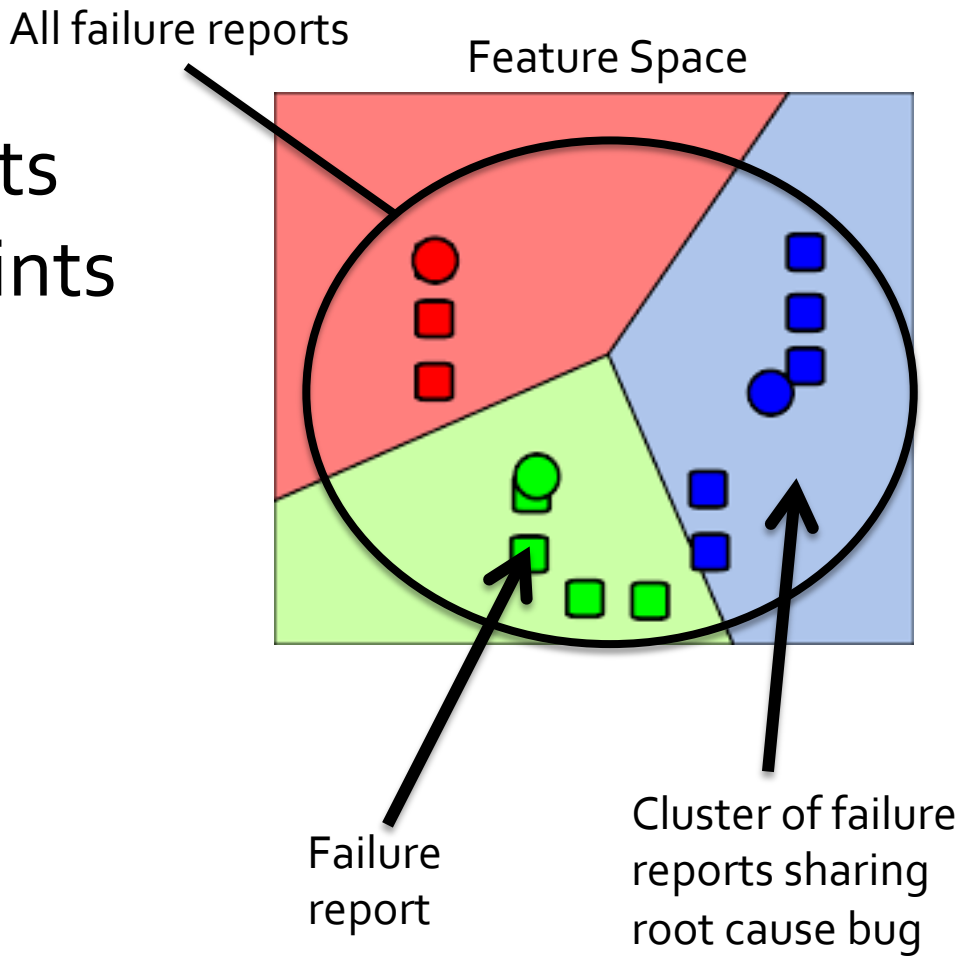
# Feature Extraction: Partitioning

- How do we describe failure reports to a Machine Learning algorithm? → Features
- Use ParMETIS
- Scalability is important



# *K*-means

- Input: *k* and set of points
- Output: *k* groups of points
- NMI: metric measuring mutual dependence
- NMI of 0 → Entirely Imperfect
- NMI of 1 → Perfect

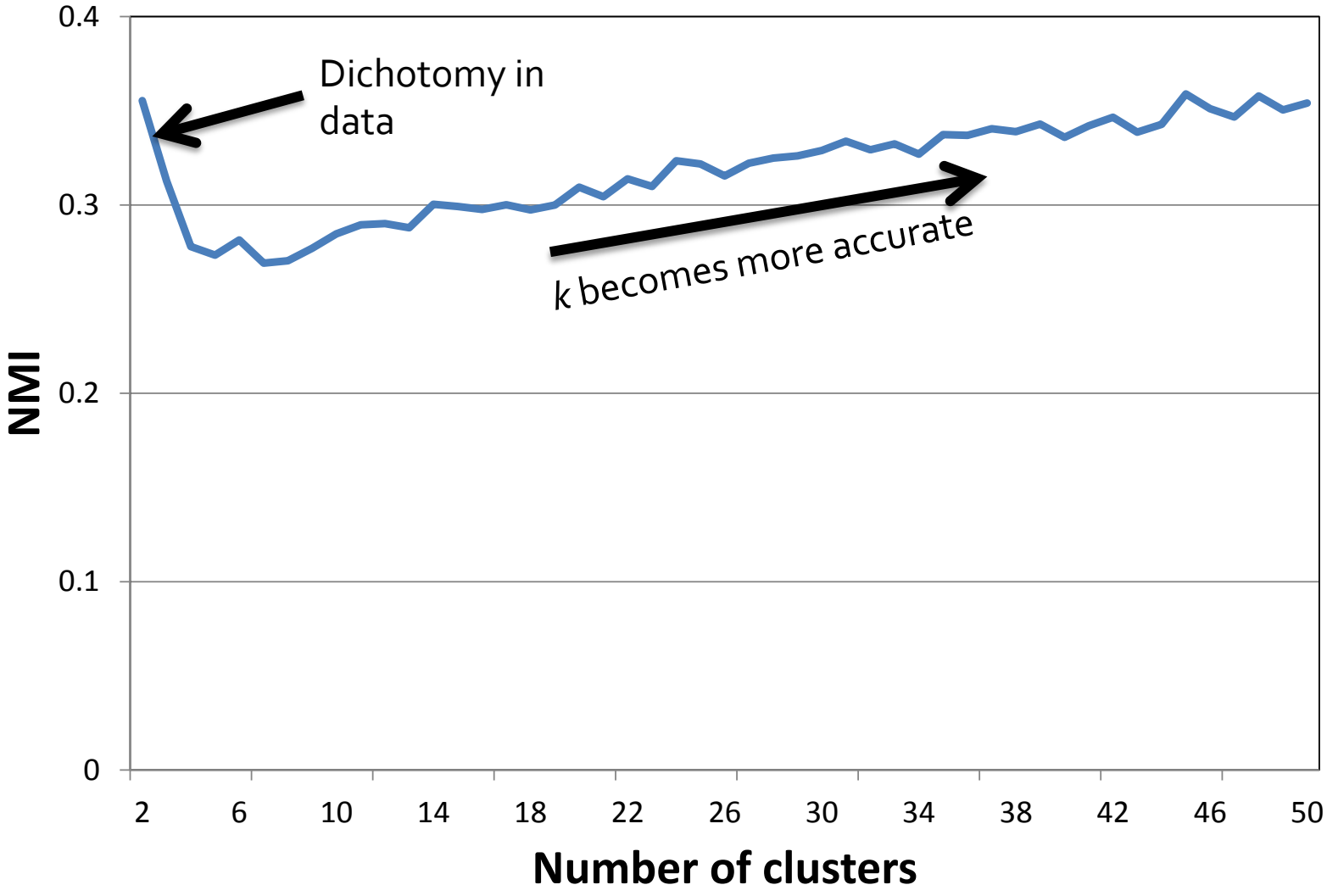


# Experimental Setup

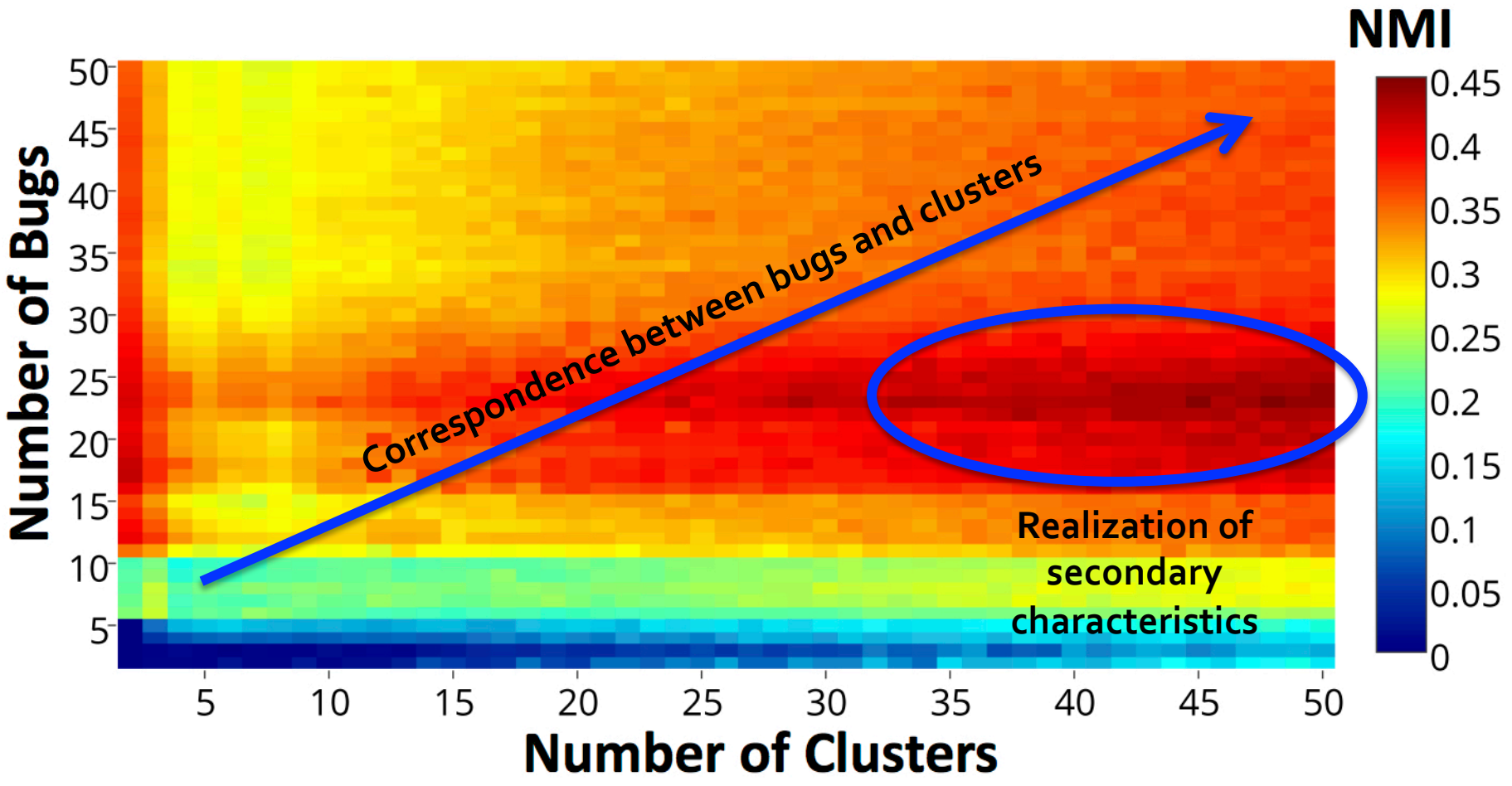
- Industrial-size OpenSPARC T2
- Bug Injection:
  - Force control signal to 1
  - 50 total bugs, evenly spread across 10 modules
- Injected a bug at cycle 10,000
- At cycle 10,100, record the control signal values from 10 modules
- 5000 simulations

OpenSPARC™

# Experimental Results



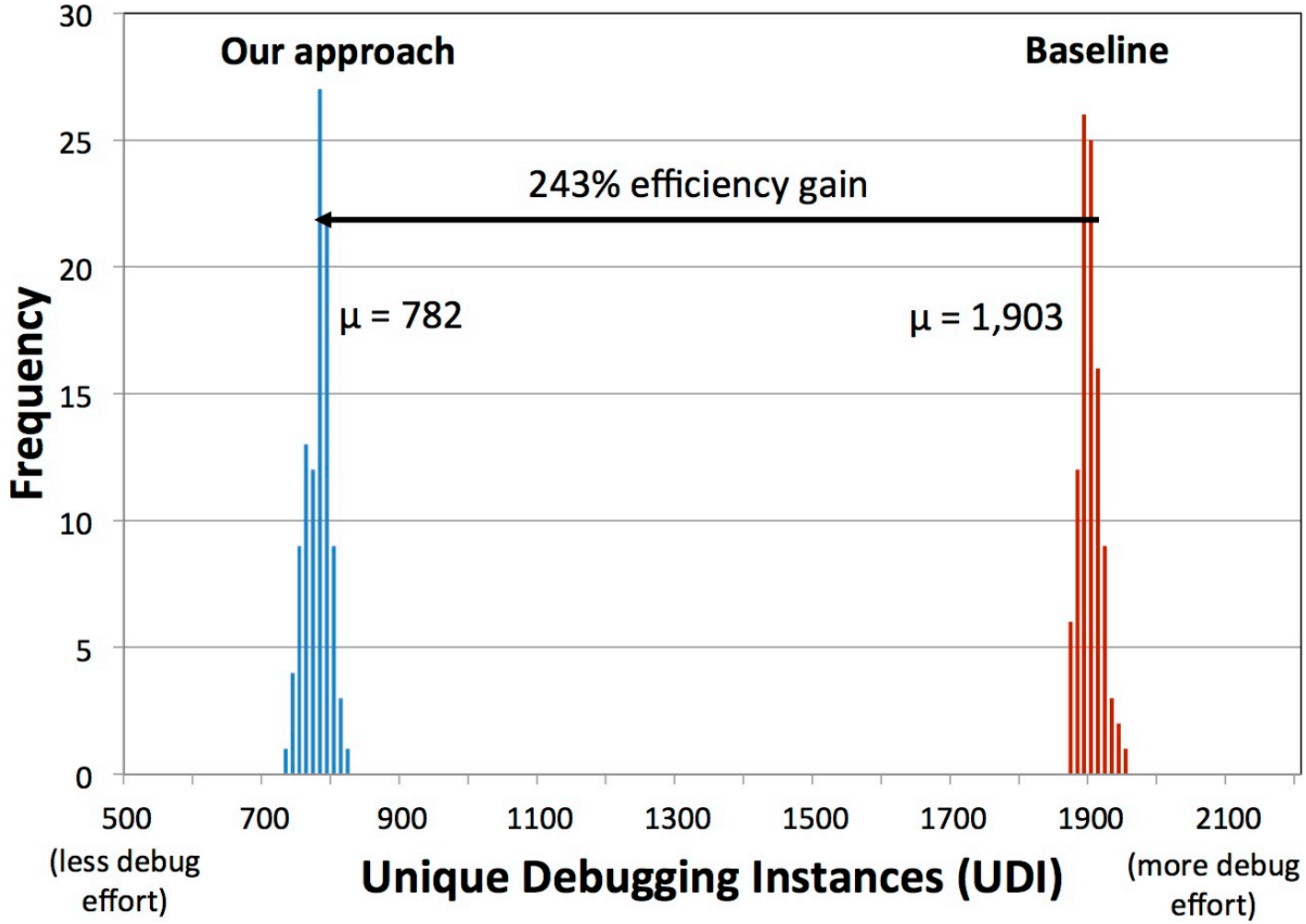
# Experimental Results



# Unique Debugging Instances (UDI)

- Goal: A metric to measure verification efficiency
- Measured by aggregating the number of unique root causes represented by failure reports in each cluster
- Low UDI → Triaged well
- High UDI → Triaged poorly

# Impact on Verification Efficiency



# Related Work

- Partitioning graph representations of software systems [Mancoridis '99]
- Triaging software bug reports using supervised Machine Learning [Cubranic '04]
- Utilizing assertions and SAT-based debugging data to automatically triage failure reports [Veneris '14]

# Conclusions & Future Work

- Proposed a method to group failure reports based on root cause
- Our approach improves verification efficiency by 243% as measured by Unique Debugging Instances
- Future work:
  - Exploration of better features
  - Ways of handling the manifestation of multiple bugs during a single execution
  - Other Machine Learning algorithms