Dependability for Computer Systems meets Data Analytics

Saurabh Bagchi
School of Electrical and Computer Engineering
Department of Computer Science
Director, Center for Resilient Infrastructures, Systems and Processes (CRISP)
Purdue University


Presentation available at: *engineering.purdue.edu/dcsl*

PURDUE UNIVERSITY

Roadmap

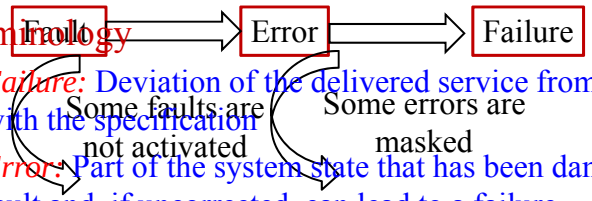
- Dependability basics
  - System design principles
  - Terminology and basic approaches
- Challenges and current results from
  - Embedded and mobile networks
  - Computational genomics
- Dependability in a cellular network [SRDS-16, Middleware-17, Crowdsense-17, Eurosys-18 (under submission)]
- Dependability in approximate computing [PACT-15, CGO-17, ASPLOS-18 (under submission)]

PURDUE UNIVERSITY

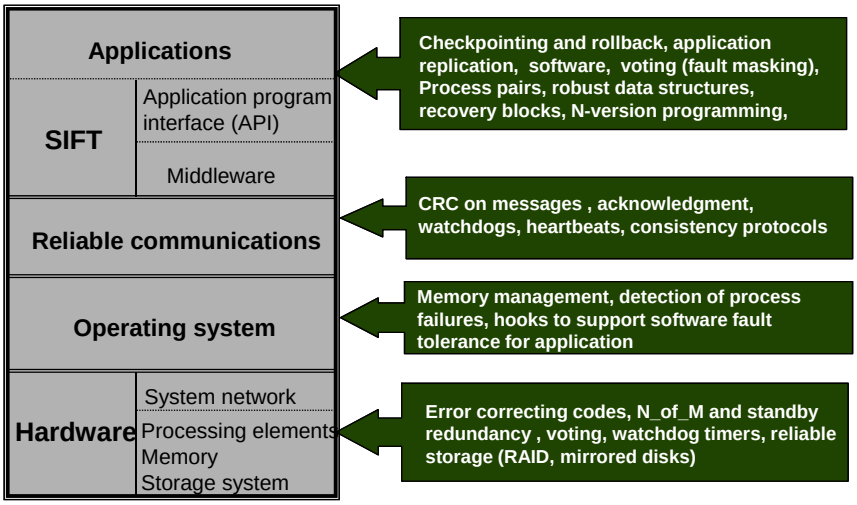
What is Dependable Computing?

- Dependability: Property that the computer system meets its specification despite the presence of faults
  - Faults can be due to natural causes (bugs, defects in hardware), or
  - Maliciously induced (attacks from external or internal sources)

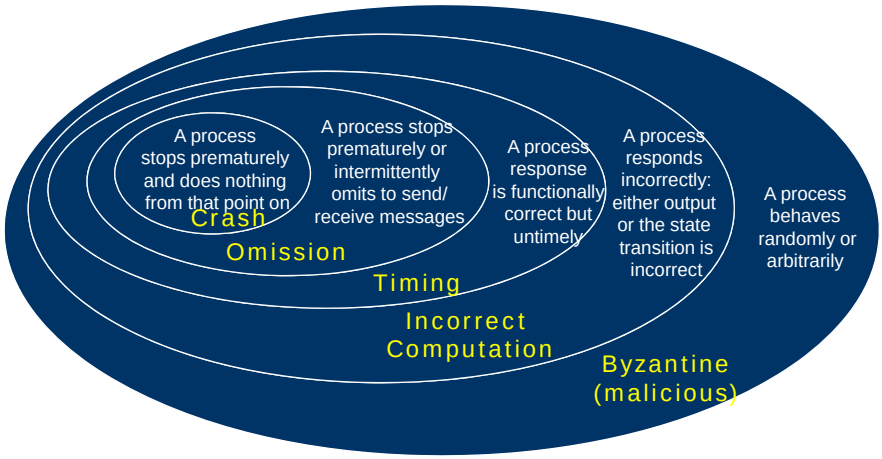
- Terminology
  - *Failure:* Deviation of the delivered service from compliance with the specification
  - *Error:* Part of the system state that has been damaged by the fault and, if uncorrected, can lead to a failure.
  - *Fault:* The adjudged or hypothesized cause of an error



How do We Achieve the Objectives?



Your Failure is Not My Failure

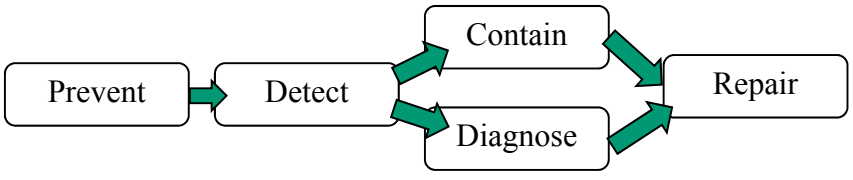


3 Most Important Things in Dependability

1. Redundancy
2. Redundancy
3. Redundancy

1. How much redundancy?
2. Where to put the redundancy?
3. How to validate the redundant operation?

Means of achieving dependability

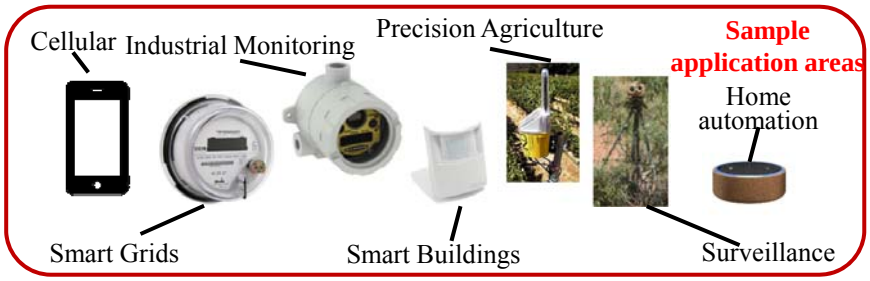


Roadmap

- Dependability basics
  - System design principles
  - Terminology and basic approaches
- Challenges and current results from
  - Embedded and mobile networks
  - Computational genomics
- Dependability in a cellular network [SRDS-16, Middleware-17, Crowdsense-17, Eurosys-18 (under submission)]
- Dependability in approximate computing [PACT-15, CGO-17, ASPLOS-18 (under submission)]

PURDUE UNIVERSITY

Embedded and Mobile Networks



- Challenges
  - Systems have fundamental resource constraints and are often deployed in unprotected or uncertain environments
  - Constraints include: Energy, Bandwidth, Untrusted nodes, Disconnected networks
  - Nodes have low-end microcontrollers and lightweight OS without security protection

PURDUE UNIVERSITY

Embedded and Mobile Networks

- Opportunities
  - Fewer modes of user interaction
  - Single purpose
  - Dense deployment
- Some active research directions

1. Distributed monitoring for mobile networks (such as, vehicular networks)
2. Cellular radio access problems
3. Record and replay for problem diagnosis

PURDUE UNIVERSITY

A Significant Result

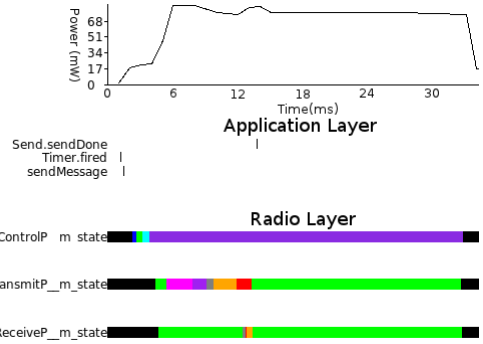
- TARDIS: A software-only approach for deterministic record and replay of embedded nodes[1, 2]
- Basic idea of any record-and-replay mechanism:
  1. Record detailed execution trace as application is executing on node in situ
  2. Use the trace to replay the execution and debug the problem
- Records all sources of non-determinism and compresses different traces in a domain-specific manner
- *Result*: Log growth = 1.5 KB/s – 88% reduction ⇨ Flash on Amazon Echo Dot can hold 1 month of blackbox information
- Need fine-grained monitoring information to debug subtle timing bugs

[1] M. Tancreti, V. Sundaram, S. Bagchi, and P. Eugster, "TARDIS: Software-Only System-Level Record and Replay in Wireless Sensor Networks," IPSN, pp. 286-297, 2015.
[2] M. Tancreti, M. S. Hossain, S. Bagchi, and V. Raghunathan, "AVEKSHA: A Hardware-Software Approach for Non-intrusive Tracing and Profiling of Wireless Embedded Systems," SenSys, pp. 288-301, 2011.

PURDUE UNIVERSITY

A Significant Result

- Need fine-grained monitoring information to debug subtle timing bugs



- Given rise to a community building debugging tools for low-end embedded devices and networks [Minerva-Sensys13, Flocklab-IPSN13, D2-RTSS13, Thiele-et al.-EWSN17, …]

Data Analytics and Dependability (In Embedded and Mobile Networks)

Current state of practice	Very limited
Level of sophistication of data analysis needed	Simple

- Examples of promising convergence of the two

1. Learn the pattern of traffic between two interacting devices – use that for anomaly detection
2. Learn the pattern of sensed values in a region or in a time period – use that for optimal compression

Roadmap

- Dependability basics
  - System design principles
  - Terminology and basic approaches
- Challenges and current results from
  - Embedded and mobile networks
  - Computational genomics
- Dependability in a cellular network [SRDS-16, Middleware-17, Crowdsense-17, Eurosys-18 (under submission)]
- Dependability in approximate computing [PACT-15, CGO-17, ASPLOS-18 (under submission)]

PURDUE UNIVERSITY

Computational Genomics

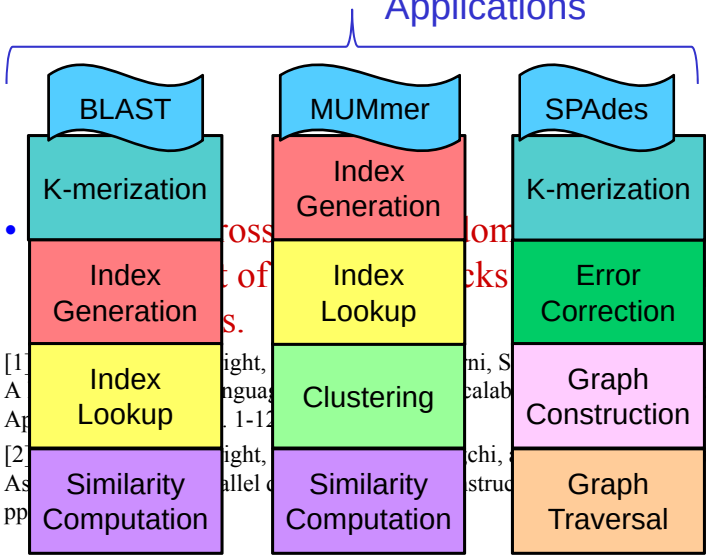
- Phenomenal growth in amount of genomic (and epigenomic) data which has to be processed for insights
- Sequencing instruments are error prone – redundancy in reads and algorithmic tricks have to correct for errors
- Common dependability challenges:
  - Scalability, scalability, scalability
  - Fragile code bases
  - Lack of efficient cyberinfrastructures

PURDUE UNIVERSITY

Computational Genomics

- Opportunities
  - Some emerging standardized building blocks
  - Some embarrassingly parallel parts to many algorithms
  - Good metrics for judging accuracy of results and benchmarking
- Some active research directions
  - *Error correction*: Methods to correct various kinds of errors in sequencing reads
  - *Applying standard distribution techniques*: From graph algorithms or string matching for example
  - *Standardized workflows and testing*: Reduces incompatibilities among multiple software packages and allows us to judge quality of results

A Significant Result

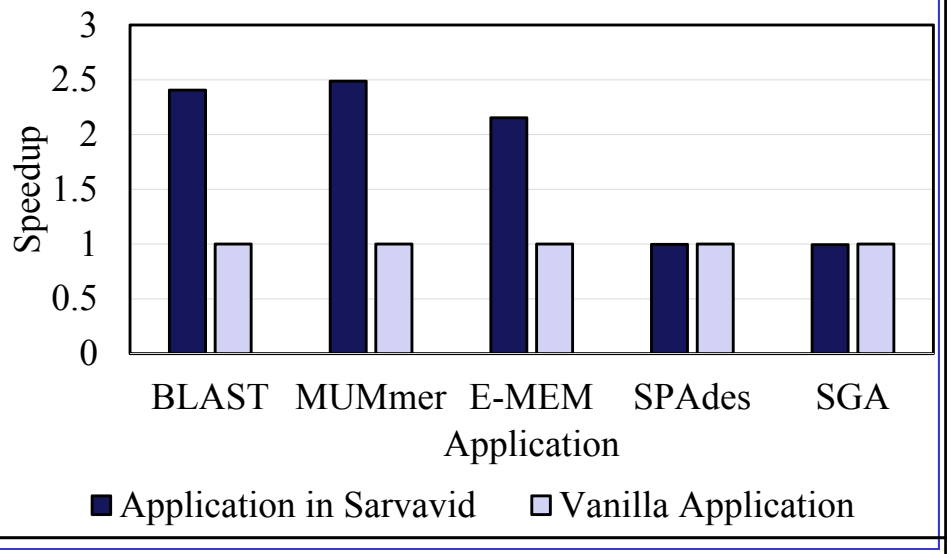


- ross lom ontaining
  t of cks hich become
  s.

[1] ight, ni, S terji. "SARVAVID: A nguag calab enomics Ap . 1-12

[2] ight, chi, lable Genomic As allel struc mers," ACM BCB, pp

Performance Comparison of SARVAVID over Vanilla Applications



Data Analytics and Dependability (In Computational Genomics)

Current state of practice	Primitive
Level of sophistication of data analytics needed	High but needs to be scalable as well

- Examples of promising convergence of the two

1. Learn the pattern of errors in a particular instrument and particular configuration to decide how best to correct errors
2. Learn the loading pattern of different tasks so as to optimally distribute load
3. Complex multi-dimensional pattern in epigenomic markers to tell why some genes are repressed

Roadmap

- Dependability basics
  - System design principles
  - Terminology and basic approaches
- Challenges and current results from
  - Embedded and mobile networks
  - Computational genomics
- Dependability in a cellular network [SRDS-16, Middleware-17, Crowdsense-17, Eurosys-18 (under submission)]
- Dependability in approximate computing [PACT-15, CGO-17, ASPLOS-18 (under submission)]

PURDUE UNIVERSITY

Motivation

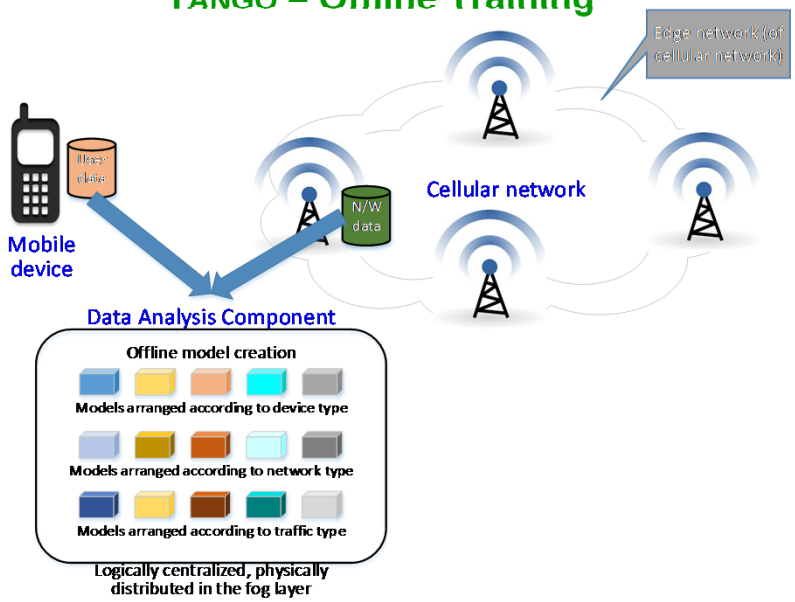
- Mobile devices consider cellular network as "dumb pipe"
  - Connectivity is not always reliable
  - Can only react after connectivity is degraded
- With cooperation between device and network, failures can be managed better

PURDUE UNIVERSITY

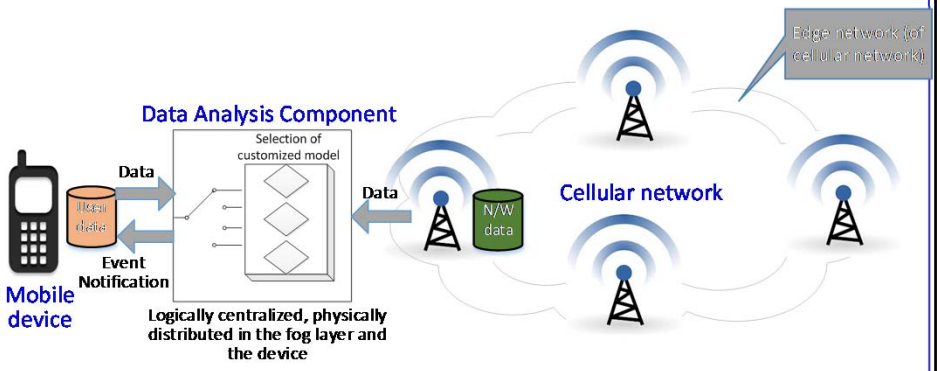
TANGO

- Framework that enables real-time cooperation between mobile device and cellular network
  - Many services can be built on top
- Service performs real-time data analysis to alert device of certain events
  - Alert streaming application before user enters congested area
- Device/application decides how to mitigate problem
  - Streaming application: pre-cache more content
  - Switch to a different carrier

TANGO – Offline Training



TANGO – Online Operation



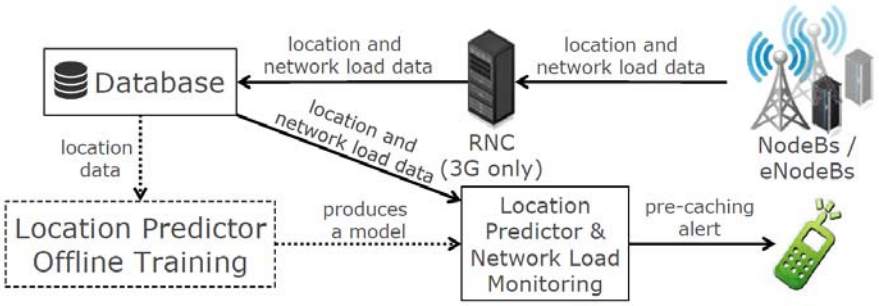
Pre-caching Service

- Sends alert to application before connectivity degradation
  - Predicting user location
  - Monitor network load on predicted locations
- Applications
  - Audio/Video streaming
  - GPS navigation
  - Web browsing

Streaming Application – Current Practice

- Mobile streaming applications limit download rate
  - To conserve user's bandwidth and energy
  - Usually by limiting buffer size
- Connectivity degradation (e.g., due to congestion) results in playback disruption
- Ideal buffering strategy:
  - Small buffer when connectivity is good
  - Large buffer when connectivity is bad
- With current systems, can only react after connectivity degraded
- With TANGO, application increases buffer size and/or reduce bit-rate

Pre-caching Service – Overview



Offline phase
- Location predictor training

Online phase
- User location prediction
- Network load monitoring

AppStreamer: Reducing mobile applications' storage requirements through predictive streaming

PURDUE UNIVERSITY

Motivation

- Demand for storage has been growing faster than storage capacity of smartphones
  - 2016 survey found amount of content stored grew 55% over 10 months
  - Many popular games are 1-4 GB
  - Smartphone capacity roughly doubles every 2.5 years
  - Users need to uninstall some apps to make space for new apps
- Current practice: all or none
  - Installation also takes long time

PURDUE UNIVERSITY

Available Solutions to Storage Crunch

- Storing apps on the cloud
  - Unacceptable delay even with good connectivity
  - Application usage is difficult to predict
- Running apps on the cloud and streaming only video (thin client)
  - ~100ms input latency unacceptable in interactive applications such as games
  - High bandwidth consumption

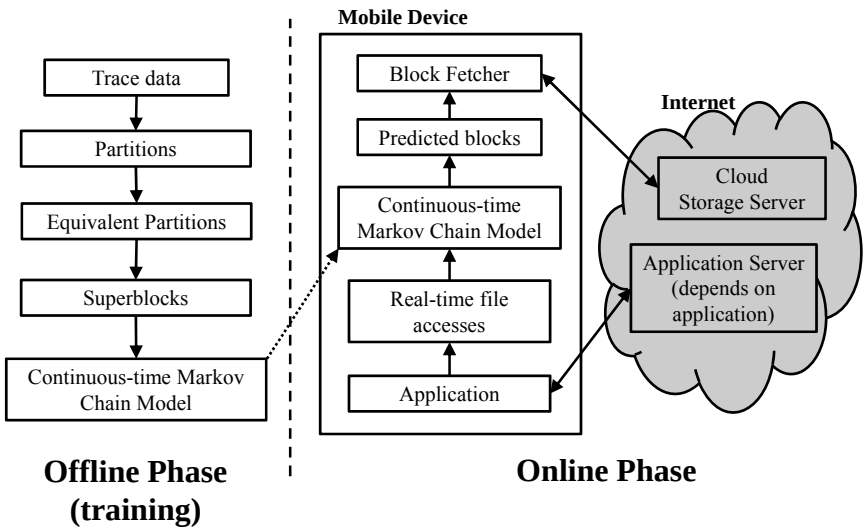
PURDUE UNIVERSITY

Solution Approach

- ***Intuition:*** Large apps only access small portion of their files
- AppStreamer: predictive application streaming
  - Collect data about file accesses from multiple users
  - Build a model that predicts future file accesses based on recent behavior
  - Bring in relevant file blocks before they are needed by the application

PURDUE UNIVERSITY

AppStreamer Overview



Key Requirements

- Work for all applications without source code
  - Transparent to the applications
  - Blocks need to be on local storage before they are read
- Does not lower user experience by introducing delays
  - Accurate model (high recall)

AppStreamer Components

1. File access capture (offline & online phases)
2. File access prediction model (online phase)
3. Data block fetcher (online phase)

File Access Prediction Model (1)

- Requirements
  - Temporally based
  - Low overhead
  - Probabilistic
- Continuous-Time Markov Chain (CTMC)
  - Captures transition between states & duration spent
  - Need to define "state"

PURDUE UNIVERSITY

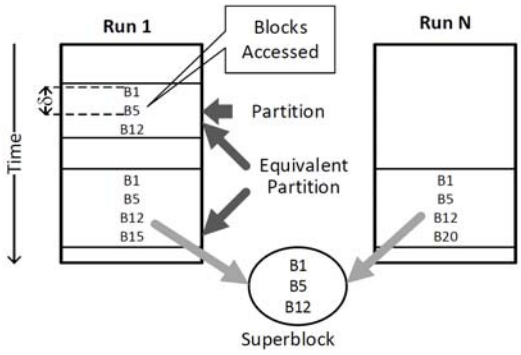
File Access Prediction Model (2)

- Idea 1: block = state
  - 1 GB game has 250,000 blocks. Need to store $250{,}000^2 = 62.5$ billion transition probabilities.
- Idea 2: read call = state
  - Similar reads considered distinct
  - One large read can be separated into multiple reads in many ways

PURDUE UNIVERSITY

Block Grouping (1)

- Input: File access traces from multiple runs
- Three steps
  1. Partition
  2. Equivalent Partition
  3. Superblock



Markov Chain

- State = superblock

- Prediction done using depth-first search
  - Bounded by probability (p_{stop}) and lookahead time (L)
  - User playing speed taken into account by adjusting L
- Blocks with probability $\geq p_{download}$ are put in download queue
- Single Markov model can capture different control paths taken by different users

Data Block Fetcher

- Speculative blocks
  - Fetches predicted blocks in background
- Application-requested blocks (cache miss)
  - Read system call will block
  - Fetches immediately from cloud storage server

PURDUE UNIVERSITY

Evaluation: Dead Effect 2

- First-person shooter
- Single player
- Gameplay divided into linear levels (maps)
- 1.09 GB in size

PURDUE UNIVERSITY

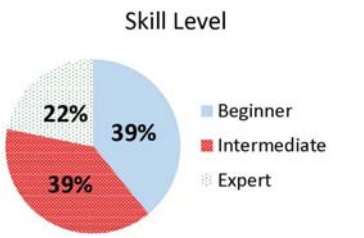
User Study Setup

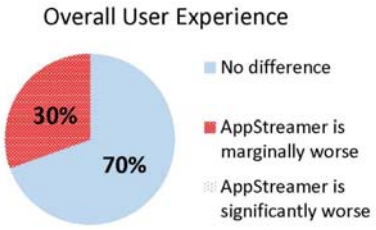
- Cloud storage server with 17.4 Mbps network speed
- Nexus 6P running Android 6.0.1
- User plays game for 20-30 minutes
  - With and without AppStreamer
- Fill questionnaire to report user experience

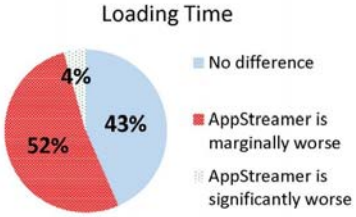
PURDUE UNIVERSITY

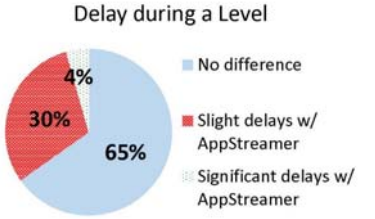
Dead Effect 2 User Study Results (1)

- 23 participants





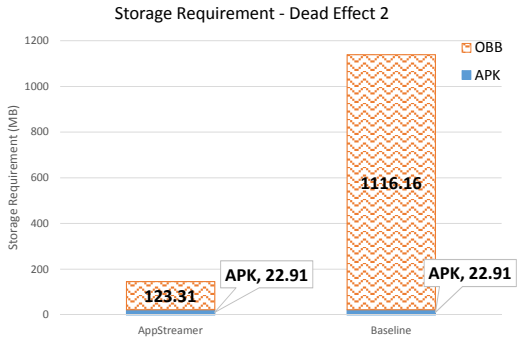




PURDUE UNIVERSITY

Dead Effect 2 User Study Results (2)

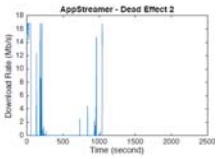
- 336.2 KB cache miss per run on average
  - Translates to 0.15 second of delay (out of ~30 minutes)
- 87.16% lower storage requirements



Comparison with Cloud Gaming

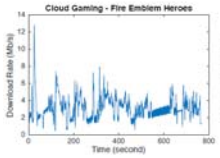
- GamingAnywhere [2]
  - Android emulator Nox on cloud server
  - GamingAnywhere client on smartphone

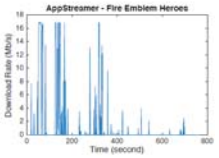
3.01 Mbps



706 Kbps

3.20 Mbps





745 Kbps

[2] Huang, Chun-Ying, et al. "GamingAnywhere: an open cloud gaming system." In *Proceedings of the Fourth ACM Multimedia Systems Conference*. ACM, 2013.

Concluding Insights

- Cooperation between mobile devices and cellular network can enable highly reliable operation
  - Reliable streaming of content
  - Reducing streaming of mobile applications
- Such cooperation can reduce pressure on constrained resources
  - Storage on the device
  - Wireless bandwidth

PURDUE UNIVERSITY

Roadmap

- Dependability basics
  - System design principles
  - Terminology and basic approaches
- Challenges and current results from
  - Embedded and mobile networks
  - Computational genomics
- Dependability in a cellular network [SRDS-16, Middleware-17, Crowdsense-17, Eurosys-18 (under submission)]
- Dependability in approximate computing [PACT-15, CGO-17, ASPLOS-18 (under submission)]

PURDUE UNIVERSITY

Performance Errors versus Accuracy

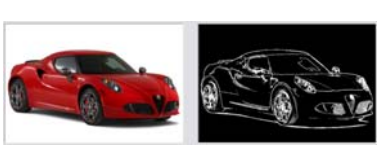
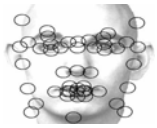


Image Processing

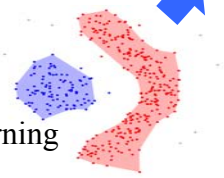
Media Applications

Can tolerate some imprecision

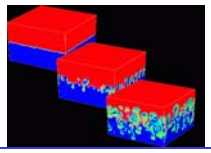
Data Analytics



Computer Vision



Machine Learning



Scientific Simulations

Simple Examples of Approximation Techniques

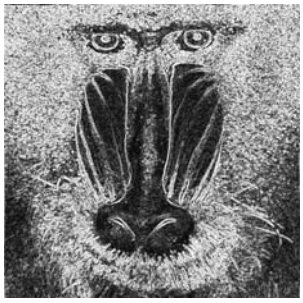
```
for(i=0;i<n;i=i+approx_level)
        result=compute_result();
```

Loop Perforation

```
for(i=0;i<n;i++)
if(0==i%approx_level)
        cached_result=result=compute_result();
else
        result=cached_result;
```

Memoization

Output quality degradation in Sobel



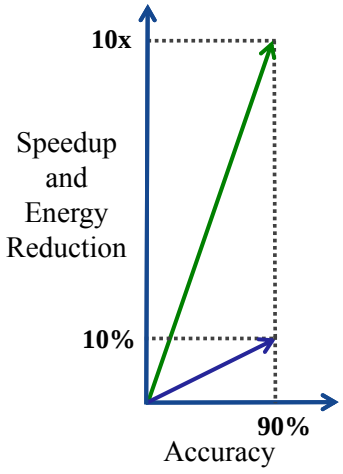
0% Quality Loss

5% Quality Loss

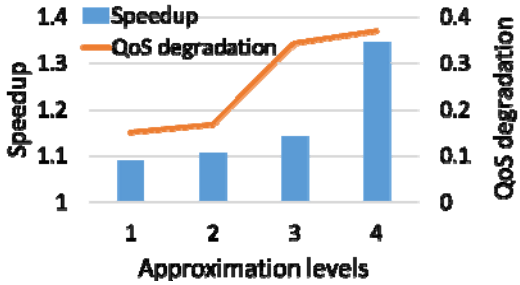
10% Quality Loss

10% Quality loss is nearly indiscernible to the eye and yet provides 57% energy savings

Approximate computing: Trade accuracy for energy saving or computation speedup



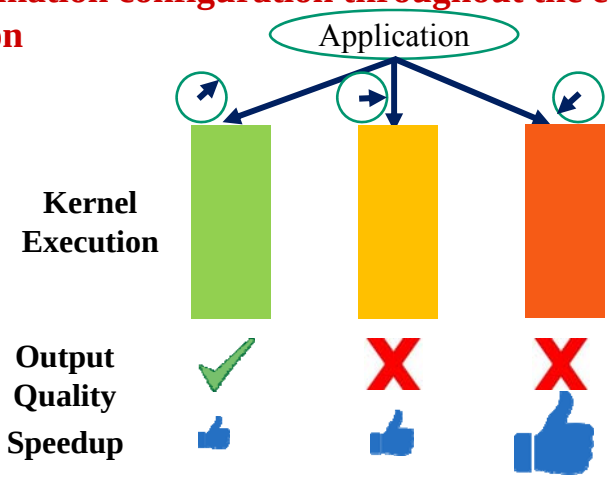
Adjust knobs to control the approximation-levels of computation



LULESH – A Hydrodynamic Simulation

Assumption of Monolithic Execution

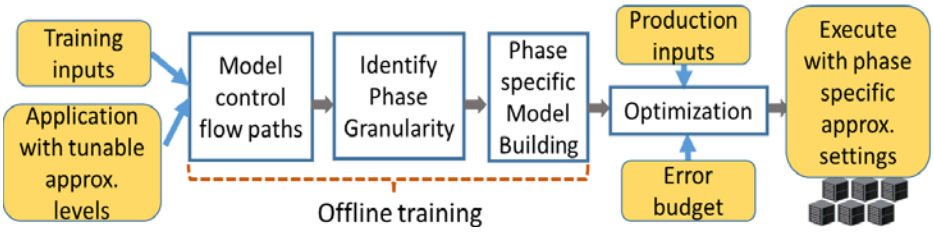
- **The general approach has been to have single approximation configuration throughout the entire execution**



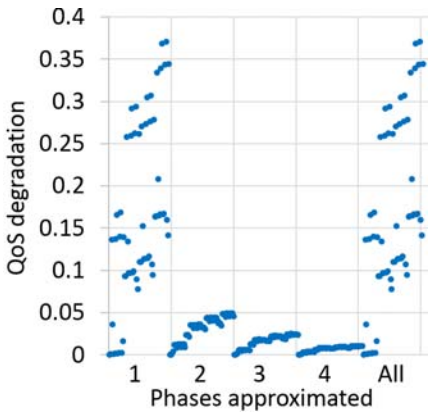
Phases in super-loop computations

```
1   elements = A set of elements to simulate
2   While(state->stable ==false)
3   {
4         increment_simulation_time();
5         forces_on_elements();
6         acceleration_of_elements();
7         velocity_of_elements();
8         position_of_elements();
9         strain _of_elements();
10        calculate_timeconstraints();
11        state = get_current_state();
    }
```

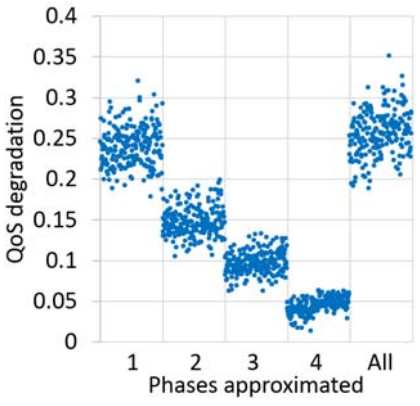
Workflow of OPPROX



Phase-specific QoS Characteristics

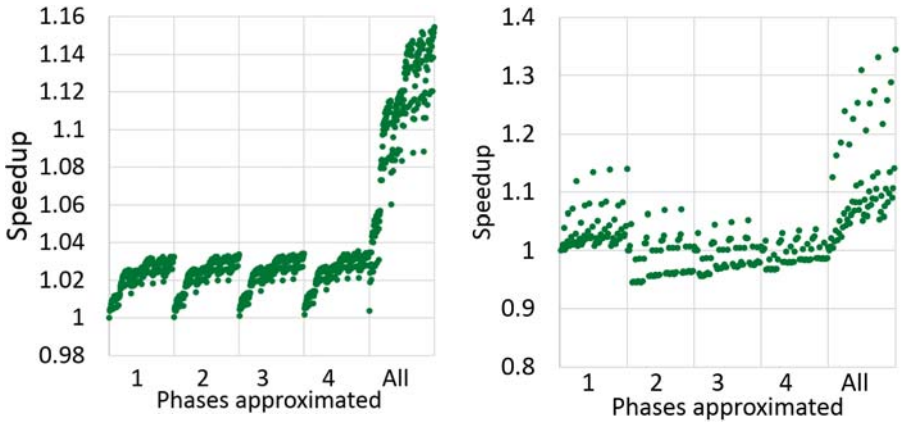


LULESH



Bodytrack

Phase-specific Speedup Characteristics



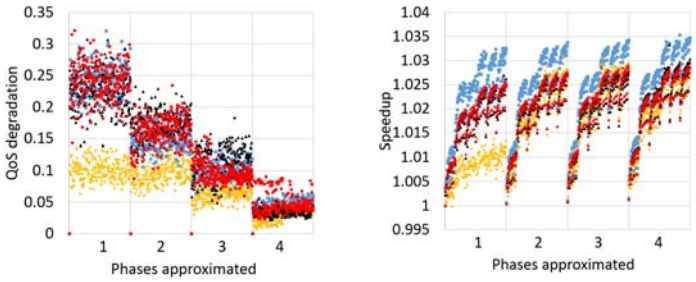
Bodytrack LULESH

PURDUE UNIVERSITY

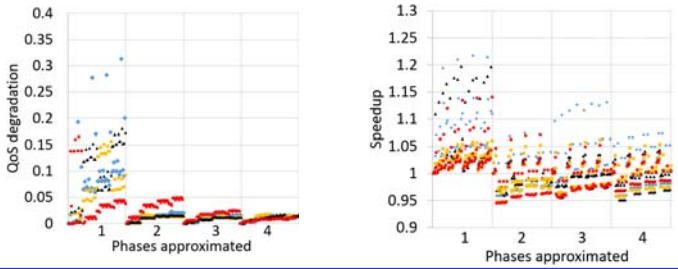
Input-Dependent Behavior

QoS degradation Speedup

Bodytrack

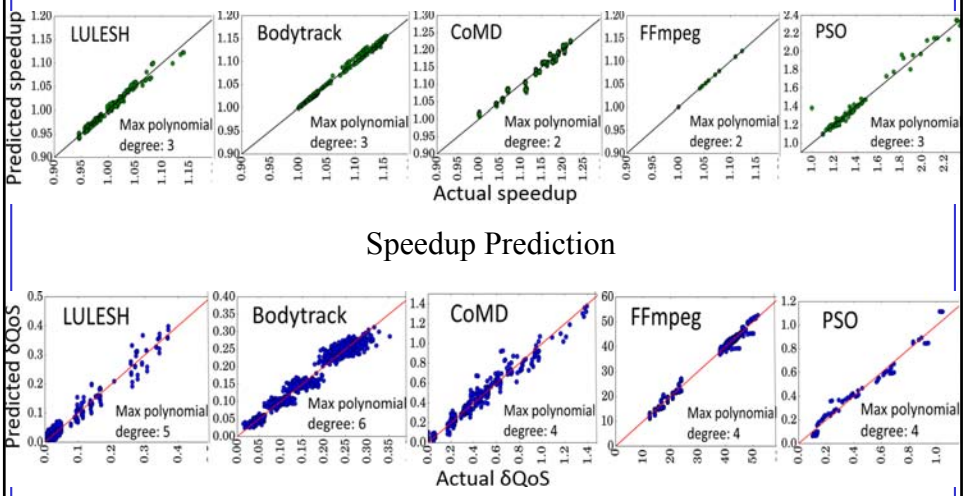


LULESH



PURDUE UNIVERSITY

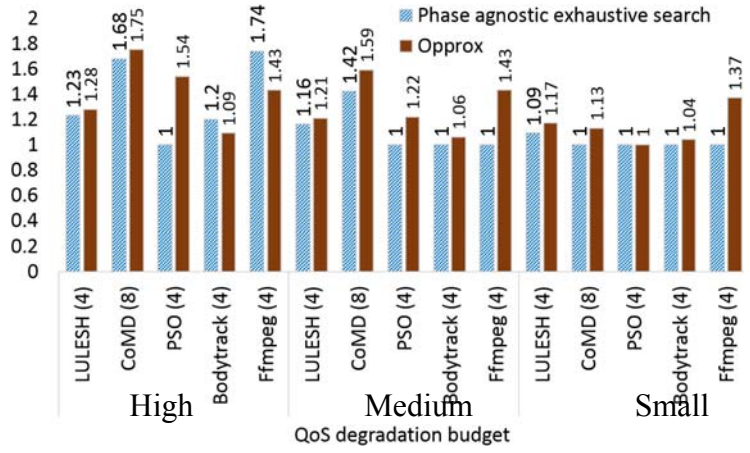
Evaluation of the Models



Speedup Prediction

QoS degradation Prediction

Speedup obtained by OPPROX



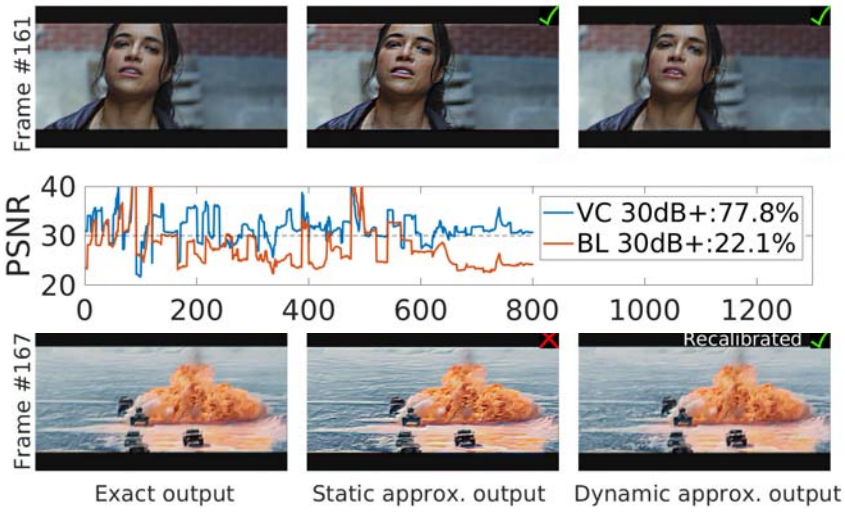
Phase-specific approximation is more attractive when operating under small error budget

Approximation in Video Processing

- **Fundamental challenge #1:** Best settings of approximate computations that can provide acceptable results *for the given input*
  - Optimal approximation setting is dependent on the content of the video
  - Empirically: Even different frames within the same video should have different values of approximation settings
- PSNR plot
- Actual video 1
- Approximate video 1 (acceptable)
- Actual video 2
- Approximate video 2 (unacceptable)

PURDUE UNIVERSITY

Perils of Input-Agnostic Approximation

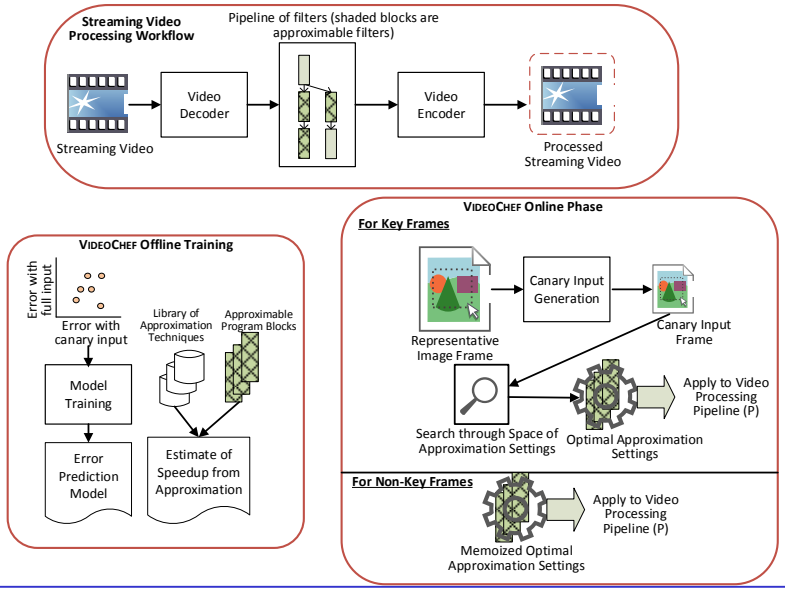


PURDUE UNIVERSITY

Approximation in Video Processing

- **Fundamental challenge #2:** Search space of approximation settings is large
  - Say: k stages in pipeline (small, say 2-5), n approximation settings for each stage (large, $n \geq 10$), then search space = $\Theta(n^k)$
  - Some model needs to be developed for efficiently searching through the space

Our Solution Approach: VideoChef



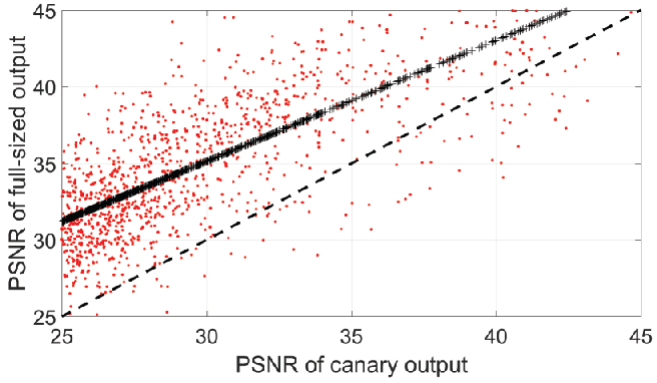
Our Solution: VideoChef

- Uses optimization techniques to find the best suitable settings for various approximation techniques inside video processing kernels
- Uses small-sized canary input to guide choice of approximation settings during optimization search
- Builds a prediction model for mapping the error from the canary input to that with the original input
- Online, VideoChef performs efficient search through approximation search space and bounds the search so that benefit of approximating the computation is realized
- VideoChef is the first technique that can apply tunable approximation algorithms to a streaming application in a manner that the benefit of approximation (minimization of computation cost) is actually realized

PURDUE UNIVERSITY

Modeling Error

- Error mapping model: Characterize the relation between error in the canary output and error in the full output, for the same approximation levels
- Mod
- Mod evel in effec
- Mod el in effec



PURDUE UNIVERSITY

Triggering Search

- MPEG-4 and many other video formats defines three main types of frames: I, P, and B frames.
- An I-frame uses intra-prediction meaning, the P- and the B-frames use inter-prediction
- When to insert an I-frame: a big difference in the frame triggers the insertion of a new I-frame, since inter-coding will give almost as long a code as intra-coding.
- In VideoChef, we leverage this observation and do a single search for a group of pictures, where a group is demarcated by I-frames at its two ends.

PURDUE UNIVERSITY

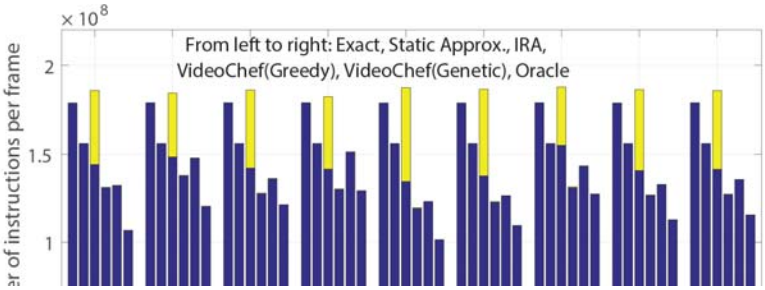
Evaluation

- 106 YouTube MPEG-4 videos
  - Collected from 8 different categories to cover a spectrum of different motion and color artifacts in the frames
- The user-provided acceptable video quality threshold is 30 dB, which is considered a typical lower bound for lossy image and video compression
- The different comparable protocols are:

1. Exact computation,
2. static approximation
3. IRA [PLDI 2016]
4. VideoChef Greedy
5. VideoChef GA
6. Oracle

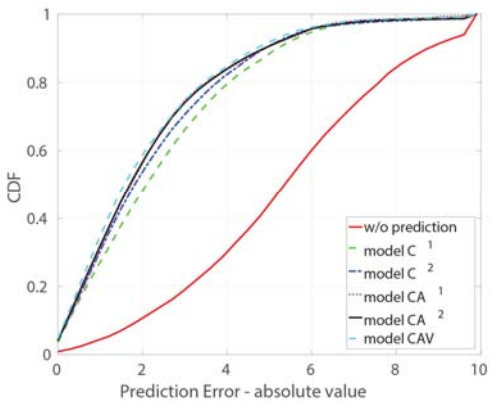
PURDUE UNIVERSITY

End-to-End Workflow Results



- VideoChef increases speedup by 28.7% over exact computation and is within 10.3% of the Oracle
- It outperforms both static approximation and IRA, by 22.4% and 33.7%
- Advantage exists for all the content categories; greatest savings in category 5 (movie trailers)
- Search overhead of VideoChef is small but it finds more aggressive approximations
- Counter-intuitively, for IRA, the number of executed instructions is **higher** than for exact computation, by 4.9%

Data Analytics in Systems: A Cautionary Tale



- We need data analytics (no prediction is really bad)
- But model CA does nearly as well as CAV
- CAV model parameters much harder to obtain at runtime

Concluding Insights

- Human perception is tolerant to errors
  - Allows for approximation and thus savings in energy and runtime
  - Approximation has to be done in a content-specific way
- But approximate computation needs to be agile
  - Needs to keep quality of output above user-specified threshold
  - Needs efficient mechanism to extract features from content
  - Needs efficient mechanism to search through large space of approximation settings
- VideoChef: First small step for approximation in streaming applications, in a content-dependent manner

PURDUE UNIVERSITY

Concluding Insights: Dependability and Data Analytics

- Dependability involves handling natural and malicious failures
- Handling involves: prevent, detect, diagnose, contain, repair
- Rule-bases, exact solutions giving way to data analytic solutions
  - Too much data
  - Too much noise
  - Too many modes of interaction
  - Incomplete observability into some software components

PURDUE UNIVERSITY

Concluding Insights: Dependability and Data Analytics

- Data analytic solutions themselves have to be scalable
- Solutions have to make adjustable trade-offs between false positive and false negative errors
- Different domains provide different constraints and opportunities for the dependability solutions
  - Embedded and mobile networks
  - Computational genomics

PURDUE UNIVERSITY

Take-Aways: More Philosophical

- Explore at the point of greatest curiosity
  - But, beware of rabbit holes
  - Learn from what was tried but did not work
- Do not be intimated by volume of prior work
  - Question assumptions underlying the work: Have they changed due to world moving on/technological changes
  - Experiment, do not be an armchair researcher – in the small first, and see if the prior story still holds
  - Learn from your own mistakes
- Socialize your research
  - Discuss with your peers, not just in your immediate group
  - Cooperate - open source code, brainstorm ideas, give feedback on other's drafts – successful students often have a supportive ecosystem around them

PURDUE UNIVERSITY

It Takes a Village

- **Purdue**: [Faculty] Milind Kulkarni, Mathias Payer, He Wang, [Students] Abe Clements, Kanak Mahadik, Matt Creti, Nawanol Theera, Chris Wright
- **Argonne National Lab**: Folker Meyer
- **AT&T Labs:** Kaustubh Joshi, Rajesh Panta
- **Google**: Greg Bronevetsky
- **Georgia Tech**: Mostafa Ammar, Ellen Zegura
- **UIUC**: Sasa Misailovic

PURDUE UNIVERSITY