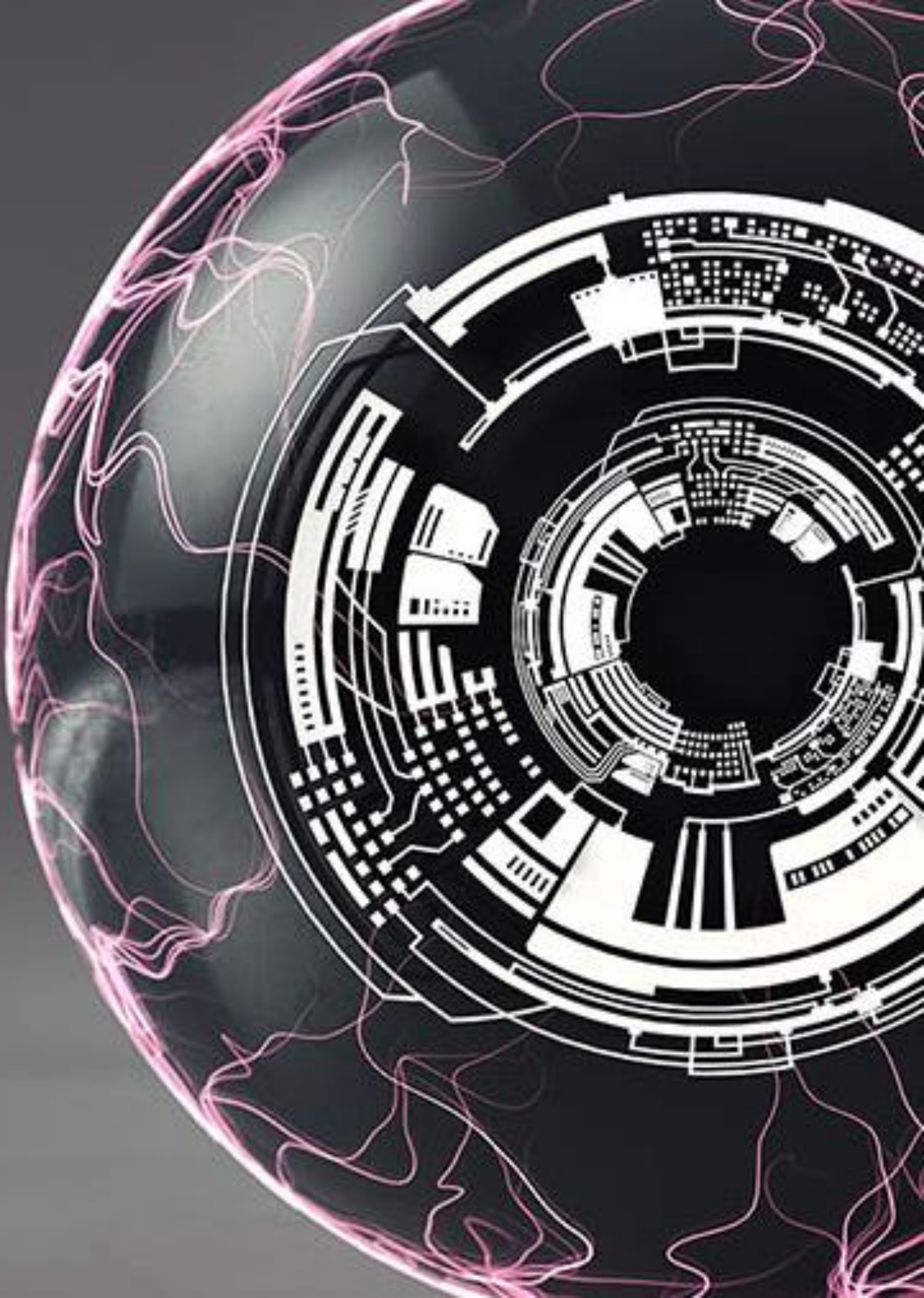
Microsoft

Odysseys in Testing and Analyzing Mobile Apps

Mayur Naik
Georgia Institute of Technology



Odyssey

: A long journey full of adventures

: A series of experiences that give knowledge or understanding to someone

Source: Merriam-Webster's Dictionary

The Mobile App Life Cycle

Reliability | Security | Performance

Development and Testing | Pre-deployment Certification | Post-deployment Adaptation

New software engineering problems in all stages → need new tools based on program analysis

Three Odysseys

Reliability	Security	Performance
		
Developmen t and Testing	Pre-deployment Certification	Post-deployment Adaptation
Dynodroid	Stamp	CirrusCloud

Dynodroid

Automated Testing of Mobile Apps

Key Idea: View app as an event-driven program

s0 −e1→ s1 −e2→ s2 −e3→ . . .

Broadly two kinds of events:

UI event: LongTap(245, 310), Drag(0, 0, 245, 310), ...

System event: BatteryLow, SMSReceived(“hello”), ...

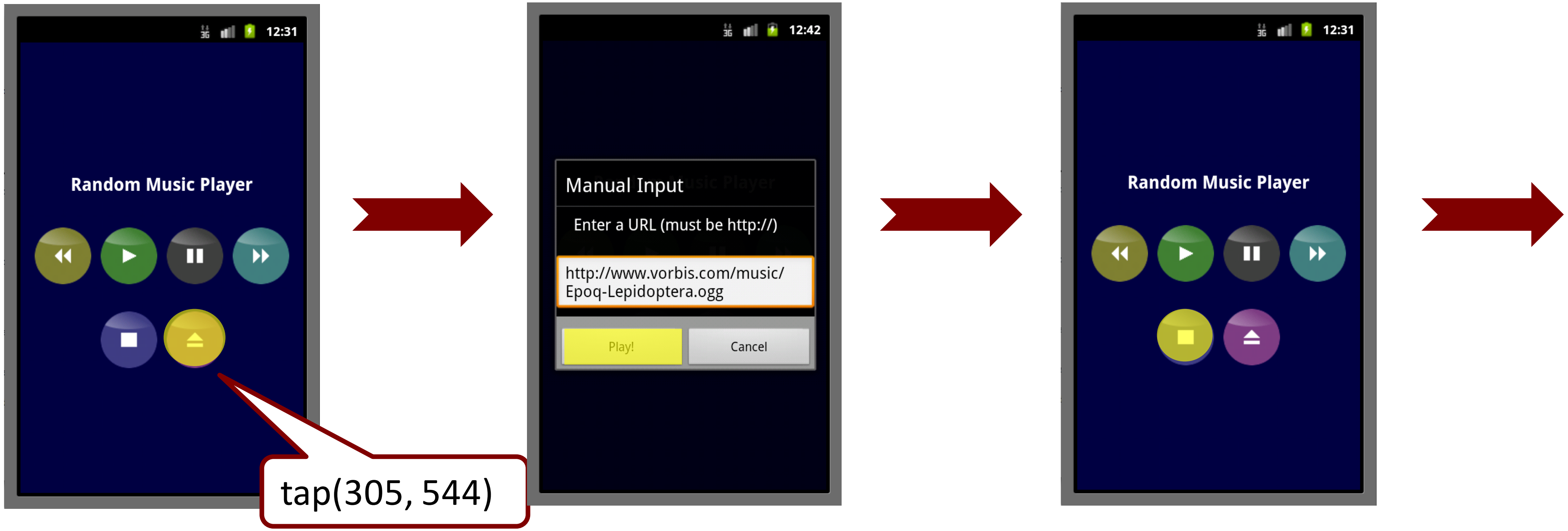
Assumption: Fixed concrete data in environment (sdcard, network, . . .)

May cause loss of coverage

Automated Testing of Mobile Apps

Key Idea: View app as an event-driven program

s0 – e1 → s1 – e2 → s2 – e3 → . . .



Automated Testing of Mobile Apps

Key challenge: Large number of possible events

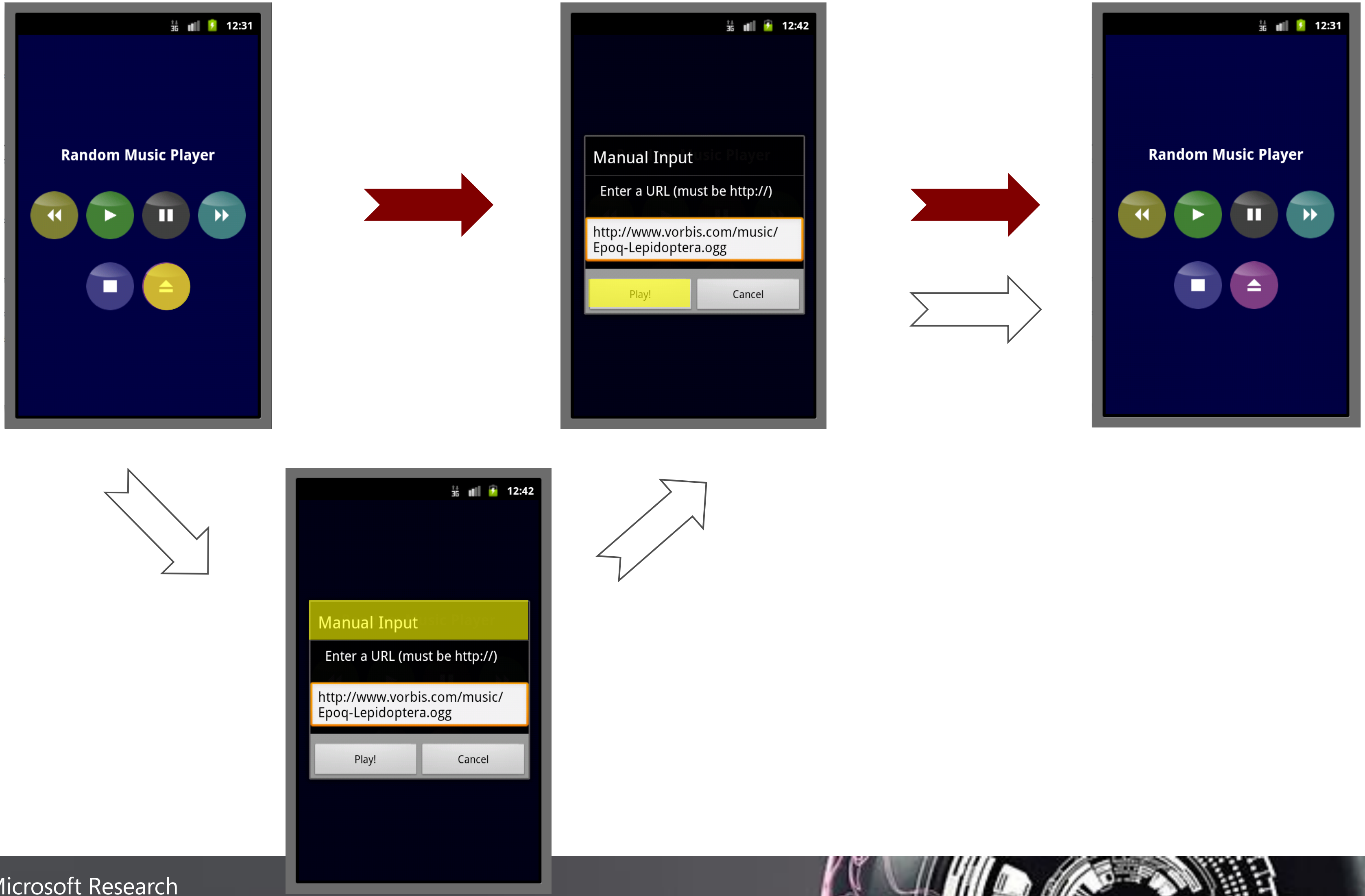
E.g., 108 system events in Android Gingerbread

Insight #1: Few events are *relevant* in any state

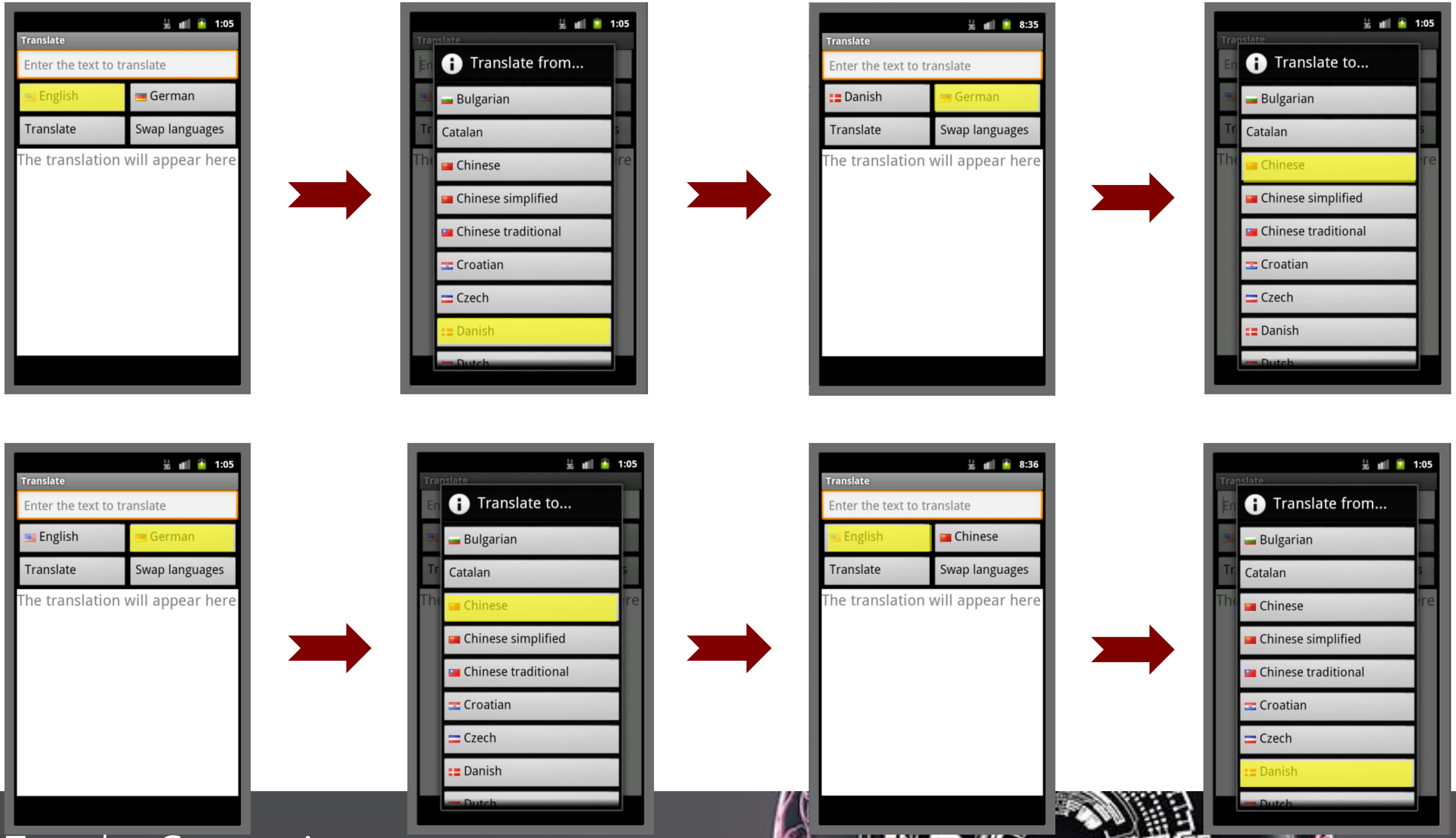
Insight #2: Many event sequences are *equivalent*

Our solution: Identify both conditions by specializing to app framework

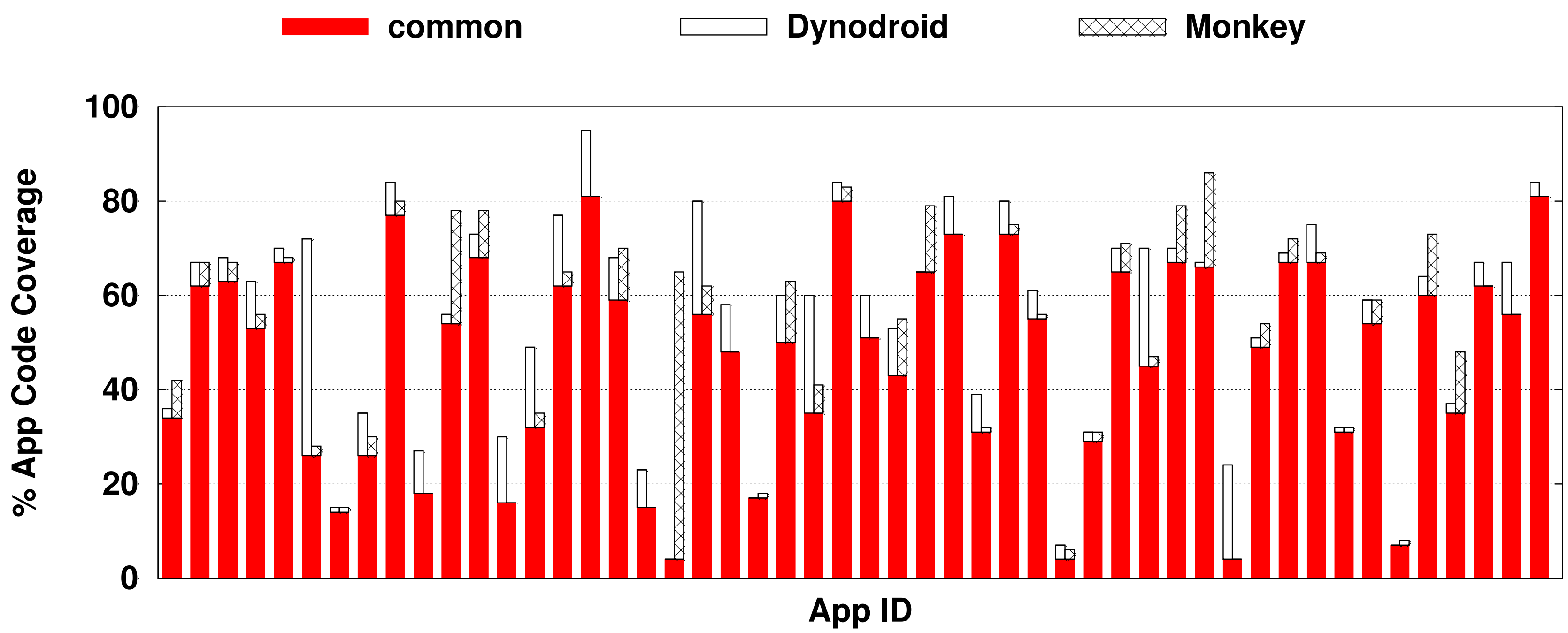
Example of Equivalent Event Sequences



Example of Equivalent Event Sequences

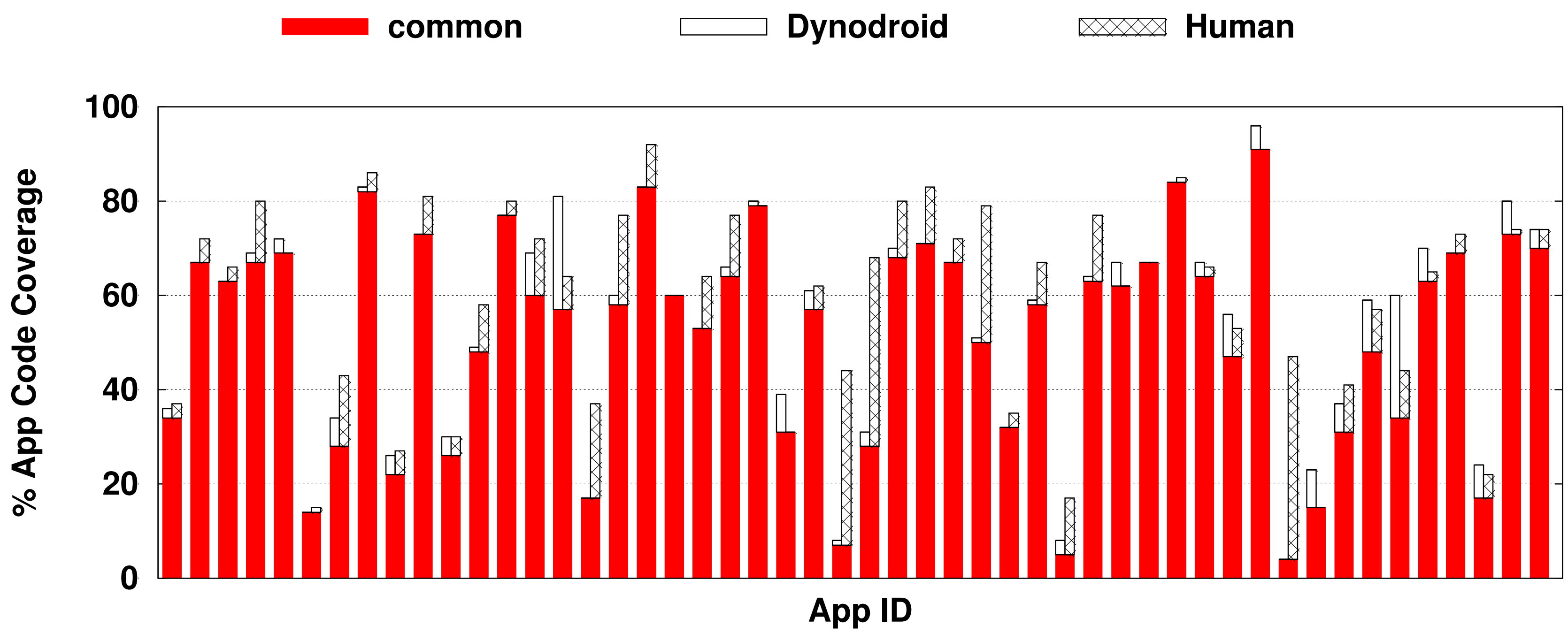


Code Coverage: Dynodroid vs. Monkey



Dynodroid achieves higher coverage than Monkey for 30 of the 50 apps.

Code Coverage: Dynodroid vs. Humans



Automation Degree = C(Dynodroid ∩ Human) /C(Human)

Range = **8-100%**, Average = **83%**, S.D. = **21%**

Sample Feedback from Participants

"Tried to cancel download to raise exception."

"Human cannot trigger change to AudioFocus."

"Many, many options and lots of clicking but no actions really involved human intelligence."

"There are too many combinations of state changes (play -> pause, etc.) for a human to track."

A Problem: Path Divergence

- Results from missed propagation of symbolic values in uninstrumented code

  Primarily native (C/C++) code, occasionally Java code (e.g., object serialization)

- Divergence can be your friend: served as a beacon for bugs in our implementation

  ~ 0% today in our implementation for Android (compared to ~ 40% for SAGE)

Automated Testing – Looking Ahead

Some old problems …

Scalability (path explosion)

- Demand-driven?
- Which inputs to treat symbolically and when?

D. Ramos and D. Engler. Under-Constrained Symbolic Execution: Correctness Checking for Real Code. USENIX Security 2015.

I. Yun, C. Min, X. Si, Y. Jang, T. Kim, M. Naik. APISan: Sanitizing API Usages through Semantic Cross-checking. USENIX Security 2016.

… and some new ones

Framework model synthesis

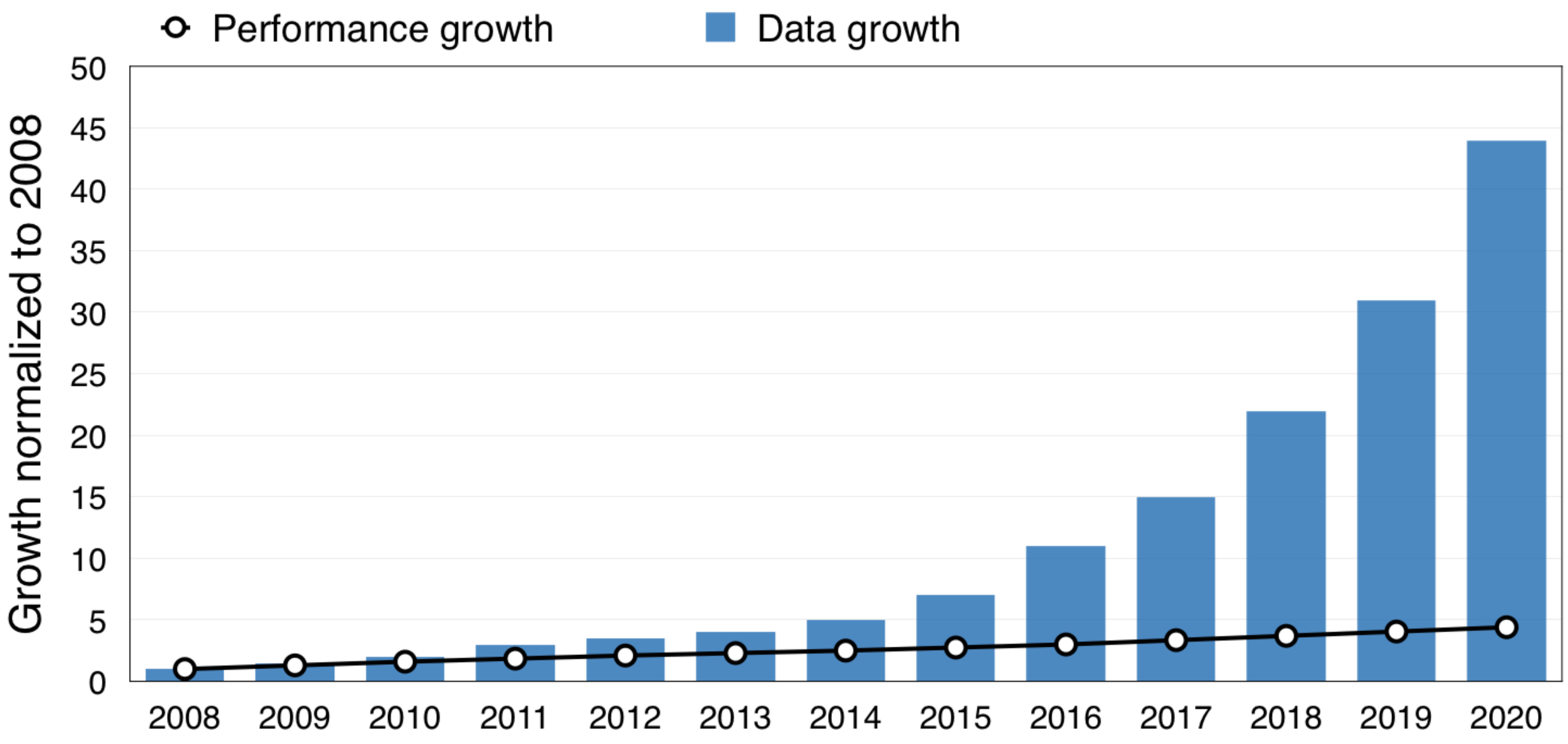
- Debugging support to localize false alarms?
- Engage user-in-the-loop?

CirrusCloud

Smartphone Trends

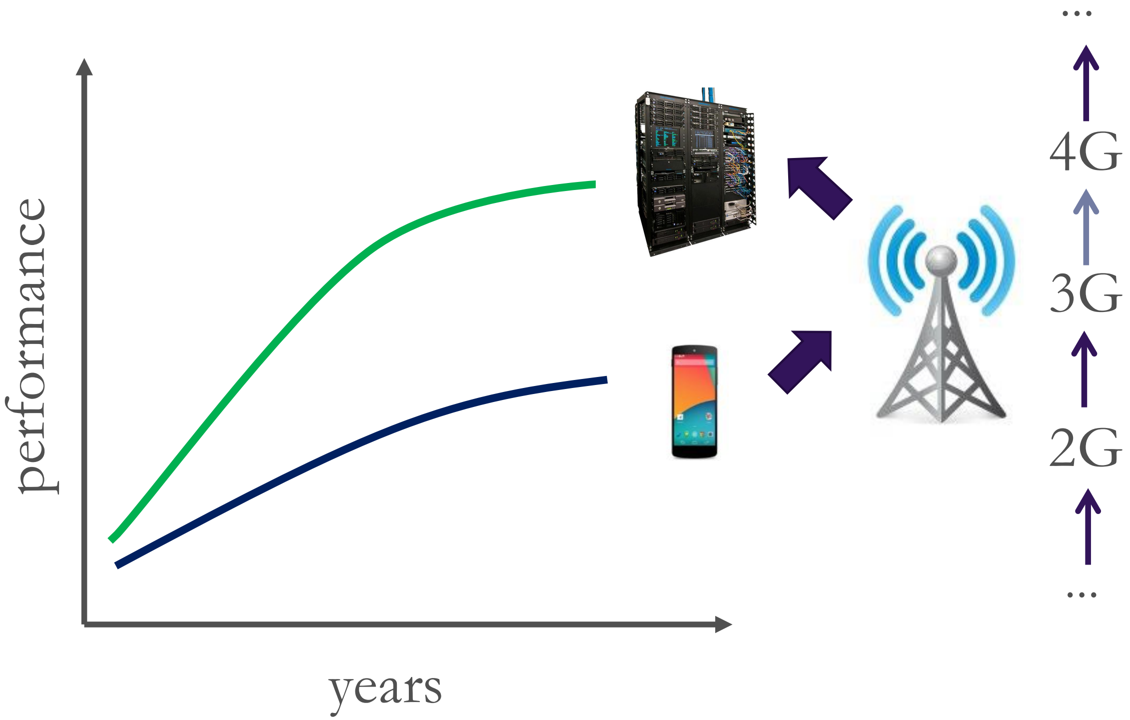
	CPU (GHz)	Screen Res. (thousand pixels)	Rear Camera (MP)	Front Camera (MP)	Sensors	Battery (mAh)
iPhone	0.4	153	2	-	3 (light, accelerometer, proximity)	1,400
iPhone 3	0.6	153	3	-	4 (light , accelerometer, proximity, compass)	1,150
iPhone 4	0.8	614	5	0.3	6 (light, accelerometer, proximity, compass, gyroscope, infrared)	1,420
iPhone 5	1.3 dual core	727	8	1.2	7 (light, accelerometer, proximity, compass, gyroscope, infrared, fingerprint)	1,560
iPhone 6	2.0 dual core	1000	12	5.0	8 (light, accelerometer, proximity, compass, gyroscope, infrared, fingerprint, barometer)	1,715

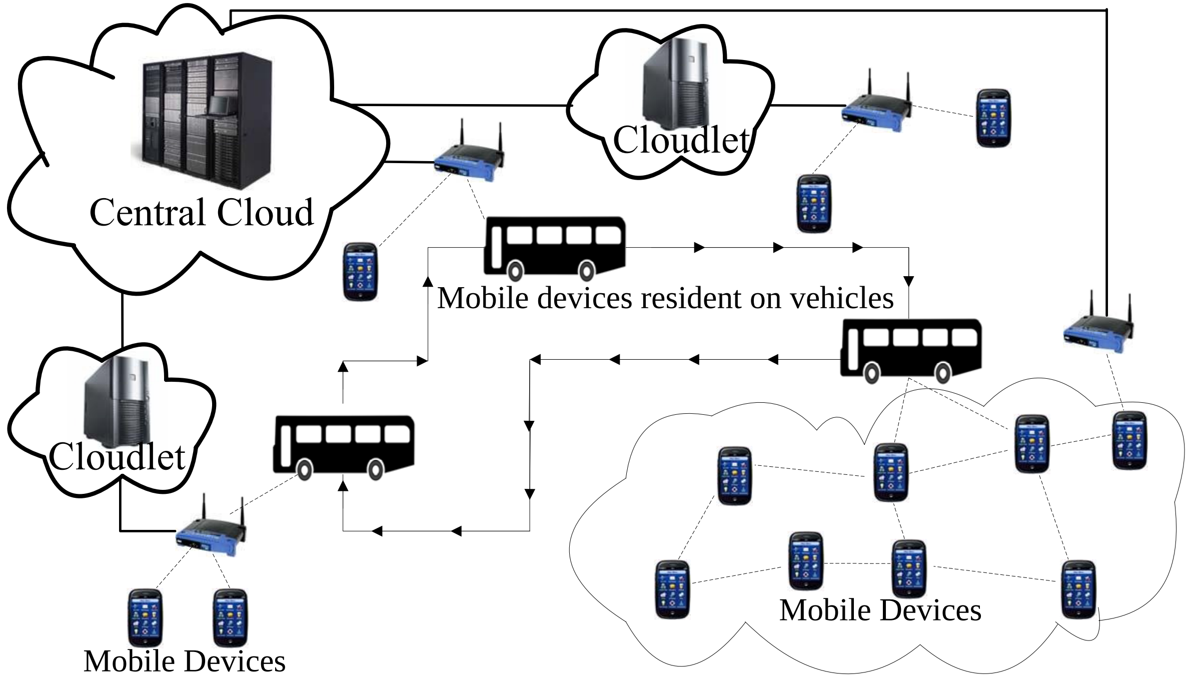
A Challenge: Data growth vs. Performance



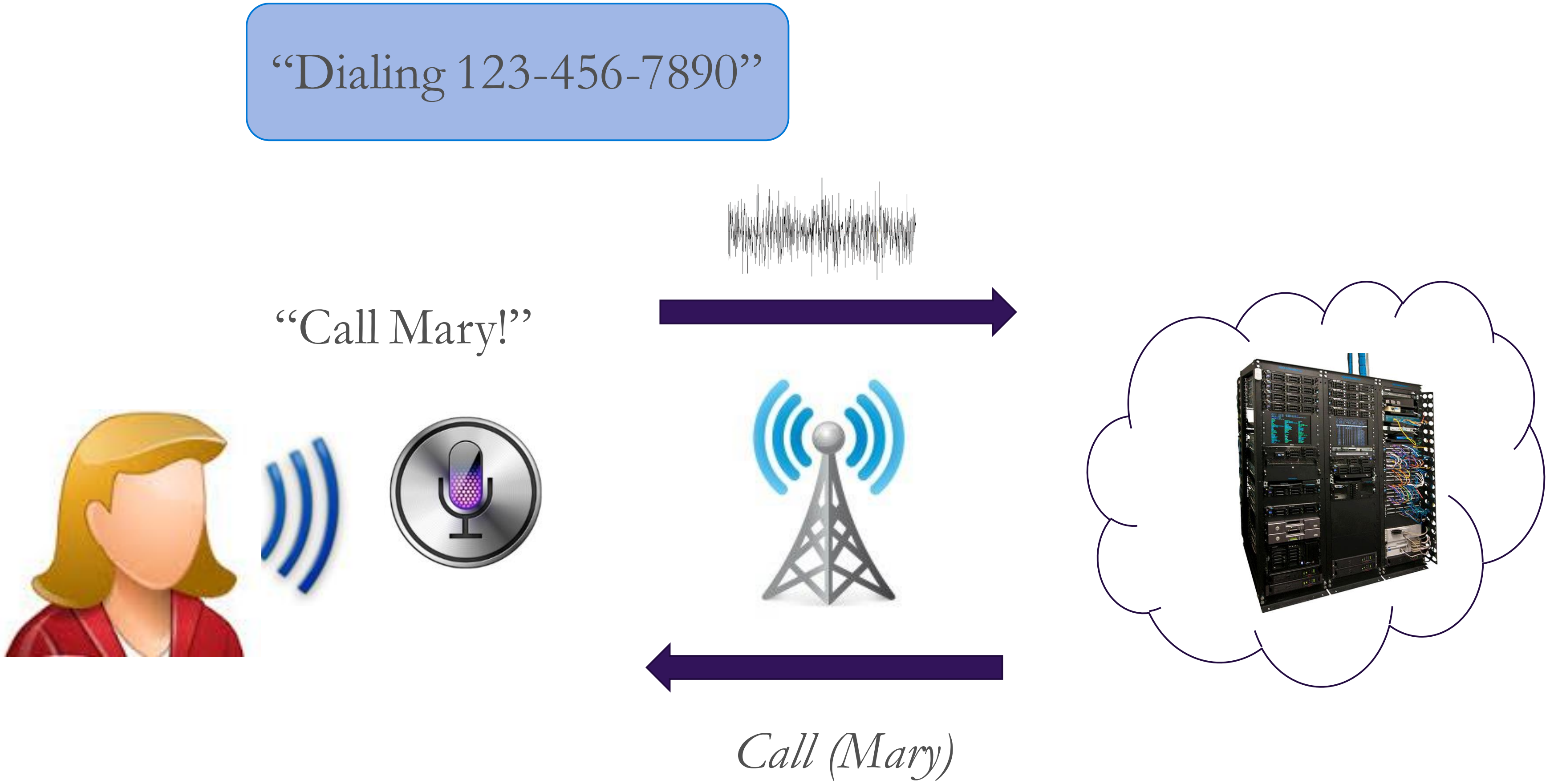
- Data growth trends: IDC's Digital Universe Study, December 2012
- Performance growth trends: Esmaeilzadeh, Blem, St. Amant, Sankaralingam, Burger. Dark silicon and the end of multicore scaling. ISCA 2011.

An Opportunity: Mobile-Cloud Computing

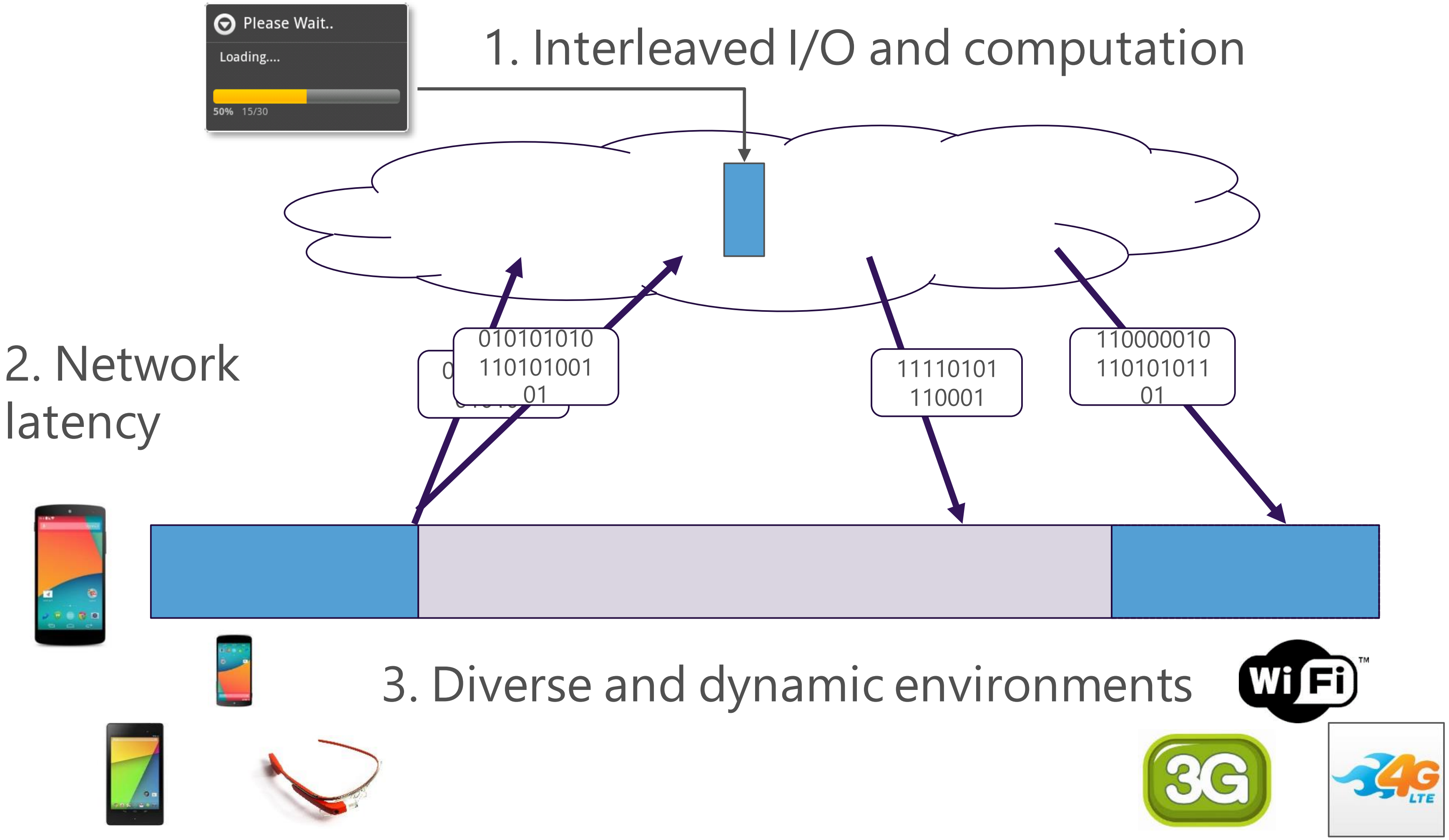




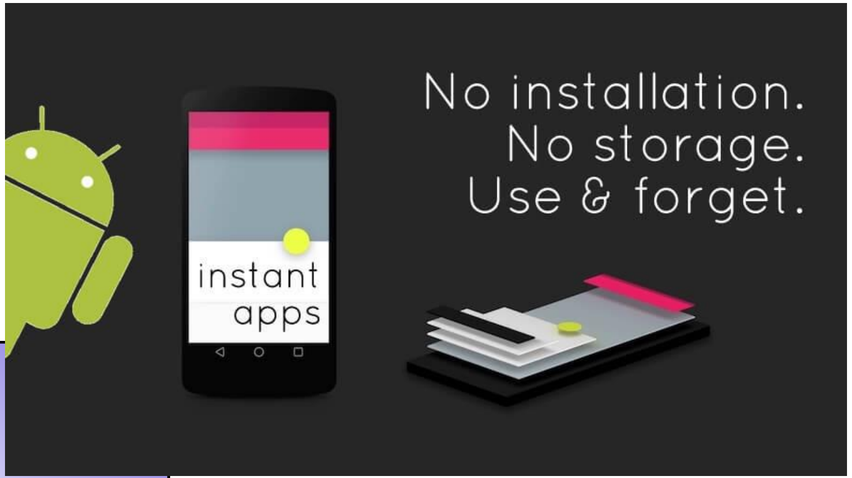
It's Already Here



Challenges to Broader Use



Our Contributions

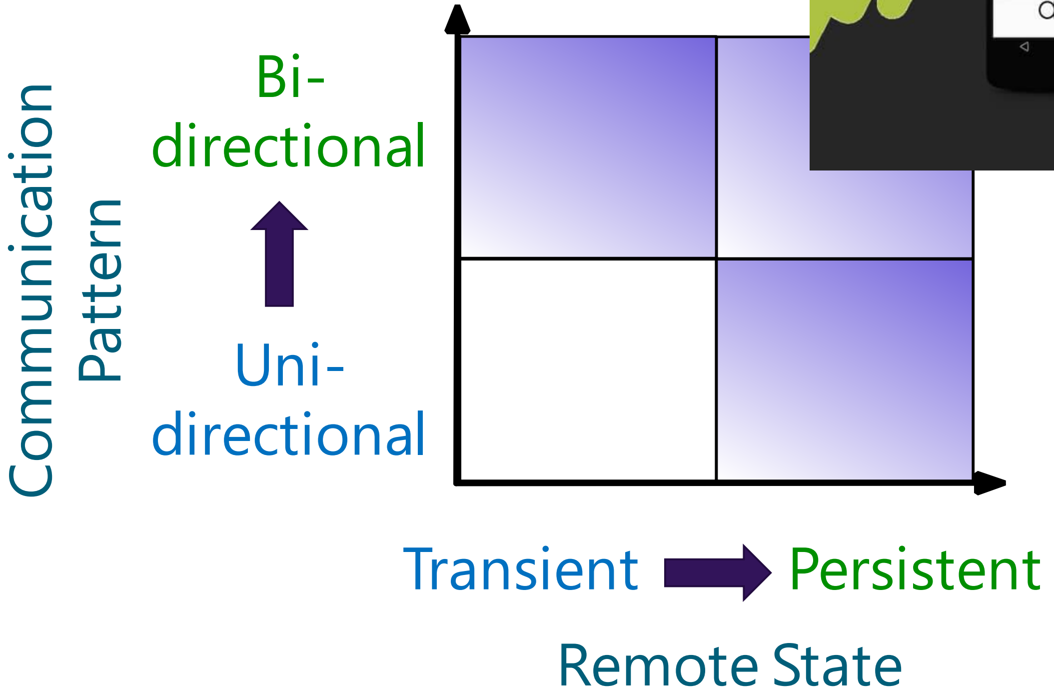


Interleaved I/O and computation

Diverse and dynamic environments

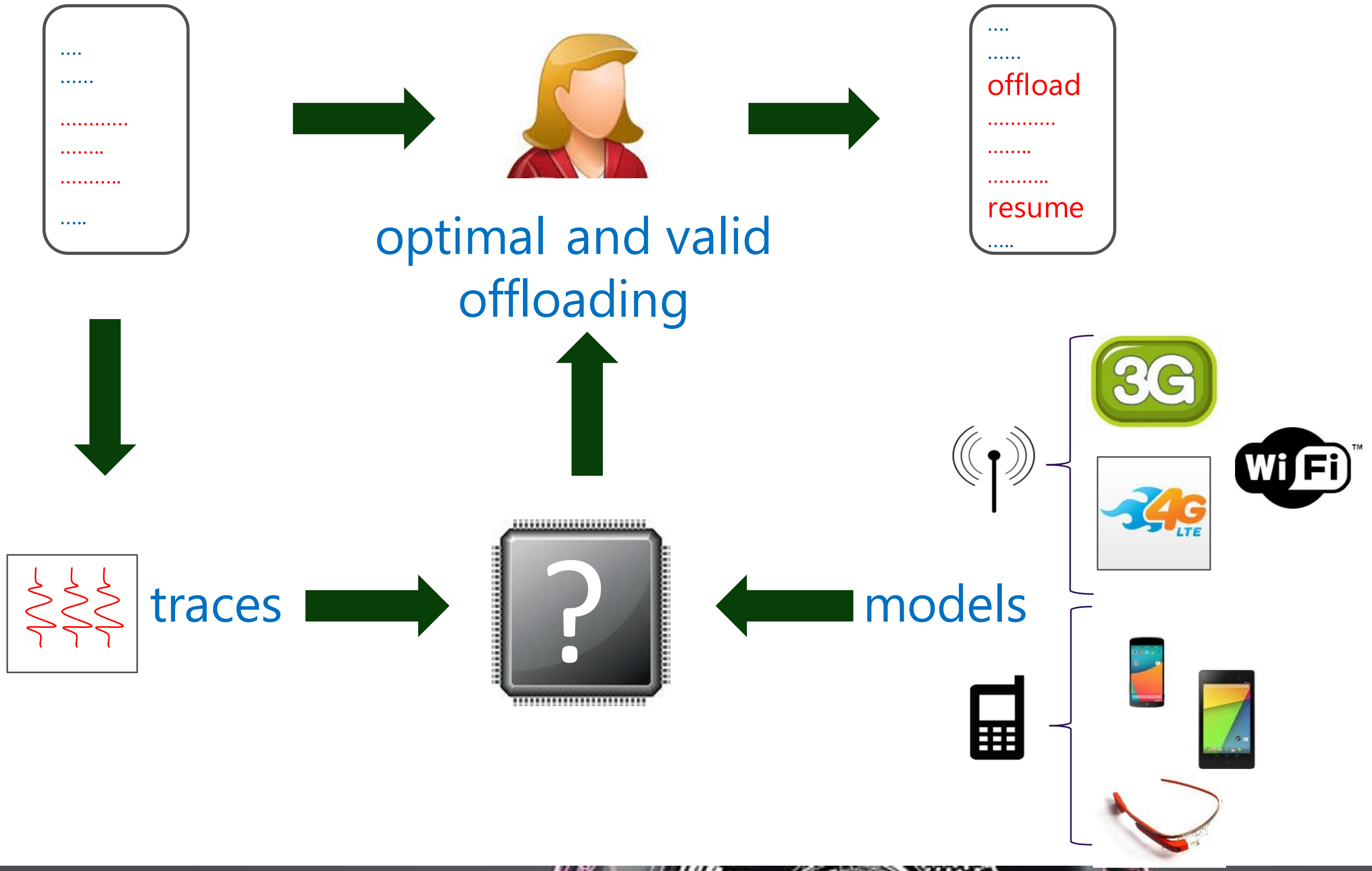
Optimization Problem

ILP ➡ Min-Cut

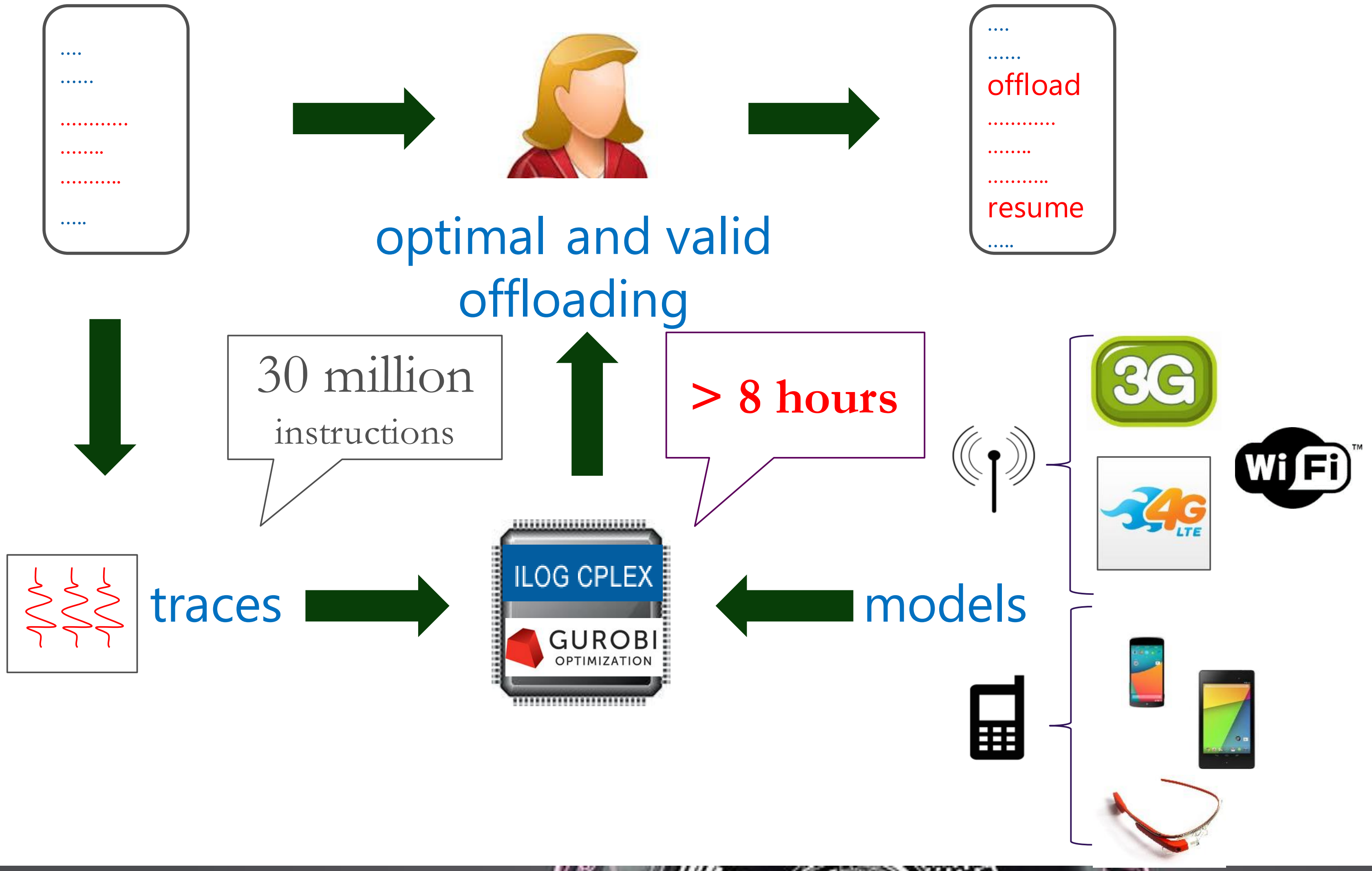


Network latency

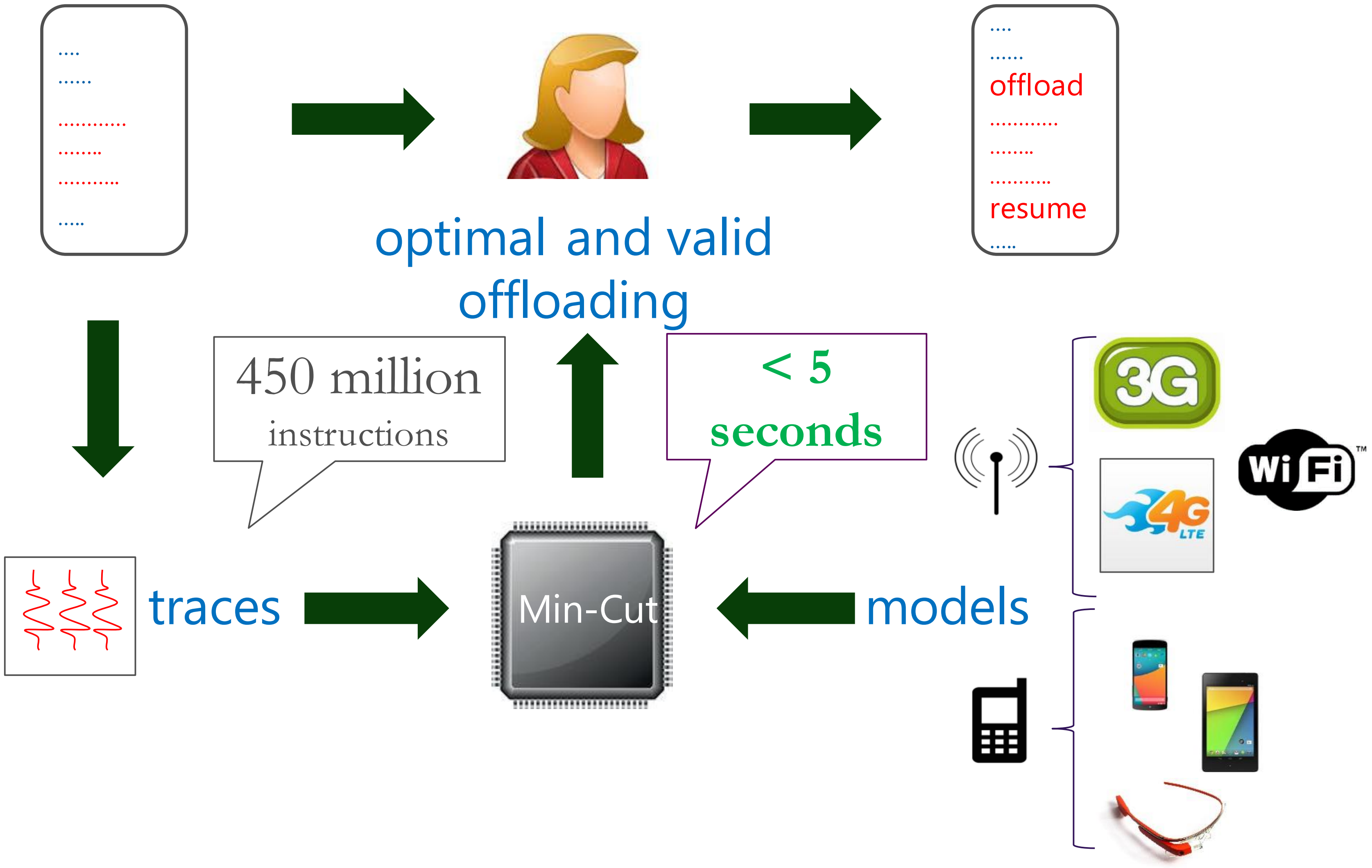
Overall Approach



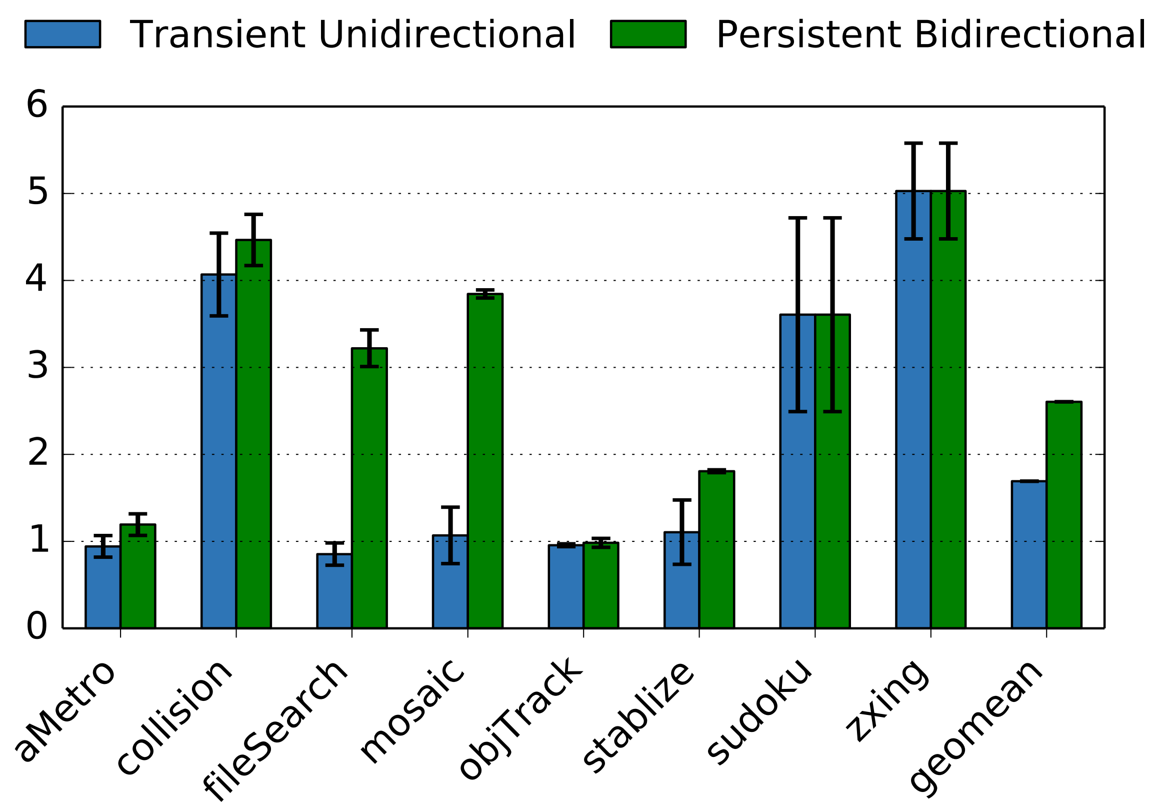
Previous Approaches: ILP

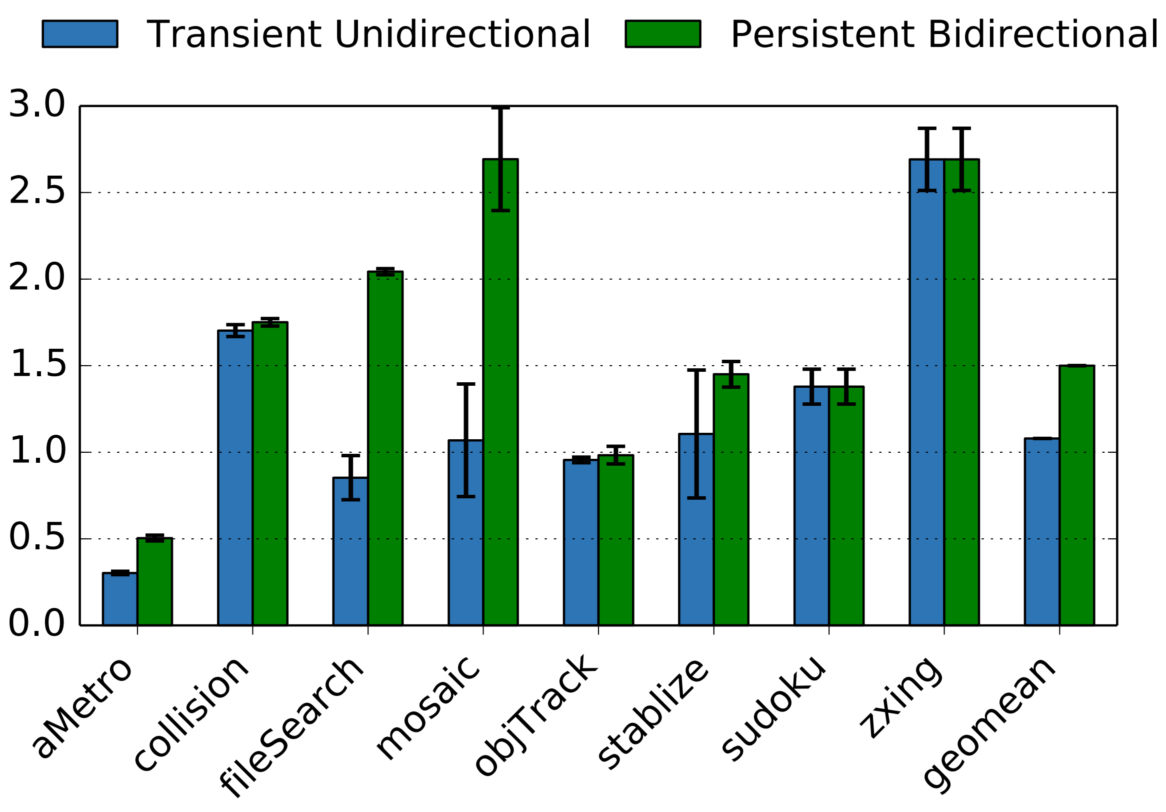


Our Approach: Min Cut

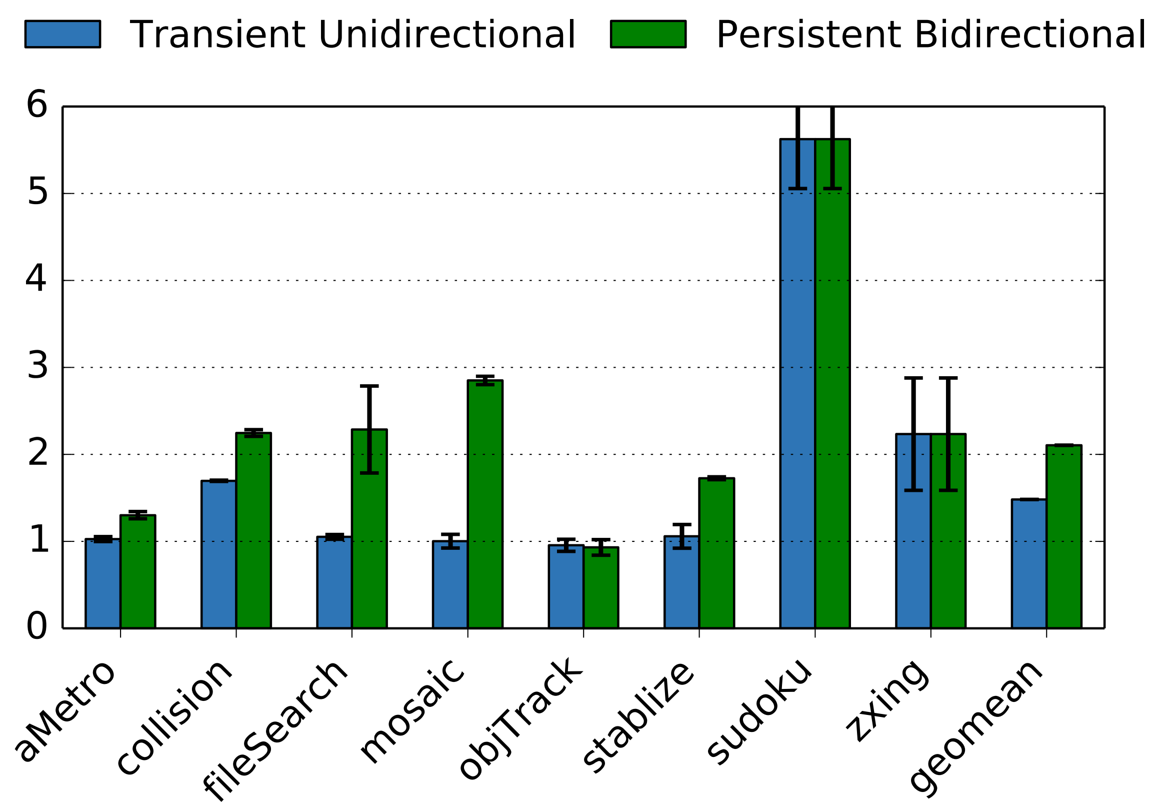


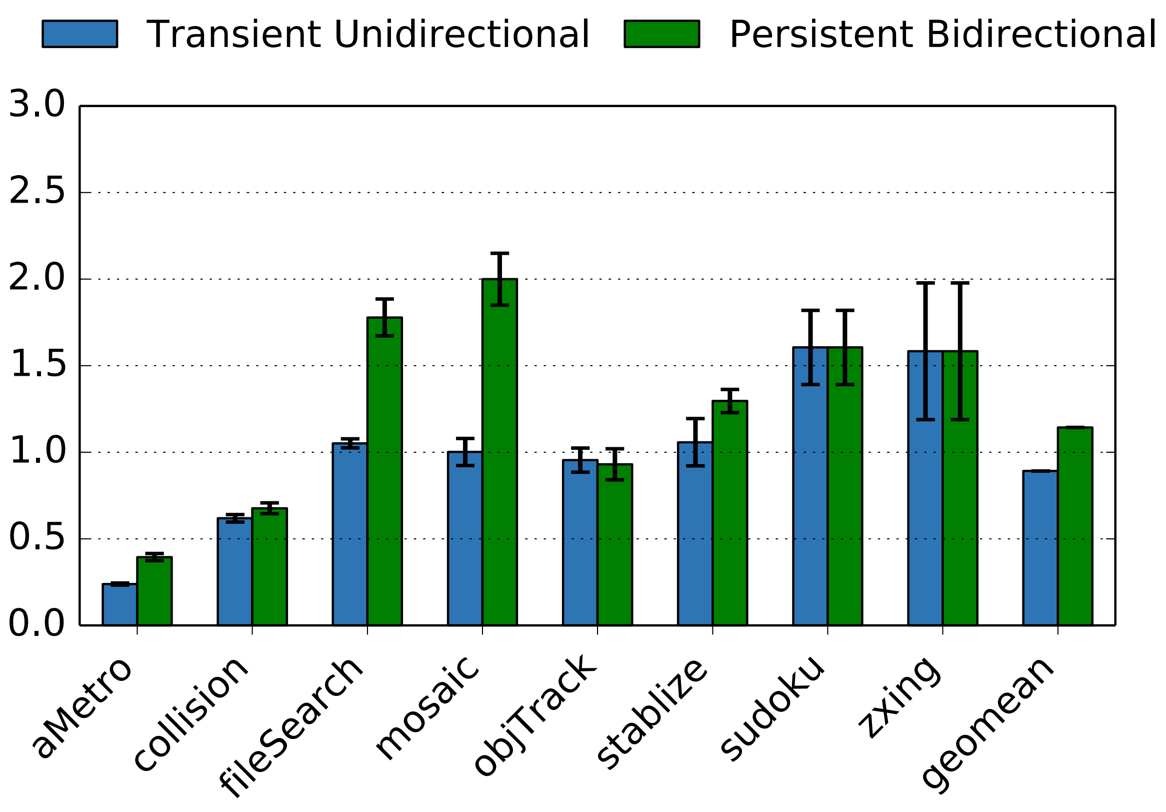
Speedup on Galaxy S3 – WiFi and LTE



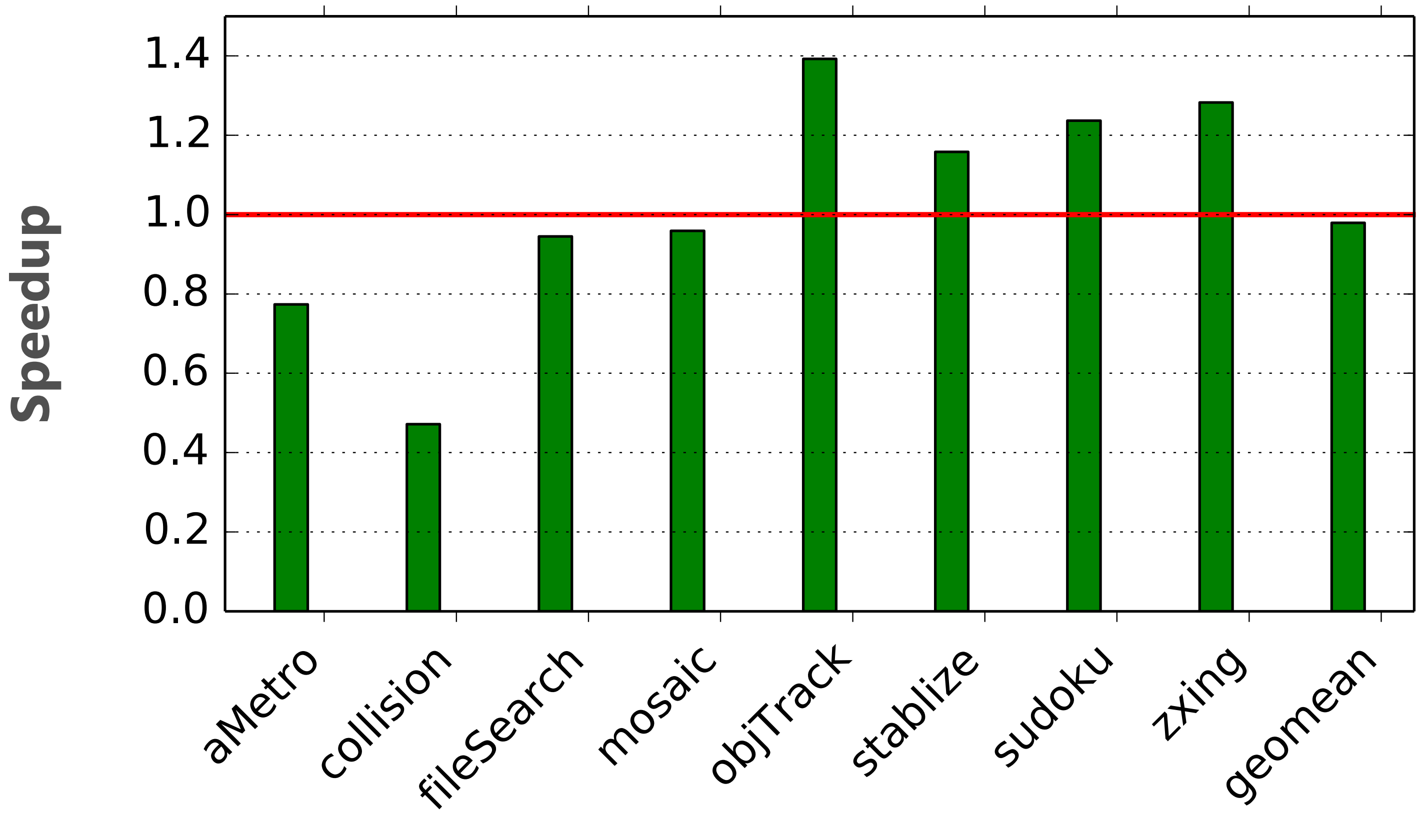


Speedup on Galaxy S2 – WiFi and LTE

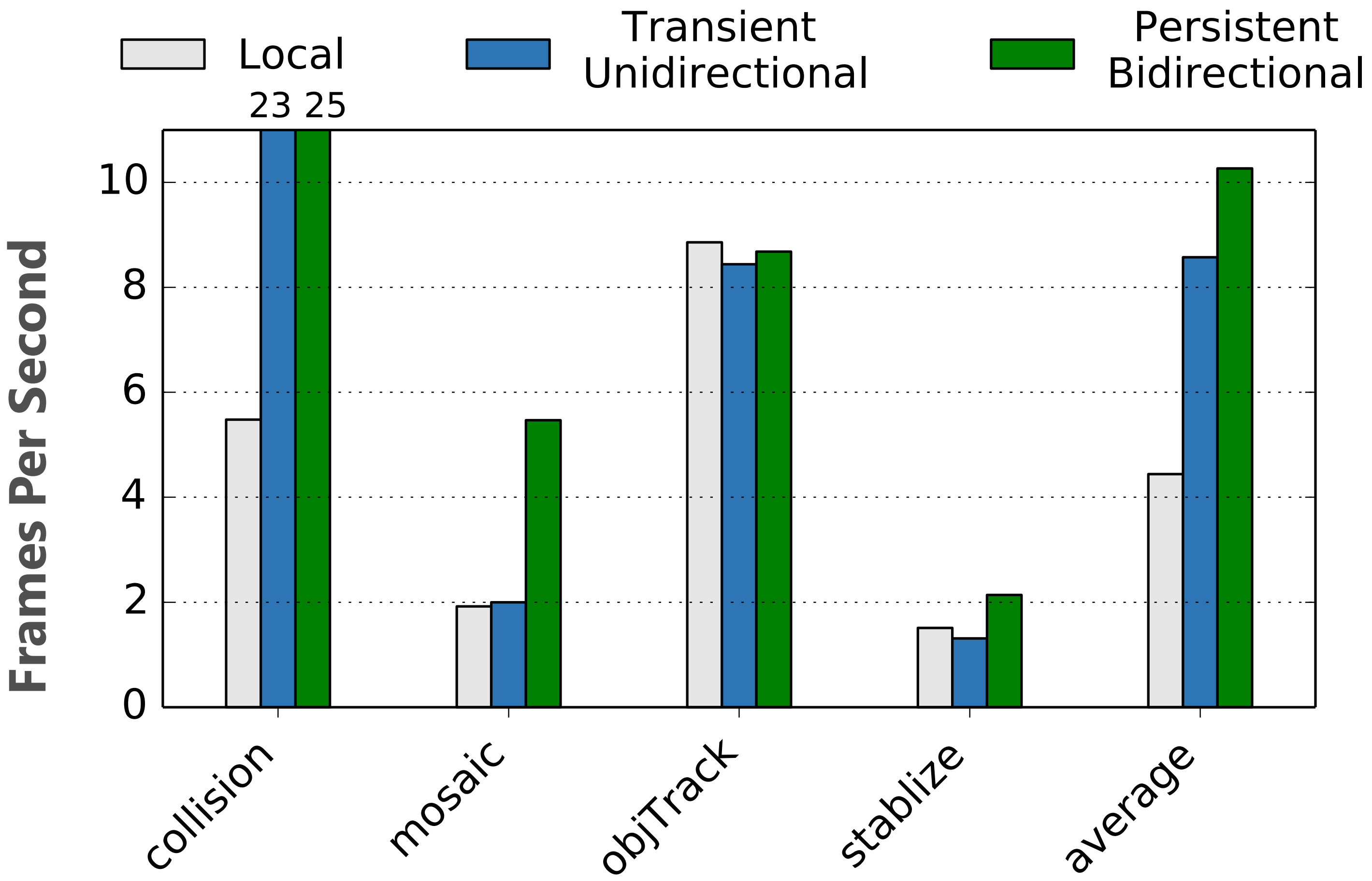




Speedup of S3 over S2



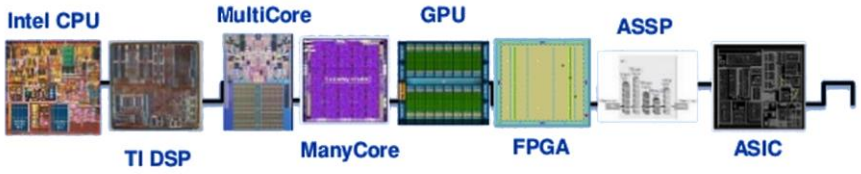
Quality for Video Apps



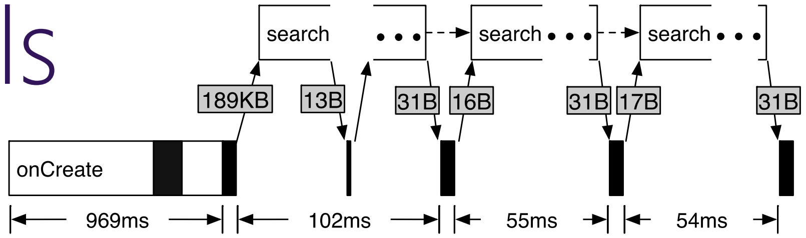
Demo: Video Stabilization

Partitioning and Offloading – Looking Ahead

Leveraging Accelerators in the Cloud



Specialized Communication Protocols



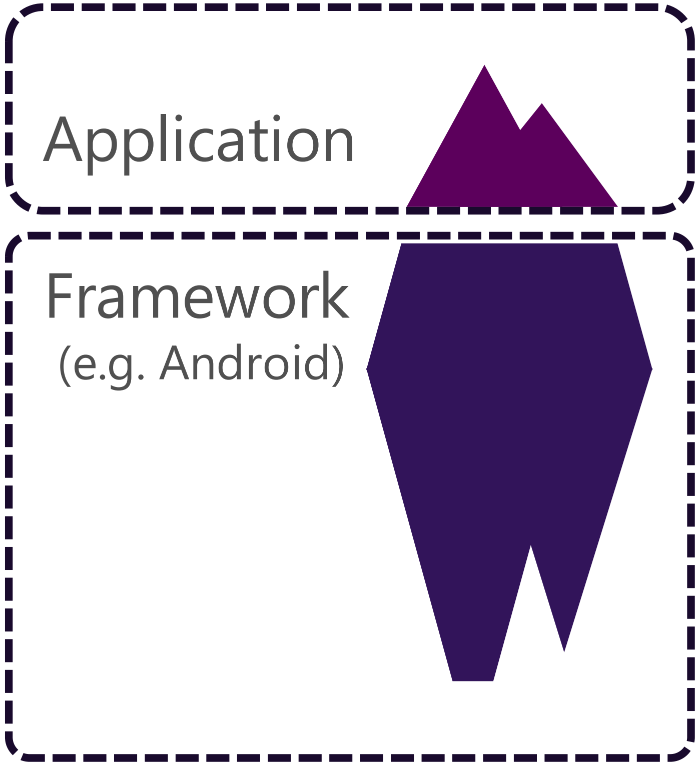
Programming Models

Approximate computing [safety, quality requirements, ...]

Distributed computing [consistency, fault tolerance, ...]

Sequential Concurrent Weakly consistent Partially consistent

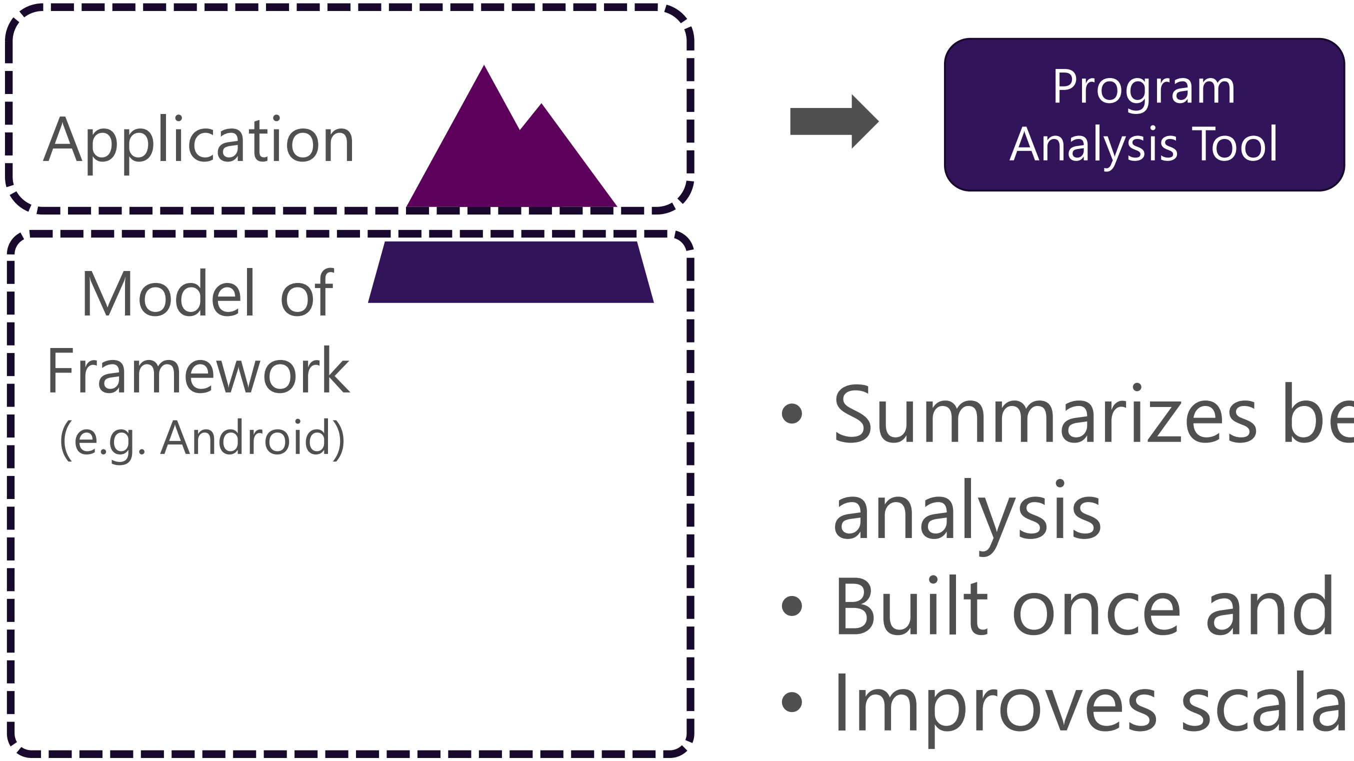
Challenge or Opportunity?



- Hard to analyze (e.g., native code, reflection)
- Very large
- Mostly irrelevant to the analysis

Challenge or Opportunity?

Challenge or Opportunity?



- Summarizes behaviors relevant to analysis
- Built once and for all
- Improves scalability of analysis