Mobile and Sensor Systems

Lecture 6: Mobile Sensing Energy and Systems Considerations

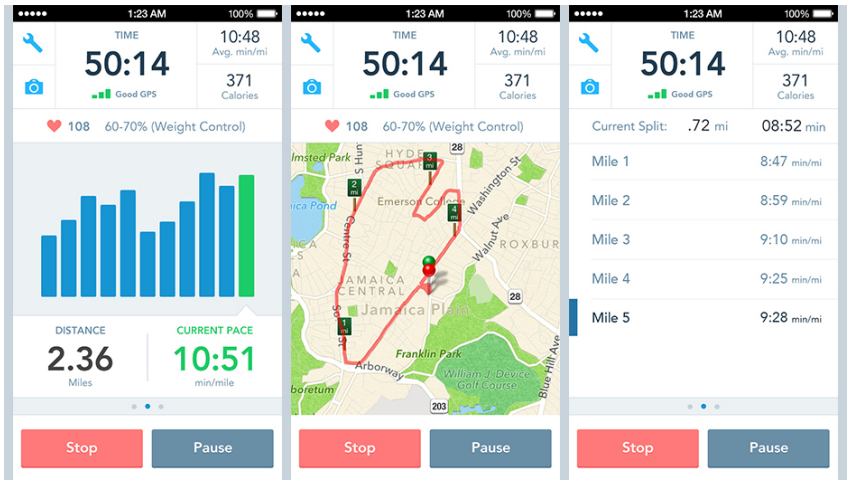
Prof Cecilia Mascolo

UNIVERSITY OF CAMBRIDGE

In this Lecture

- We will study approaches to preserve energy in mobile sensing systems
- We will look at aspects of local versus cloud computation

Context based Apps



UNIVERSITY OF CAMBRIDGE

Continuous Monitoring is "expensive"

- Apps need continuous sensing to
  - Understand and relate the context of the user
  - Trigger actions

- Monitor through sensors continuously is expensive battery wise

Solutions

- Avoid continuous sensing
  - Duty cycling the sensors: cons, are that the view of the user activity might not be complete

- Share context sensing among multiple apps
  - Shared cache of context sensed data
  - Cross app context correlations

Shared Context Sensing

- At 10am App1 asks the value of "Driving"
  - Determined through features of the accelerometer sensor
- System caches it
- At 10.05am App3 asks the value of "Driving"
  - Cache value is returned avoiding sensor sampling

Cross App Context Sharing

- Context inference from "other" features is possible. An app can learn one attribute by the attributes learned for other apps.

- App1 monitors accelerometer (waking/driving)
- App2 uses location sensors (at home/at work)
- Context History: Driving=true At home=false

ACE

- At 10am App1 asks the value of "Driving"
  - Determined through features of the accelerometer sensor
- Systems caches it
- At 10.05am App3 asks the value of "Driving"
  - Cache value is returned avoiding sensor sampling
- At 10.05 App2 asks the value of "AtHome"
  - Negative correlation in context history used to report value False if Driving is True in the cache

Best Attribute to Use

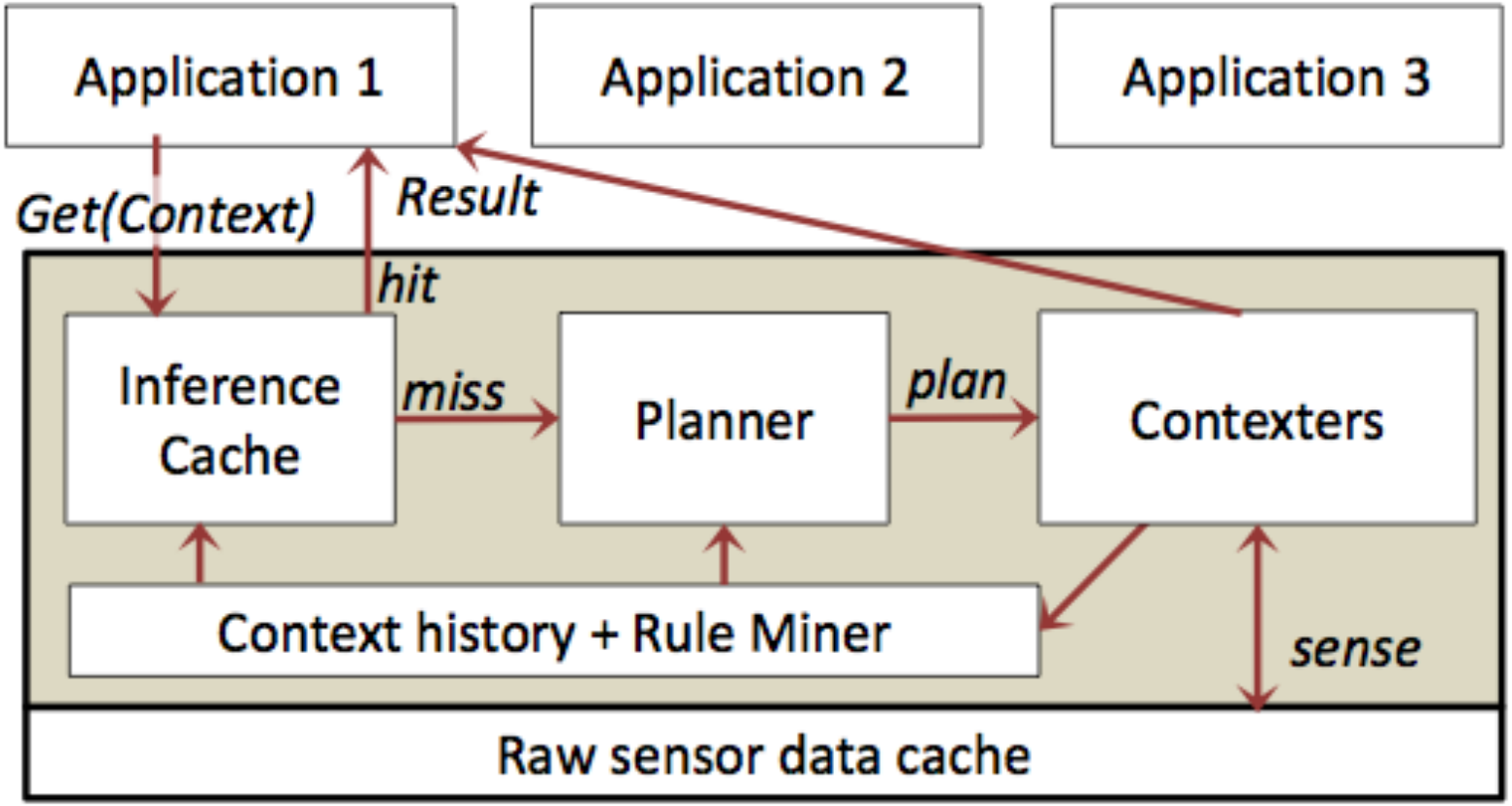
- Always use the cheapest attribute
- If Running=True or AtHome=True then InOffice=False

- Use the cheapest one cached to infer InOffice value

ACE

- How can these context correlations be learned?
- Do they even exist?

ACE System



UNIVERSITY OF CAMBRIDGE

Contexters and Rule Miner

- **Contexters**: Determining value of context with sensors by using inference algorithms
- Cache among the sensed values can be used to share among contexters instead of sensing

- **Rule Miner**: maintains user's context history and automatically learns relationships among various context attributes

Inference Cache and Planner

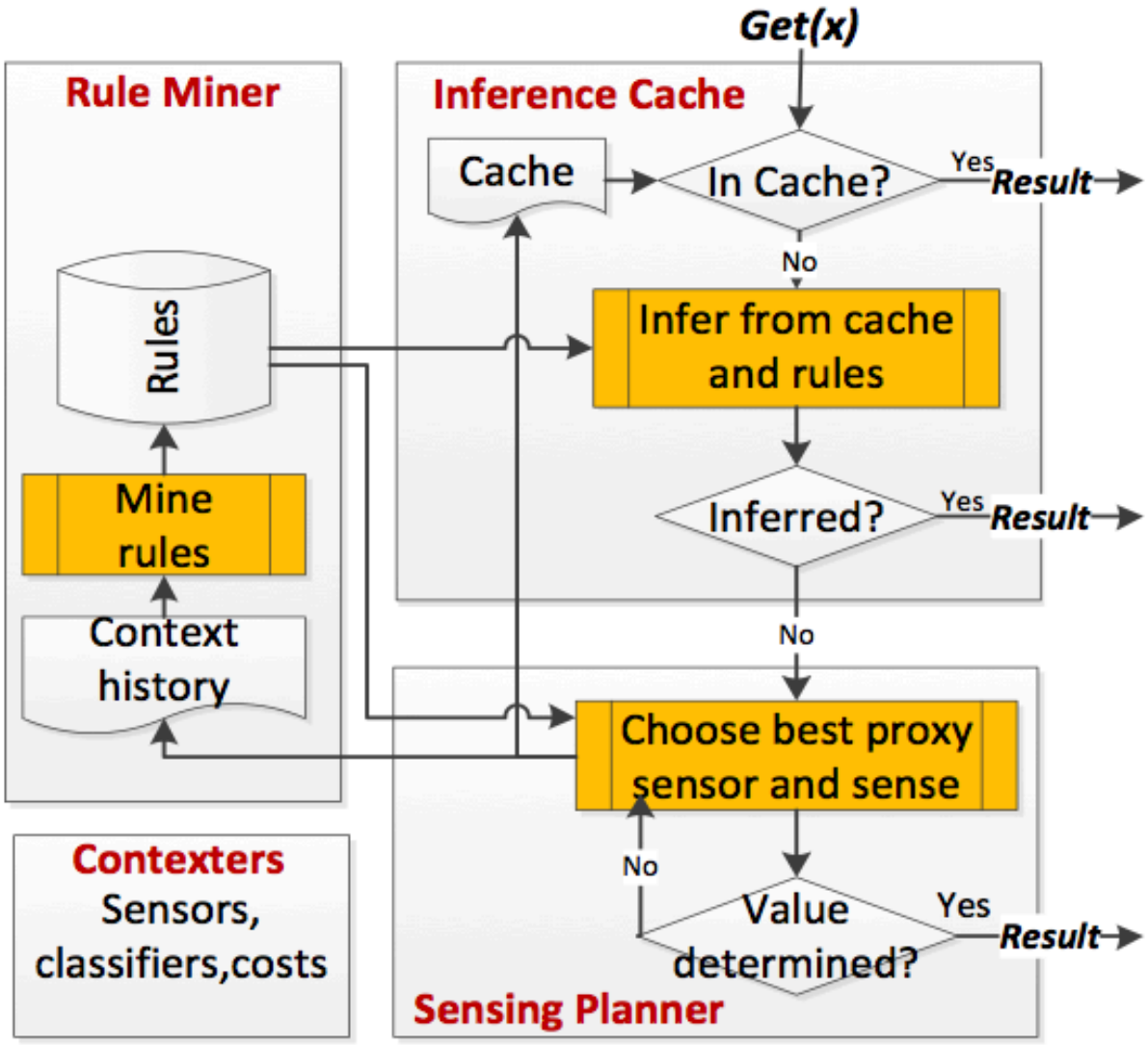
- **Inference Cache**: It returns a value not only if the raw sensor cache has a value but also if it can be inferred by using context rules and cached values of other attributes
- **Sensing Planner**: this finds the sequence of proxy attributes to speculatively sense to determine the value of the target attribute in the cheapest way.

Energy of context attributes

Table 1: Context attributes implemented in ACE

Attribute	Short	Sensors used (sample length)	Energy (mJ)
`IsWalking`	`W`	Accelerometer (10 sec)	259
`IsDriving`	`D`	Accelerometer (10 sec)	259
`IsJogging`	`J`	Accelerometer (10 sec)	259
`IsSitting`	`S`	Accelerometer (10 sec)	259
`AtHome`	`H`	WiFi	605
`InOffice`	`O`	WiFi	605
`IsIndoor`	`I`	GPS + WiFi	1985
`IsAlone`	`A`	Microphone (10 sec)	2895
`InMeeting`	`M`	WiFi + Microphone (10 sec)	3505
`IsWorking`	`R`	WiFi + Microphone (10 sec)	3505

Workflow



UNIVERSITY OF CAMBRIDGE

Learned Rules by Rule Miner

$\{\texttt{IsDriving} = \texttt{True}\} \Rightarrow \{\texttt{Indoor} = \texttt{False}\}$

$\{\texttt{Indoor} = \texttt{T}, \texttt{AtHome} = \texttt{F}, \texttt{IsAlone} = \texttt{T}\} \Rightarrow \{\texttt{InOffice} = \texttt{T}\}$

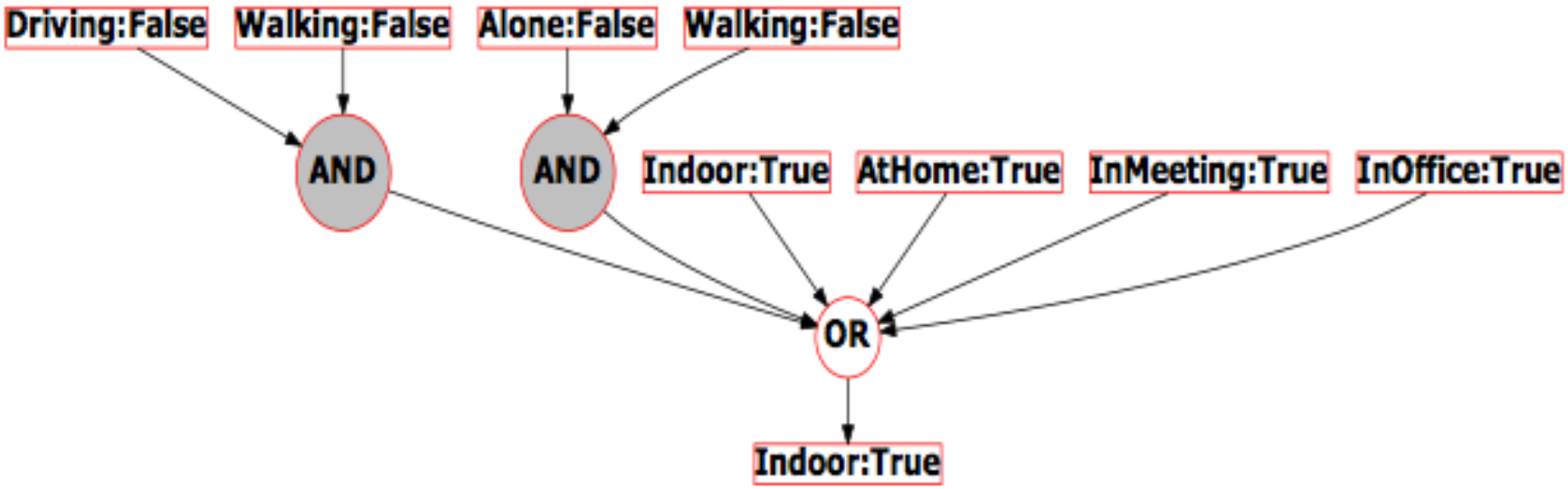
$\{\texttt{IsWalking} = \texttt{T}\} \Rightarrow \{\texttt{InMeeting} = \texttt{F}\}$

$\{\texttt{IsDriving} = \texttt{F}, \texttt{IsWalking} = \texttt{F}\} \Rightarrow \{\texttt{Indoor} = \texttt{T}\}$

$\{\texttt{AtHome} = \texttt{F}, \texttt{IsDriving} = \texttt{F}, \texttt{IsUsingApp} = \texttt{T}\} \Rightarrow \{\texttt{InOffice} = \texttt{T}\}$

$\{\texttt{IsJogging} = \texttt{T}\} \Rightarrow \{\texttt{AtHome} = \texttt{T}\}$

Rule Miner Simplification

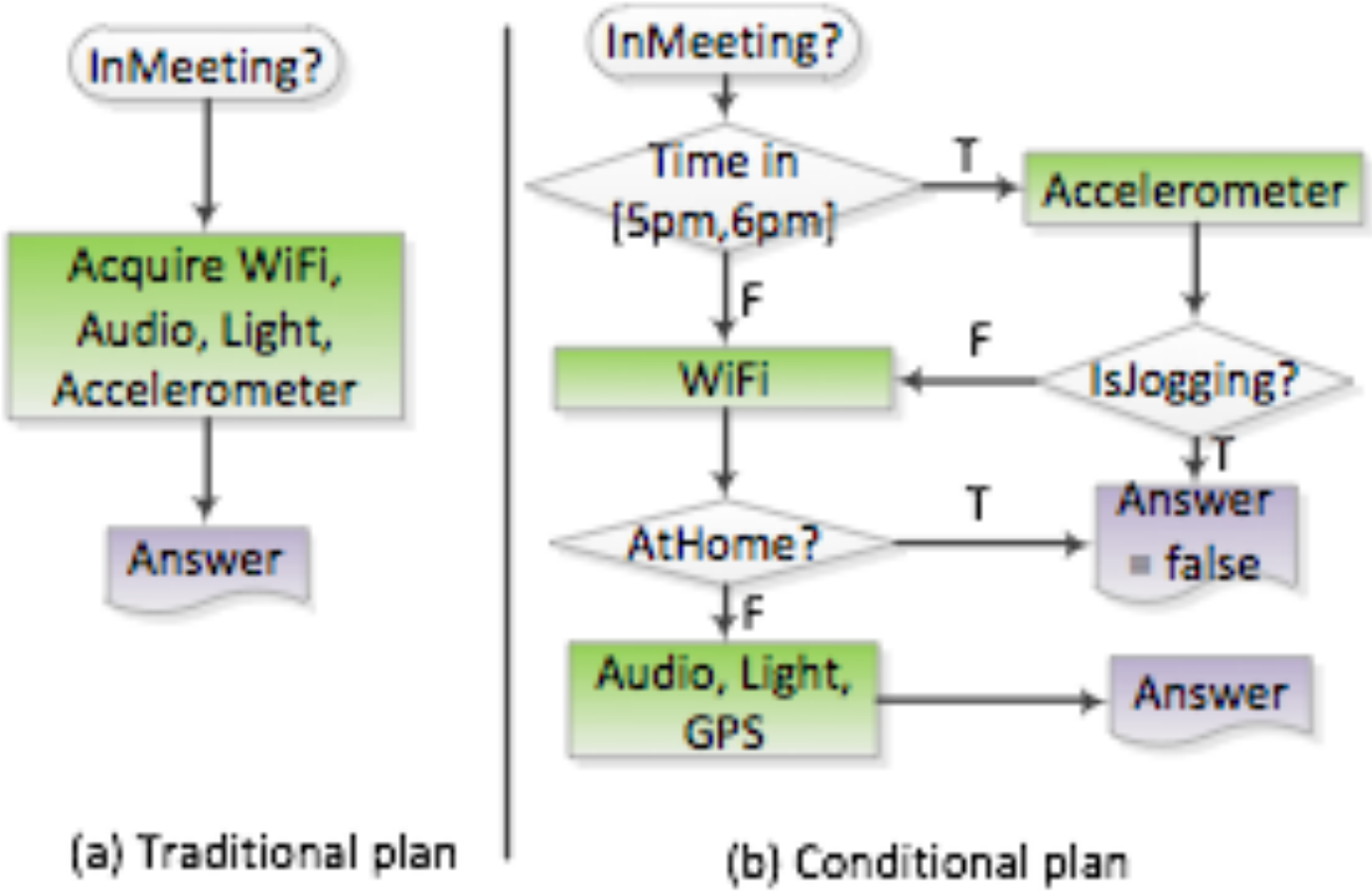


UNIVERSITY OF CAMBRIDGE

Rule Miner

- If in a context history of 1000 co-occurring contexts values where 200 contain both A and B and 80 include C then
  - {A,B}-> C with support 8% (80/1000) and confidence 40% (80/200)
  - Support is used to exclude non frequent associations
  - Confidence is used to tune on accuracy needed

Energy preserving Sensing Plan

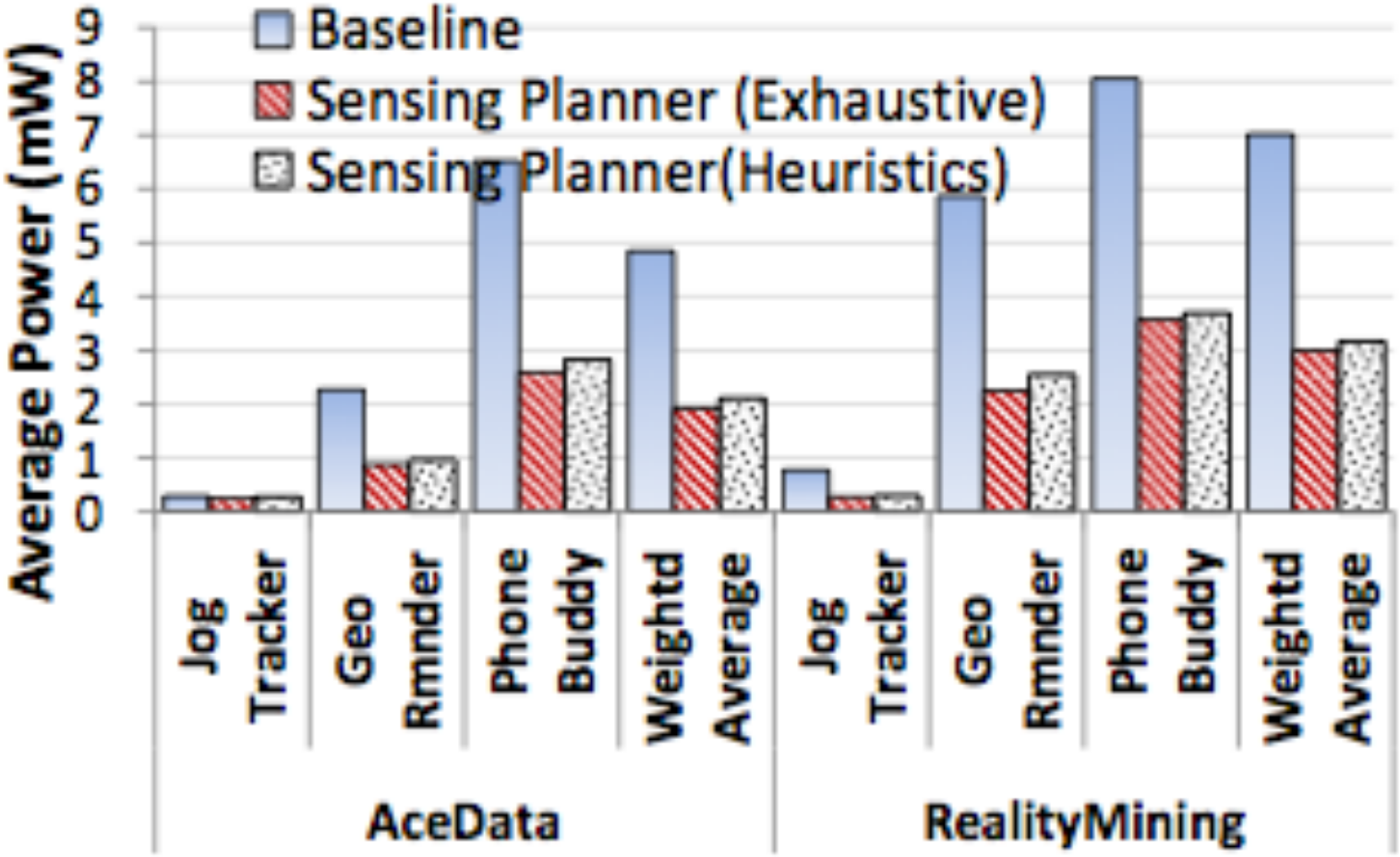


UNIVERSITY OF CAMBRIDGE

Sensing Planner Cost Finding

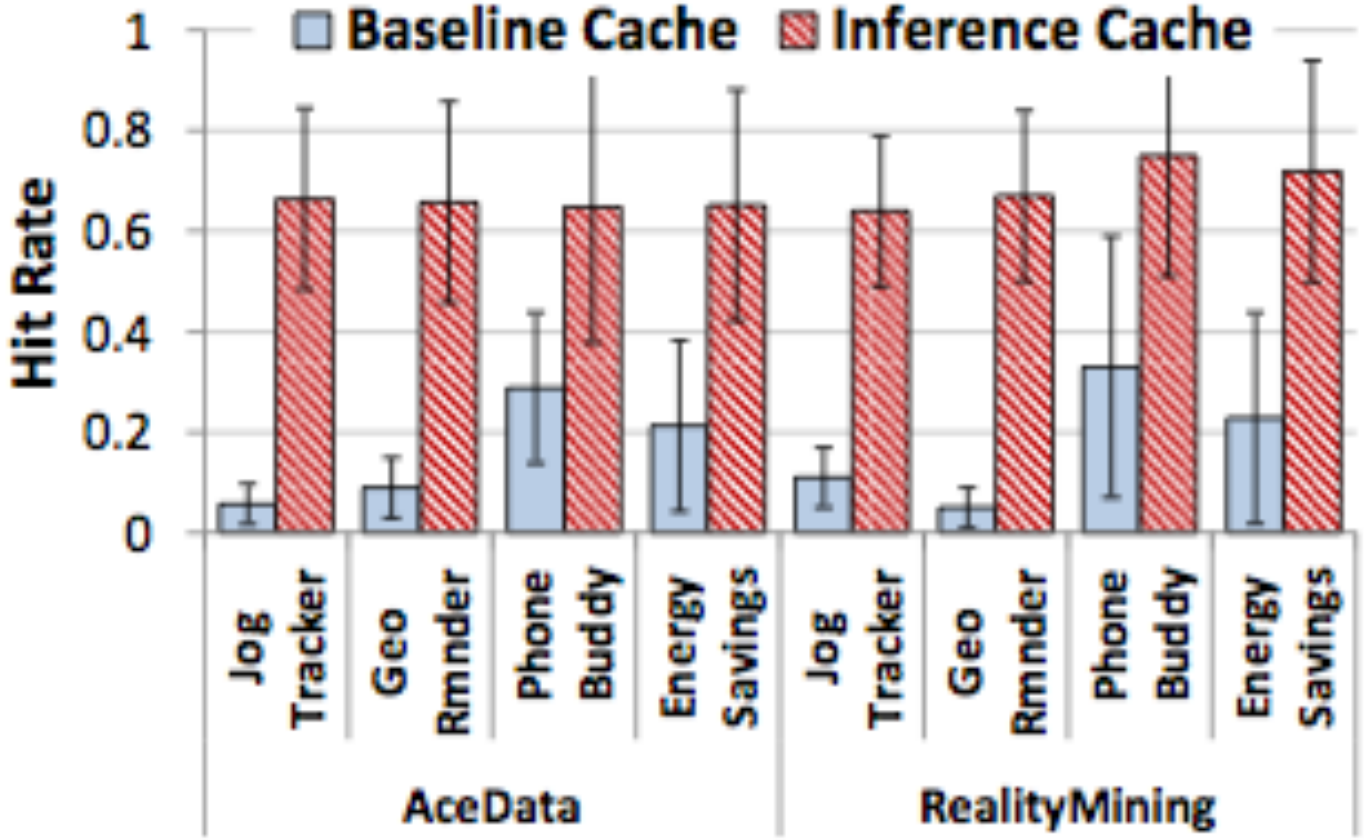
- The best solution is found by going through the costs of the various options
  - Constructed from the basic sensing values
  - Problem is NP Hard so for large number of attributes a heuristic is provided

Energy Consumed



UNIVERSITY OF CAMBRIDGE

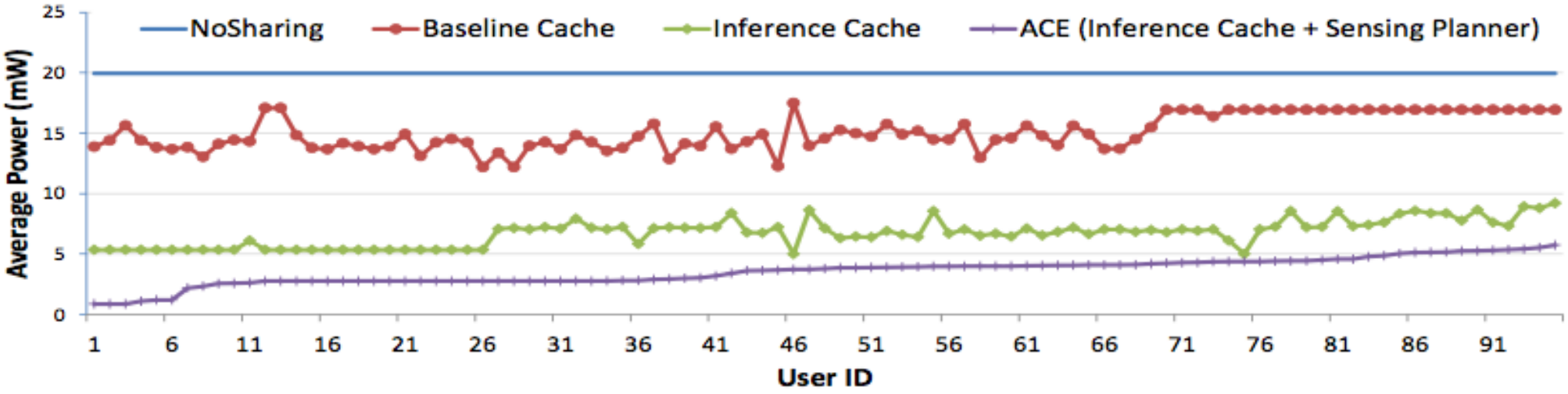
Caching Performance



UNIVERSITY OF CAMBRIDGE

baseline cache that shares sensor data and context attributes across applications

User Savings



UNIVERSITY OF CAMBRIDGE

Systems Energy Saving

- Most of current mobile sensing applications rely on cloud based computation but…
- code offloading can be quite costly...(energy and latency costs need to be considered)
- Local resources/computations are actually quite powerful

Networking Costs

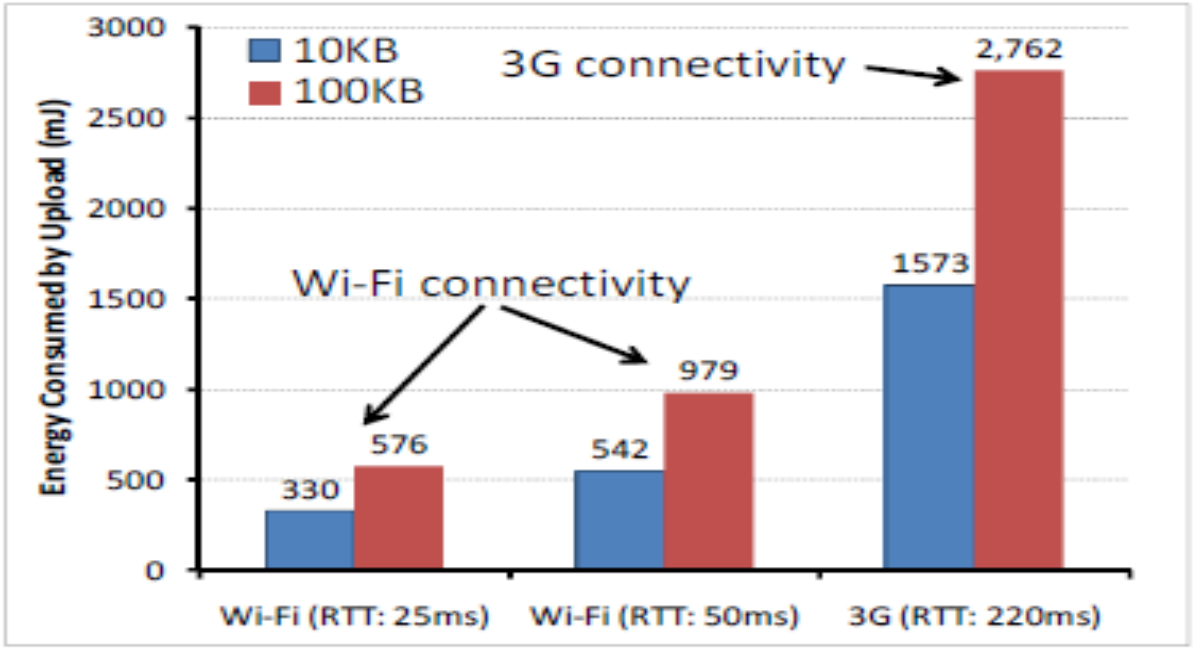


Figure 1: The Energy Consumption of Wi-Fi Connectivity vs. 3G Connectivity ***We performed 10 KB and 100 KB uploads from a smartphone to a remote server. We used Wi-Fi with RTTs of 25 ms and 50 ms (corresponding to the first two sets of bars) and 3G with an RTT of 220 ms (corresponding to the last bar).***

UNIVERSITY OF CAMBRIDGE

MAUI

- MAUI is a mobile device framework which profiles code components in terms of energy to decide if to run them locally or remotely (considering latency requirements).
  - Costs related to the transfer of code/data
  - Programming framework
  - Dynamic decisions based on network constraints
  - CPU only

MAUI Offloading

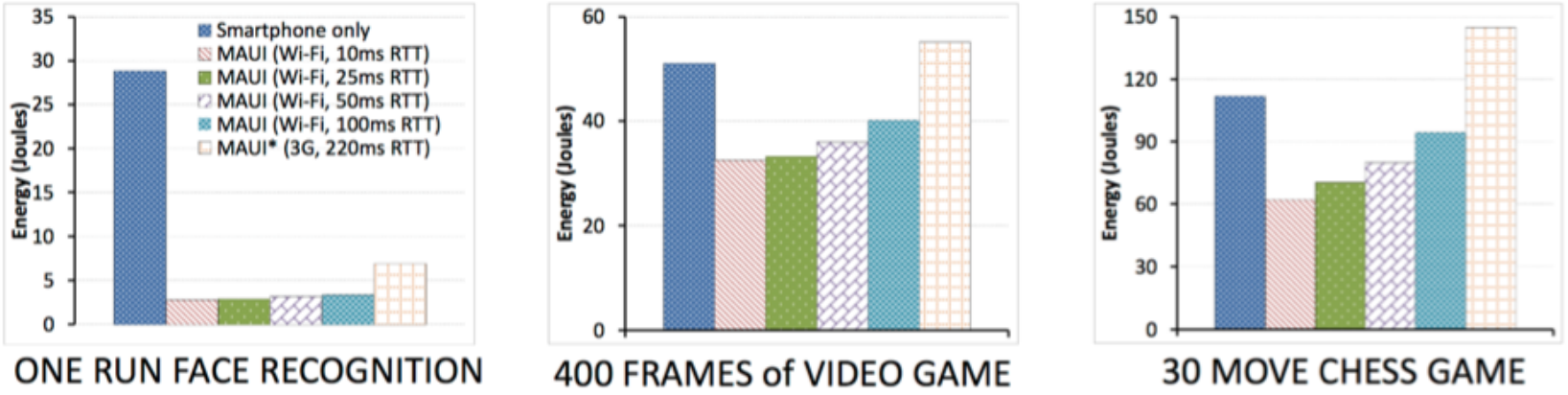
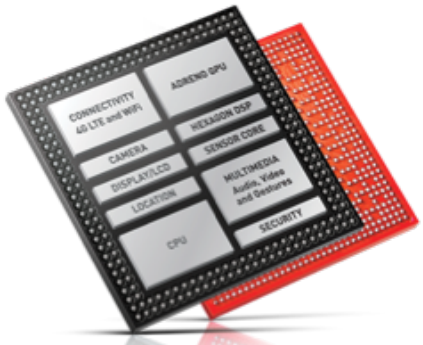


Figure 9: A comparison of MAUI's energy consumption. ***We compare the energy consumption of running three applications standalone on the smartphone versus using MAUI for remote execution to servers that are successively further away (the RTT is listed for each case). The graph on the left shows one run of the face recognition application; the graph in the middle shows running the video game for 400 frames; the graph on the right shows running the chess game for 30 moves. MAUI* is a slight modification to MAUI to bypass the optimizer and to always offload code. Without this modification, MAUI would have not performed code offload in the case of the video game and chess because offload ends up hurting energy performance.***

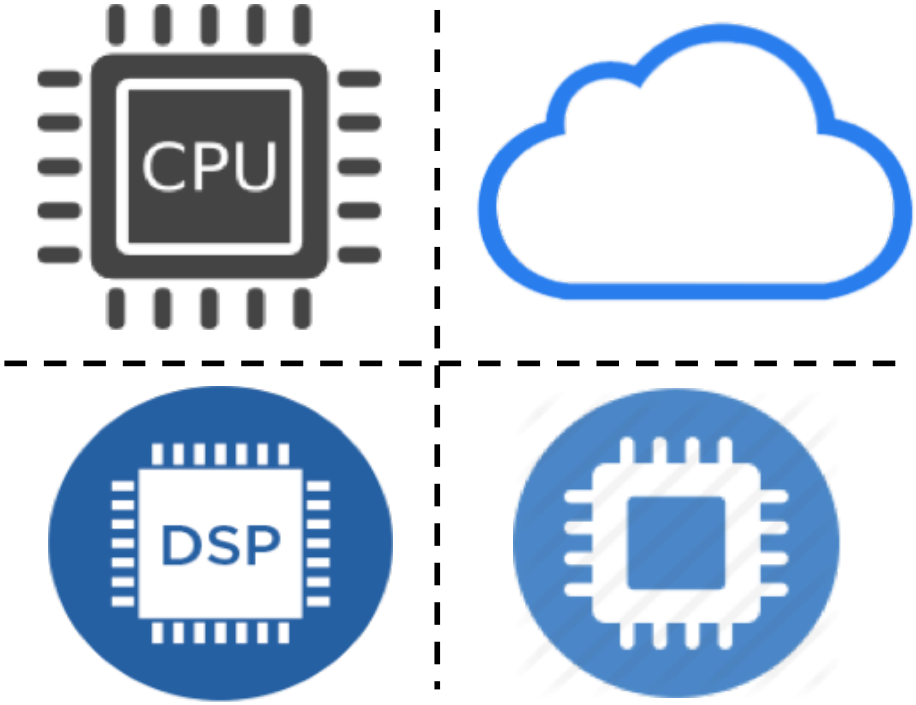
MAUI would not perform offloading with 3G…

UNIVERSITY OF CAMBRIDGE

Local Computation



- Not just the CPU…



Continuous Audio Sensing Applications

Emotion recognition

Speaker count

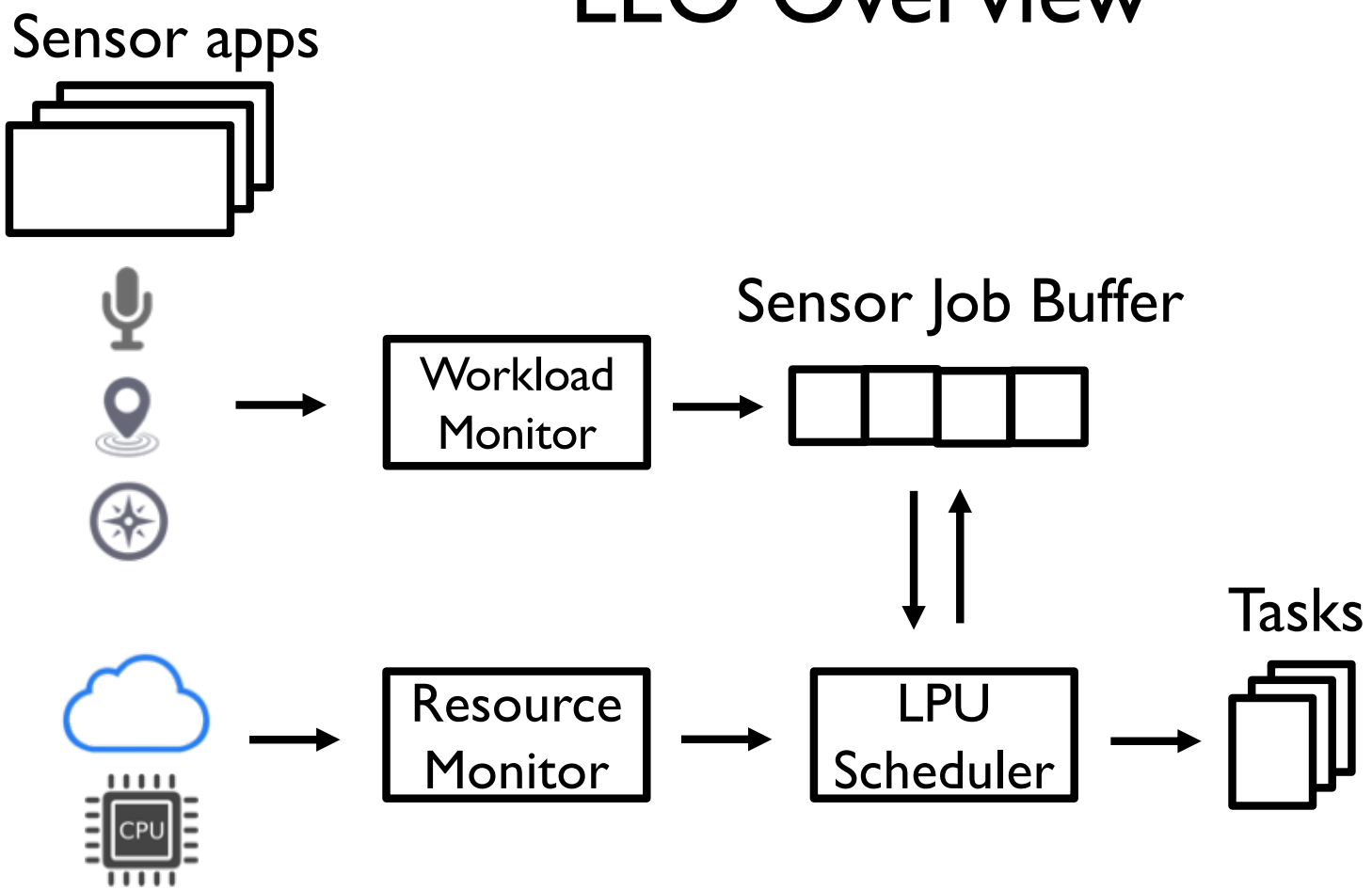
Speaker identification

Gender estimation

Ambient sound detection

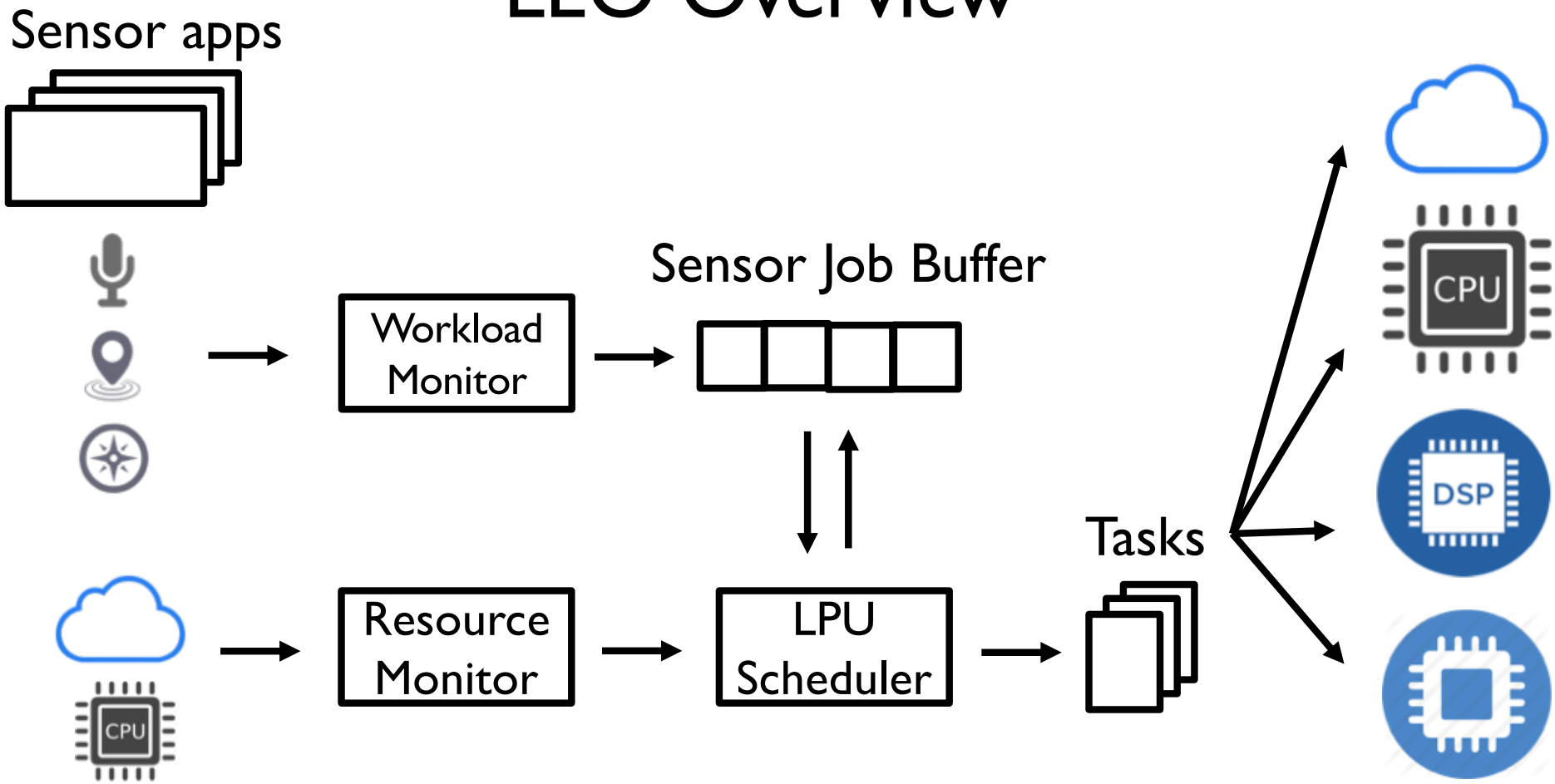
UNIVERSITY OF CAMBRIDGE

LEO Overview



UNIVERSITY OF CAMBRIDGE

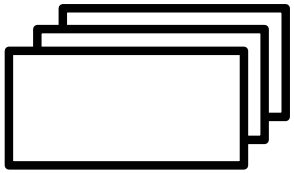
LEO Overview



UNIVERSITY OF CAMBRIDGE

Low overhead

- uses heuristics (fast runtime)
- runs on the LPU (low energy)



10 app workload

~100 ms

<0.5%

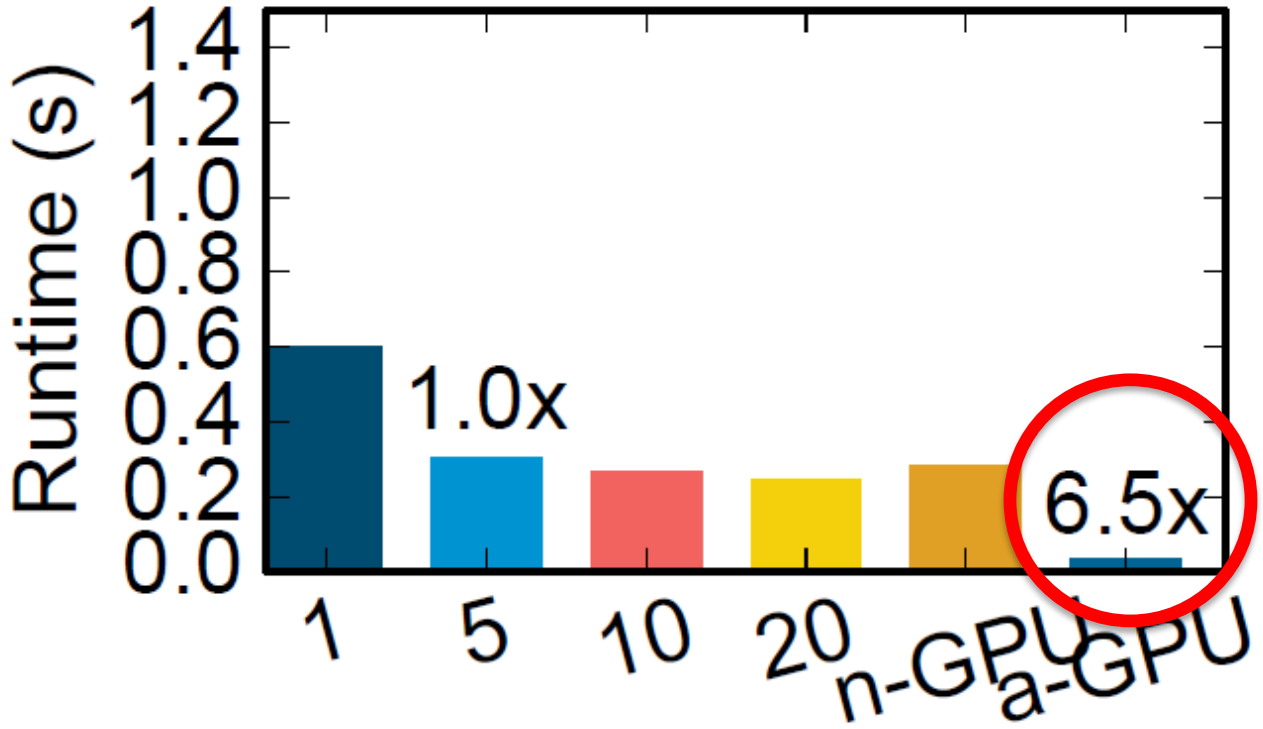
vs.

~3.5%

scheduling in cloud
(next best alternative)

Optimized GPU is Efficient

Optimized GPU is >6x faster than cloud



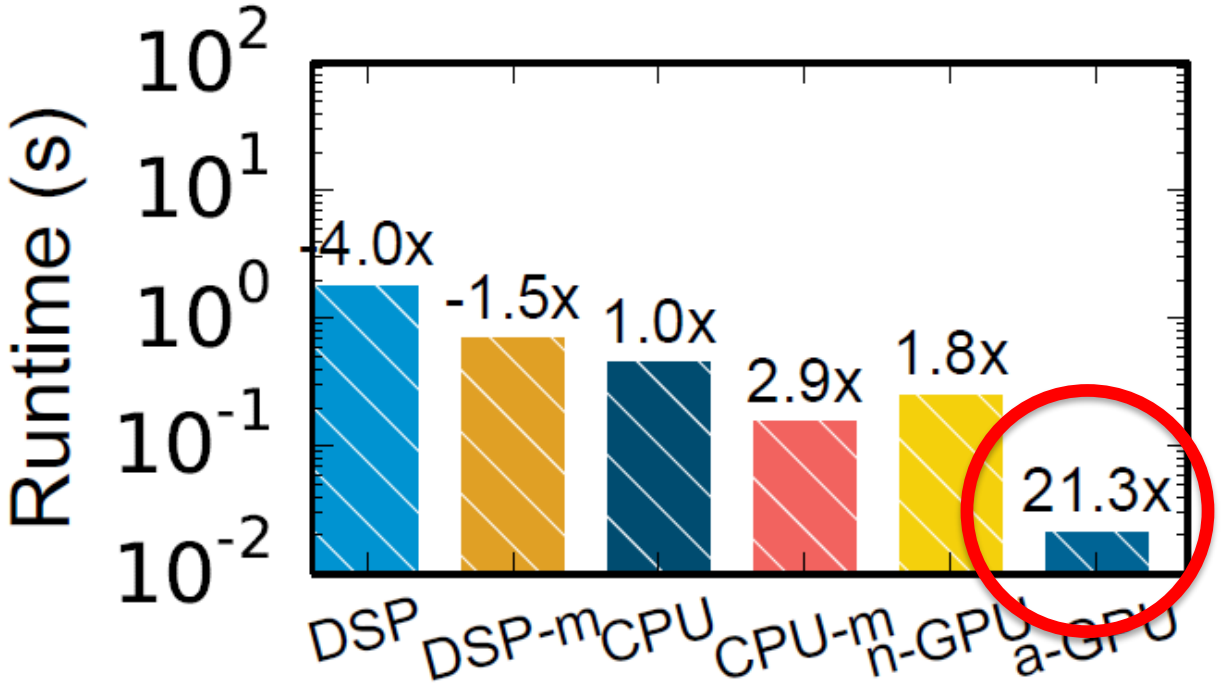
nGPU= naïve GPU usage
aGPU=optimized use of GPU

UNIVERSITY OF CAMBRIDGE

Keyword Spotting application

Optimized GPU is Efficient

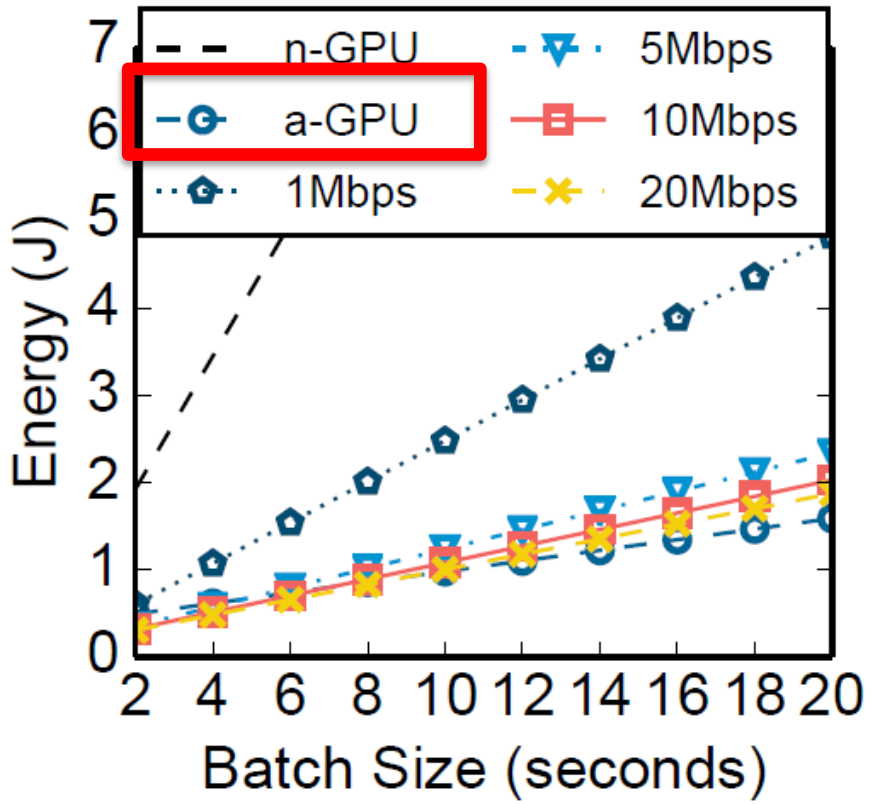
Optimized GPU is >21x faster than sequential CPU



UNIVERSITY OF CAMBRIDGE Keyword Spotting classification

Optimized GPU is Efficient

Optimized GPU with batching outperforms cloud energy-wise



UNIVERSITY OF CAMBRIDGE

Summary

- We have learned methods to avoid context sensing through correlation with other context
- We have studied how local resources and cloud offloading have an impact on energy efficiency and could be used to improve it.

References

- Suman Nath. 2012. ACE: exploiting correlation for energy-efficient and continuous context sensing. In *Proceedings of the 10th international conference on Mobile systems, applications, and services* (MobiSys '12). ACM, New York, NY, USA, 29-42.
- Eduardo Cuervo, Aruna Balasubramanian, Dae-ki Cho, Alec Wolman, Stefan Saroiu, Ranveer Chandra, and Paramvir Bahl. 2010. MAUI: making smartphones last longer with code offload. In Proceedings of the 8th international conference on Mobile systems, applications, and services (MobiSys '10). ACM, New York, NY, USA, 49-62.
- Petko Georgiev, Nicholas D. Lane, Kiran K. Rachuri, and Cecilia Mascolo. 2016. LEO: scheduling sensor inference algorithms across heterogeneous mobile processors and network resources. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking* (MobiCom '16). ACM, New York, NY, USA, 320-333.
- Petko Georgiev, Nicholas Lane, Cecilia Mascolo, David Chu. Accelerating Mobile Audio Sensing Algorithms through On-Chip GPU Offloading. In Proceedings of 15th ACM International Conference on Mobile Systems, Applications and Services (Mobisys 2017). Niagara Falls, NY. USA. June 2017.