Energy in Social Sensing and CPS-I

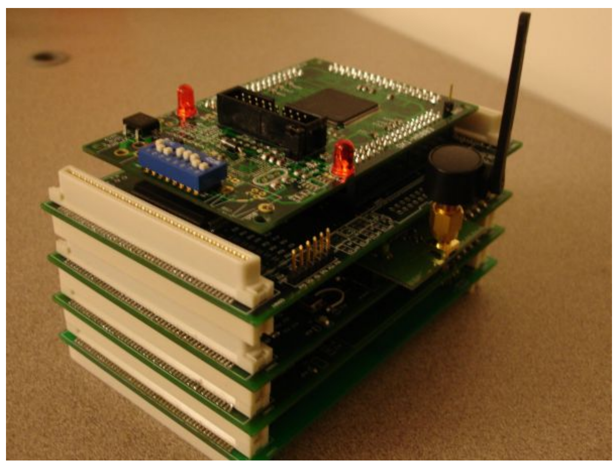
Sensor nodes and Smartphones

CSE 40437/60437-Spring 2015

Prof. Dong Wang

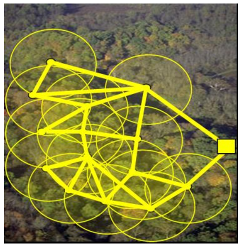
Papers

- Paper 1: "Energy-optimal Batching periods for asynchronous multistage data processing on sensor nodes: foundations and an mPlatform case study." Wang, Dong, et al. Real-Time Systems 48.2 (2012): 135-165.



Background

- **Energy**
  - primary concern in sensor network
  - much energy consumed in idle state
  - build more energy economic processor



- **Time**:
  - critical to real time and control related tasks
  - specified as end to end deadline

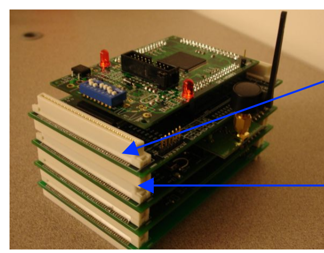
Background

- **Microserver**
  - Fast speed

- **Mot**
  -
  -
  -

Can the *high-end processor* be used to Save Energy rather than increasing computation speed?

- **mPlatform**
  - Heterogeneous platform
  - Multi-processor boards, multiple radios



Background

	High-end Processor	Low-end Processor
Active Power	High	Low
Speed	Fast	Slow
Wakeup Cost	High	Low

Q: Do you have some intuition how high-end processor can be used to save energy when it is used to process a batch of data?

Background

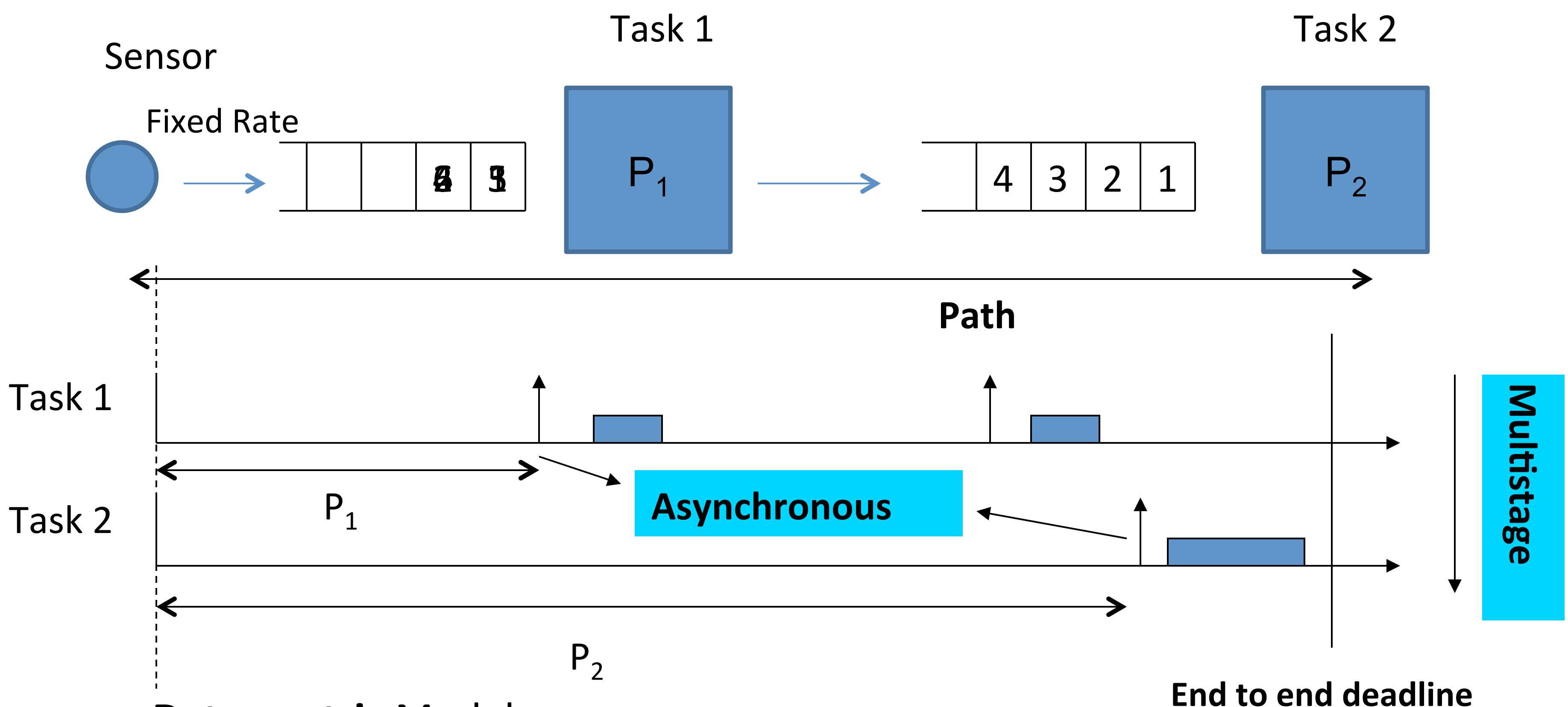
	High-end Processor	Low-end Processor
Active Power	High	Low
Speed	***Disproportionately Faster***	Slow
Energy/Unit	**Low**	High
Wakeup Cost	**High**	Low

Process data **in batches** to save energy!

Key Challenge: **Batching Period** should be carefully designed to

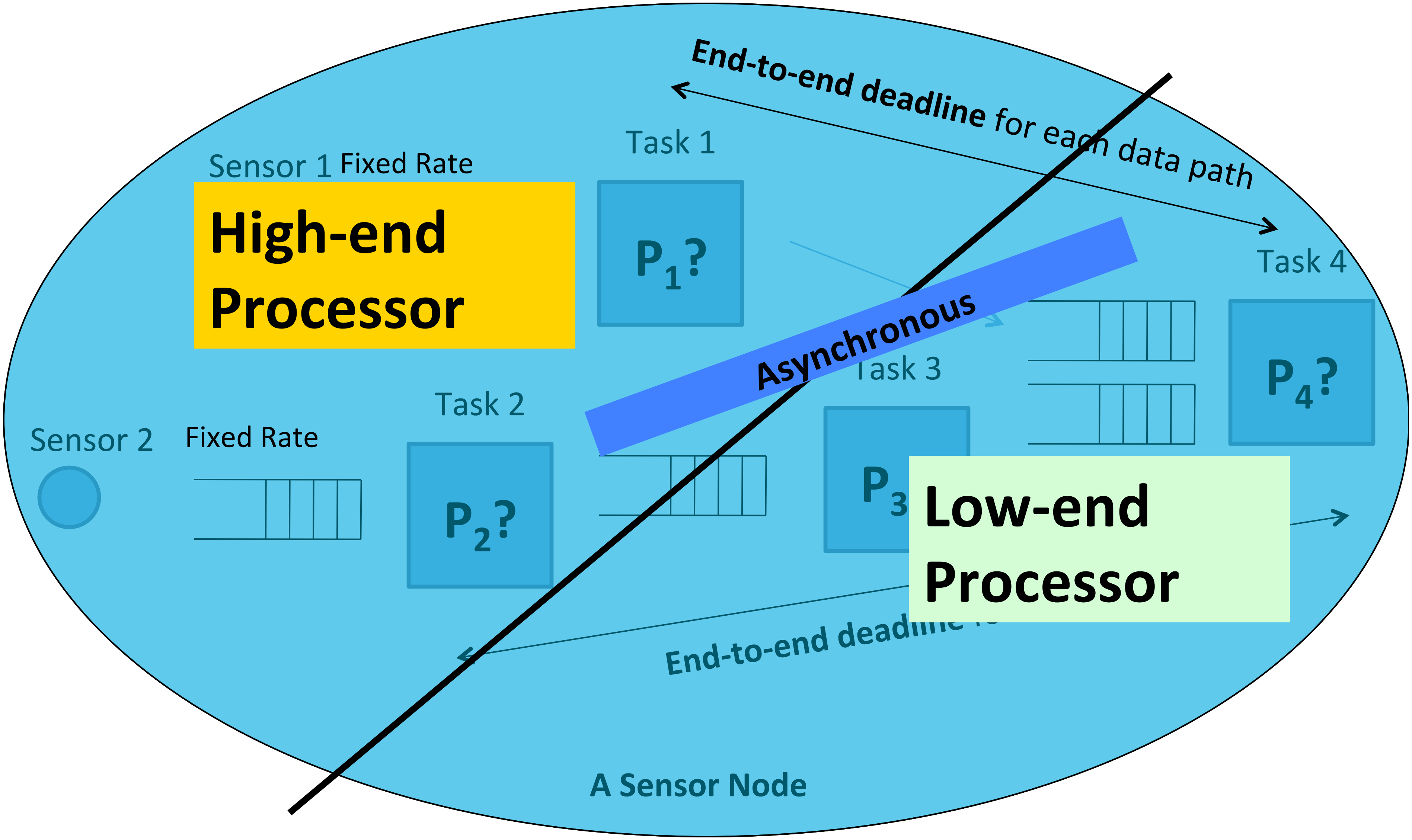
1. **Exploit heterogeneity to minimize energy** consumption
2. **Meet time requirement** of the data processing

Model



- **Data-centric** Model
- Tasks run on **batching period** to process data
- **Asynchronous**
- **Multistage**

A Task Set Example



End to End Deadline: Associated with a Path

Problem Statement

- Energy to execute a task on processor k:

$$E_i^k = E_{wakeup\_i}^{\;k} + E_{datarate\_i}^{\;k}$$

$E_{wakeup\_i}^{\;k}$: data **independent** cost (wakeup, state storage)

$E_{datarate\_i}^{\;k}$: data **dependent** cost (computation, data transfer)

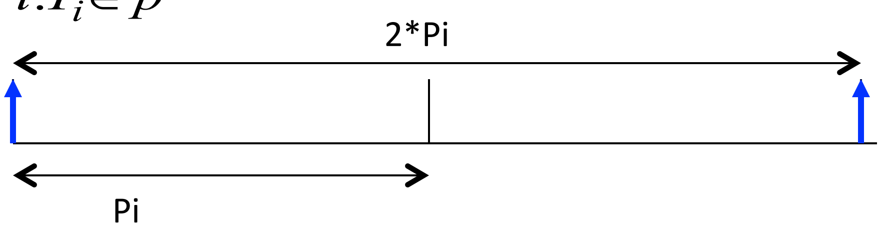
- Average power to execute a task:

$$W_i^k = \frac{E_{wakeup\_i}^{\;k}}{P_i} + W_{datarate\_i}^{\;k} \qquad W_{datarate\_i}^{\;k} = \frac{E_{datarate\_i}^{\;k}}{P_i}$$

P_i : Batching period of task i on processor k

$$2\sum_{i:T_i \in p} P_i \le D_p$$

D_p : End-end deadline on path p



Problem Statement

- **Goal**: to find *optimal batching period* P_i^* for each task to **minimize**

$$W = \sum_{1 \le i \le n} \left(\frac{E_{wakeup\_i}^{\;k}}{P_i} + W_{datarate\_i}^{\;k} \right)$$

- **Subject** to the constraint

$$\sum_{i:T_i \in p} P_i \le D_p / 2$$

Optimal Batching Period

- Method: Lagrange function

$$L = \sum_{1 \le i \le n} \left(\frac{E_{wakeup\_i}^{k}}{P_i} + W_{datarate\_i}^{k} \right) + \sum_{1 \le p \le m} \lambda_p \left(\sum_{i:T_i \in p} P_i - D_p / 2 \right)$$

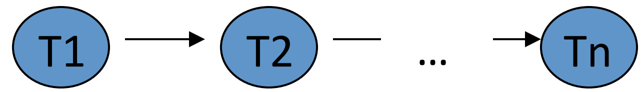
- Solution:

$$P_i^* = \sqrt{\frac{E_{wakeup\_i}^{k}}{2 \sum_{p:T_i \in p} \lambda_p}} \qquad \sum_{i:T_i \in p} P_i^* = D_p / 2$$

Solutions can be computed numerically

For particular task allocation: Ignore task index for notation simplicity

Chain Task Topology



- Chain topology:
  - n tasks $T_1 \ldots T_n$ form a single path p

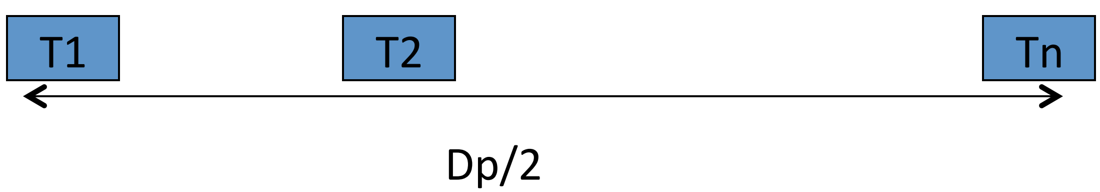
- Optimal Batching Period:

$$P_i^* = \frac{\sqrt{E_{wakeup\_i}}}{\sum_{i:T_i \in p} \sqrt{E_{wakeup\_i}}} \frac{D_p}{2}$$

- Theorem1: **Chain Period Allocation**:

$$P1 : P2 : \ldots : Pn$$

$$(E_{wakeup\_1})^{1/2} : (E_{wakeup\_2})^{1/2} : \ldots : (E_{wakeup\_n})^{1/2}$$



Chain Task Topology

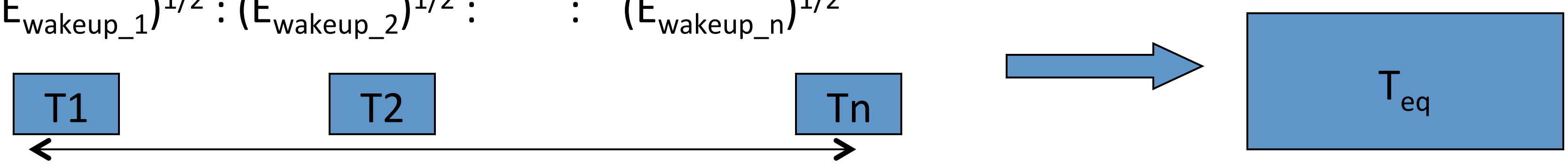
- Theorem 2: **Chain Reduction**

$$\frac{E_{wakeup\_eq}}{P^*_{eq}} + W_{datarate\_i} = \frac{\left(\sum_{1\le i\le n}\sqrt{E_{wakeup\_i}}\right)^2}{D_p/2} + \sum_{1\le i\le n} W_{wakeup\_i}$$

P1 : P2 : ... : Pn

$(E_{wakeup\_1})^{1/2} : (E_{wakeup\_2})^{1/2} : \quad : (E_{wakeup\_n})^{1/2}$

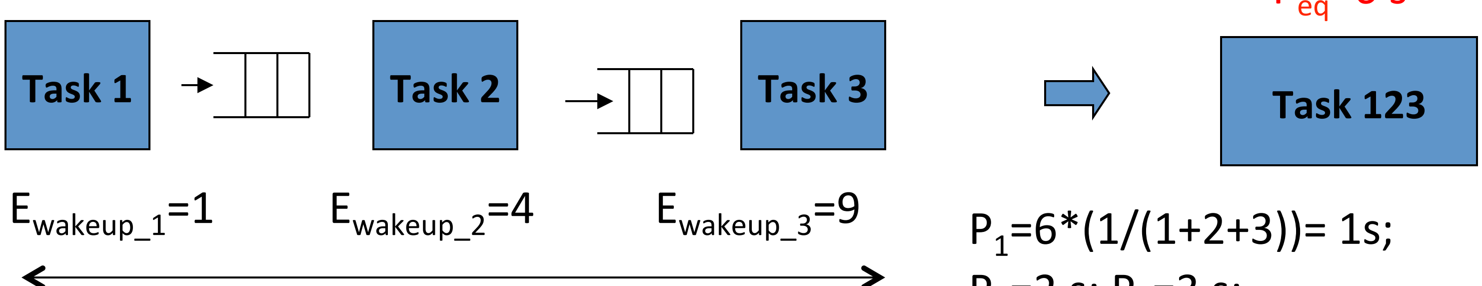
$E_{wakeup\_eq}=\{\mathrm{Sum}\,(E_{wakeup\_i})^{1/2}\}^2$



Dp/2

$P_{eq}=D_p/2$

$P_1=1$ s $\quad P_2=2$ s $\quad P_3=3$ s $\quad P_{eq}=6$ s



$E_{wakeup\_1}=1 \quad E_{wakeup\_2}=4 \quad E_{wakeup\_3}=9 \quad E_{wakeup\_123}=36$

Dp=12 s

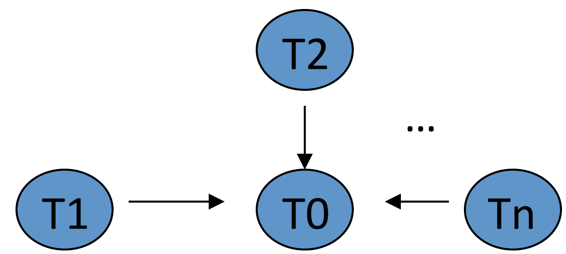
$P_1=6*(1/(1+2+3))= 1$s;

$P_2=2$ s; $P_3=3$ s;

$E_{wakeup\_123}=36$; $P_{eq}=6$ s

Star Task Topology

- Star Topology
  - Outputs of n tasks $T_1 \ldots T_n$ (leaf tasks) are inputs to a single task T_0 (aggregator task)
  - Assume: all paths have same D_p



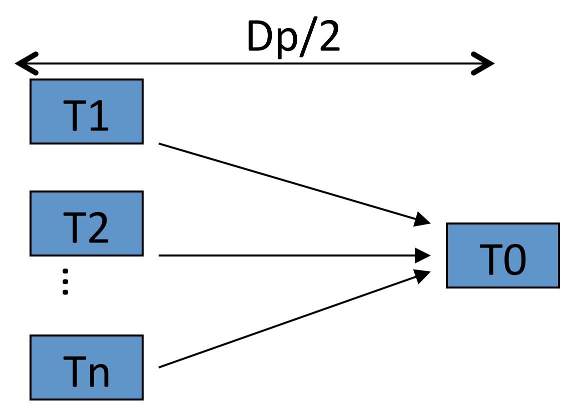
- Optimal Batching Period:

$$P_i^* = \frac{\sqrt{\sum_{1\le j\le n} E_{wakeup\_j}}}{\sqrt{\sum_{1\le j\le n} E_{wakeup\_j}} + \sqrt{E_{wakeup\_0}}} \frac{D_p}{2}$$

$$P_0^* = \frac{\sqrt{E_{wakeup\_0}}}{\sqrt{\sum_{1\le j\le n} E_{wakeup\_j}} + \sqrt{E_{wakeup\_0}}} \frac{D_p}{2}$$

- Theorem 3:

Star Period Allocation:



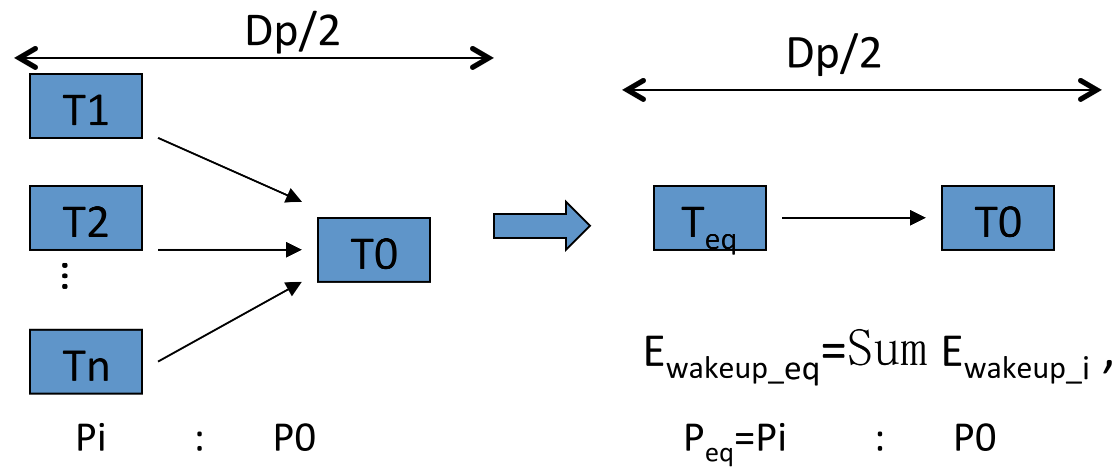
Pi : P0

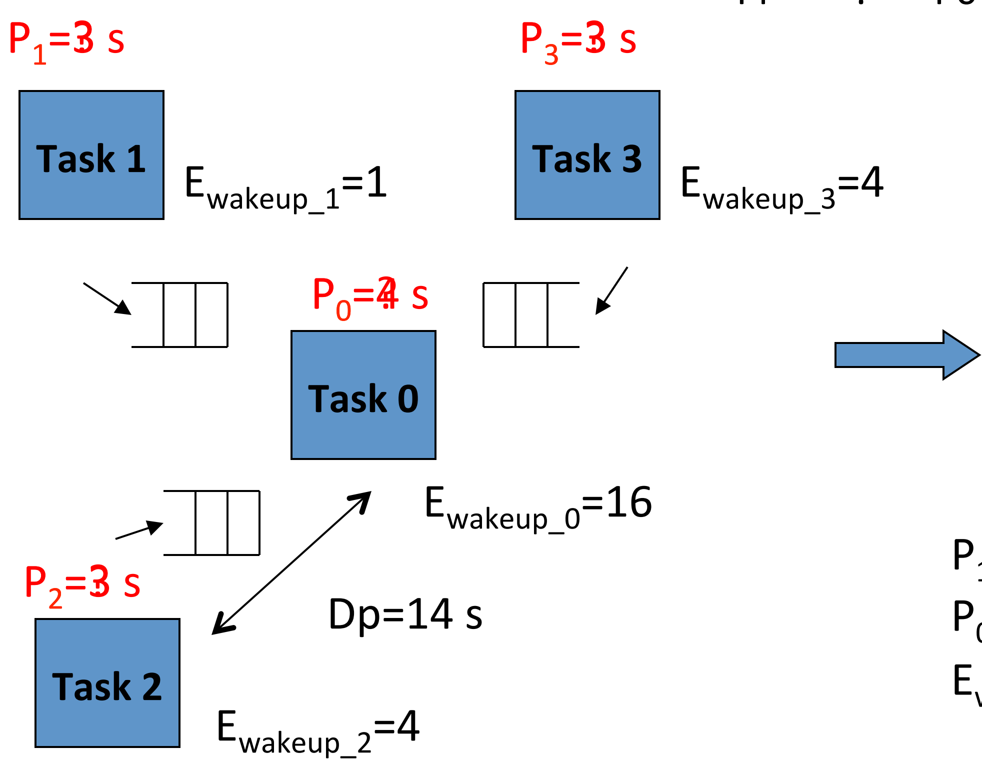
$(\mathrm{Sum}(E_{wakeup\_j})^{1/2})$: $(E_{wakeup\_0})^{1/2}$

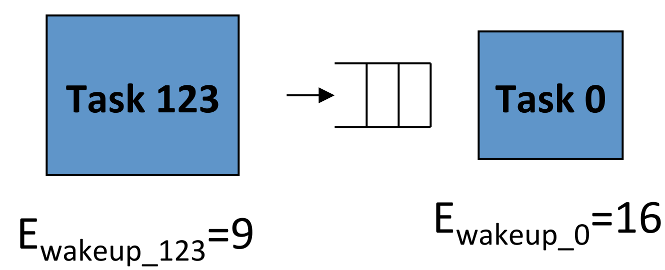
Star Task Topology

- Theorem 4

Star Reduction





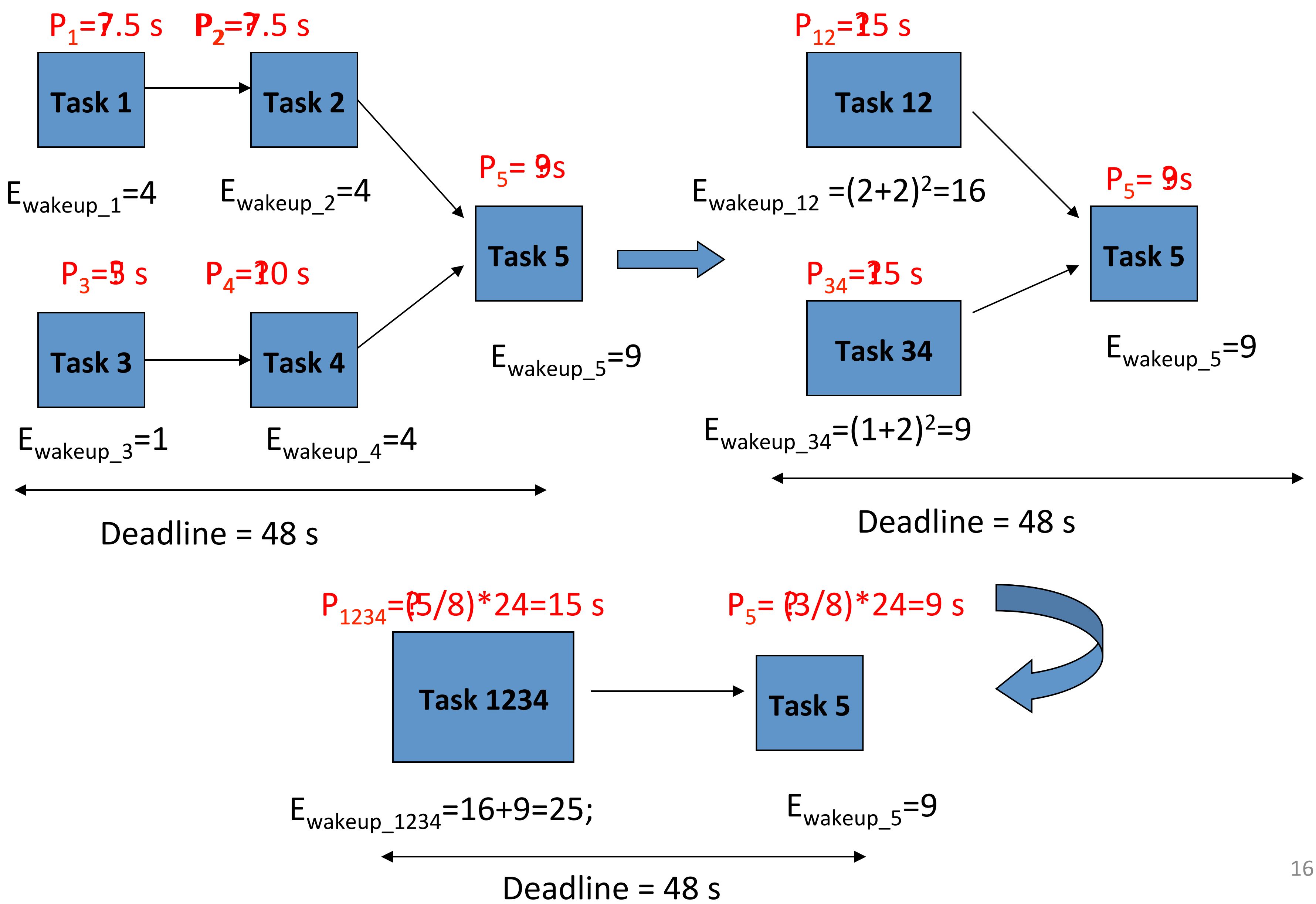


$P_1=P_2=P_3=7*(9^{1/2}/(9^{1/2}+16^{1/2}))=3$ s;

$P_0=7-P_3=4$ s

$E_{wakeup\_123}=9$;

Period Allocation in Aggregation Tree



Heterogeneous Task to Processor Assignment

- Parameters E_{wakeup_i}, $W_{datarate\_i}$ are processor dependent
- Period allocation: not entirely separable from task-processor assignment
- In general, number of tasks on sensor nodes are quite limited (e.g. 5-10)
- Run optimal period allocation for each possible task-processor allocation, find optimal solution
- Simple Heuristics can be derived to find optimal solution with high probability

Task to Processor Assignment Heuristics

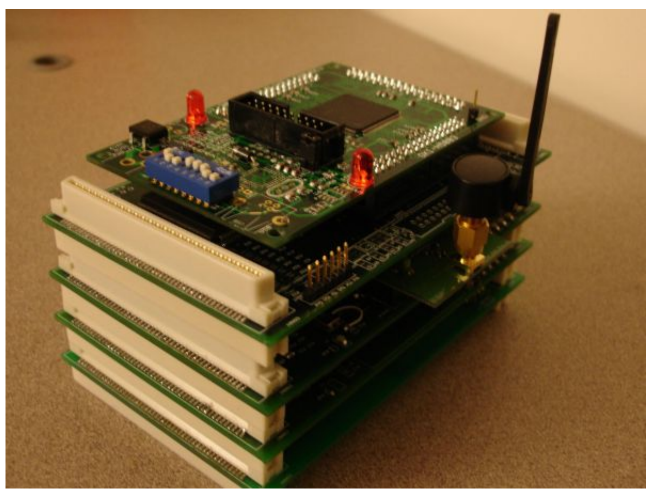
$$\frac{E_{wakeup\_i}^{hi}}{P_i^*} + W_{datarate\_i}^{hi} < \frac{E_{wakeup\_i}^{lo}}{P_i^*} + W_{datarate\_i}^{lo} \quad (*)$$

1. Run optimal batching period allocation assuming all tasks are allocated to higher-end processor (ARM)
2. Test resulting optimal batching periods for satisfying inequality (*). If a task Ti fails the test, move to lower-end processor (MSP)
3. Repeat optimal batching period allocation based on new task-processor assignment, check to see if it is different from the one got before step 2: different-go back to step2; same: reach (locally) optimal solution

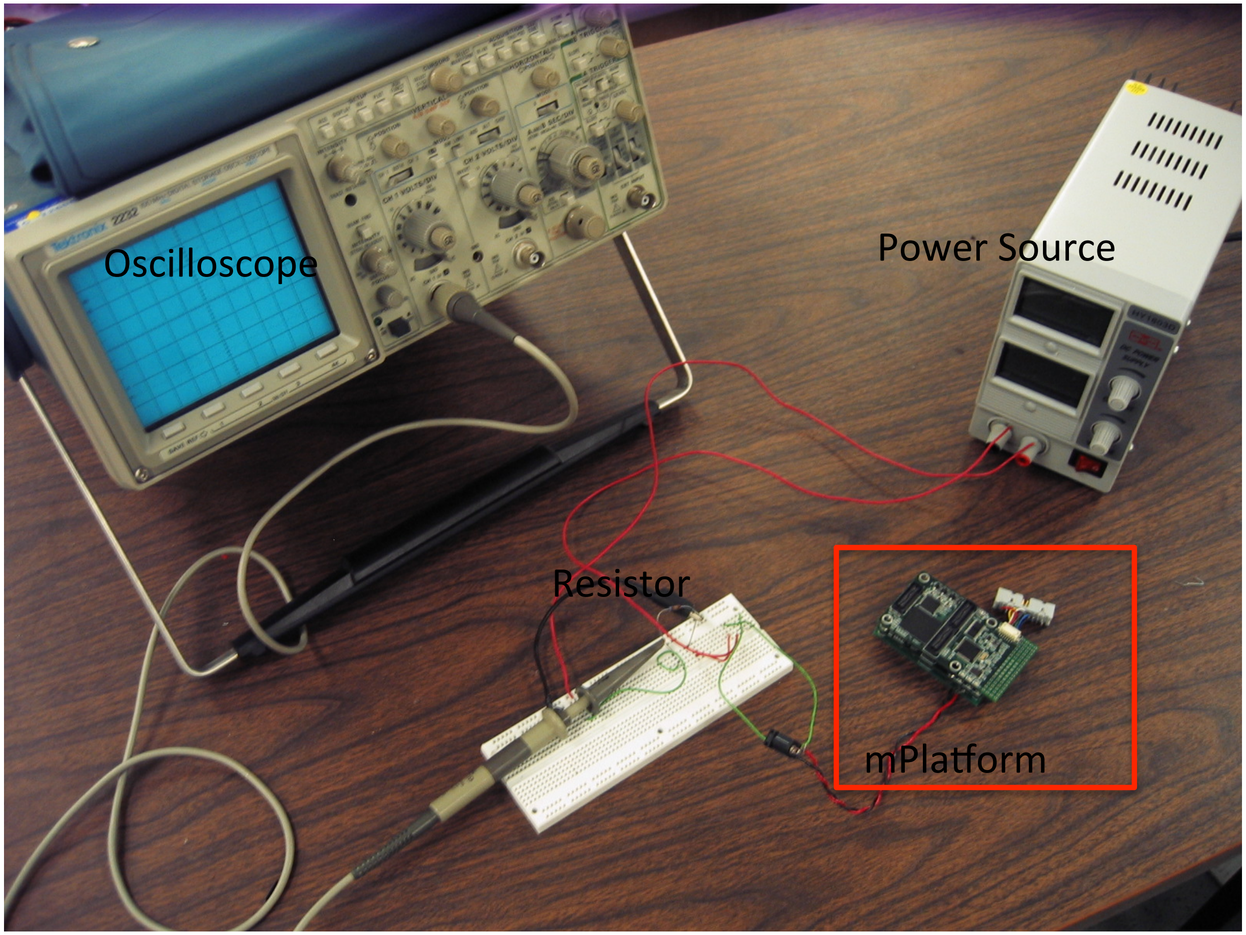
Note: the allocation found by heuristics is locally optimal, but it has a high probability to find global optimal

Evaluation

- Evaluation platform: mPlatform
  - Heterogeneous sensor platform: multiple processor boards, multi radios
  - MSP Board: MSP430F2618 processor
  - ARM Board: LPC2138 processor



Experiment Setup



Energy Profile

Parameter	MSP	ARM
Frequency	16MHz	60MHz
Active Current (mA)	8.61	75
Active Power (mW)	38.745	337.5
Sleep Current (μA)	17	150
Sleep Power (μW)	76.5	675
Wakeup time (ms)	0.7	3
Wakeup energy (μJ)	7.43	217.4
Flash access energy $(\mu J/byte)$	0.826	1.422
Inter-board Transfer time $(\mu s/byte)$	2	
Inter-board Transfer energy $(\mu J/byte)$	0.65	
Sensing Energy $(\mu J/byte)$	1.64	

Energy Profiling Comparison of MSP Board and ARM Board.
Board Supply Voltage is 4.5V.

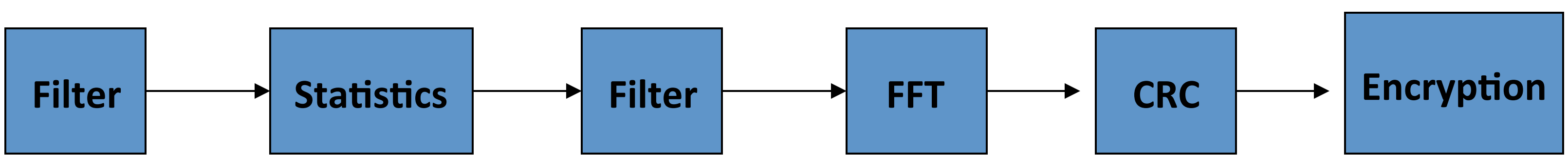
Comparison of Basic Operation on Two Processor Boards

OPERATION		Data Type	ARM Time(μs)	ARM Energy (μJ)	MSP Time(μs)	MSP Energy (μJ)
ARITHMETIC	Multiply	uint_32	0.66	**0.22275**	16.2	0.62767
		uint_16	0.66	**0.22275**	9.8	0.37970
		float	1.21	**0.40838**	20.6	0.79815
		double	1.9	**0.64125**	20.9	0.80977
	Divide	uint_32	1.12	**0.378**	26.5	1.02674
		uint_16	1.12	**0.378**	10.1	0.39132
		float	2.45	**0.82688**	26.2	1.01512
		double	8.32	2.808	26.2	**1.01512**
	Add	uint_32	0.61	0.20588	2.2	**0.08524**
		uint_16	0.66	0.22275	1.4	**0.05424**
		float	1.5	0.50625	10.1	**0.39132**
		double	2.1	0.70875	10.2	**0.3952**
	Subtract	uint_32	0.61	0.20588	2.2	**0.08524**
		uint_16	0.66	0.22275	1.4	**0.05424**
		float	1.5	0.50625	10.1	**0.39132**
		double	2.2	0.7425	10.2	**0.3952**
BIT OPERATION	AND	uint_32	0.48	0.162	1.6	**0.06199**
		uint_16	0.48	0.162	1.2	**0.04649**
	OR	uint_32	0.48	0.162	1.68	**0.06509**
		uint_16	0.49	0.16538	1.2	**0.04649**
	XOR	uint_32	0.49	0.16538	1.6	**0.06199**
		uint_16	0.49	0.16538	1.2	**0.04649**
	SHIFT	uint_32	0.46	0.15525	3.7	**0.14336**
		uint_16	0.5	0.16875	3.4	**0.13173**
RELATION	$\leq \geq$ $\equiv \neq$	uint_32	0.64	0.216	2.4	**0.09299**
		uint_16	0.68	0.2295	1.7	**0.06587**
		float	1.18	0.39825	3.6	**0.13948**
		double	1.35	0.45563	3.6	**0.13948**
LOGIC	AND OR NOT	All	0.31	0.10463	0.7	**0.02712**

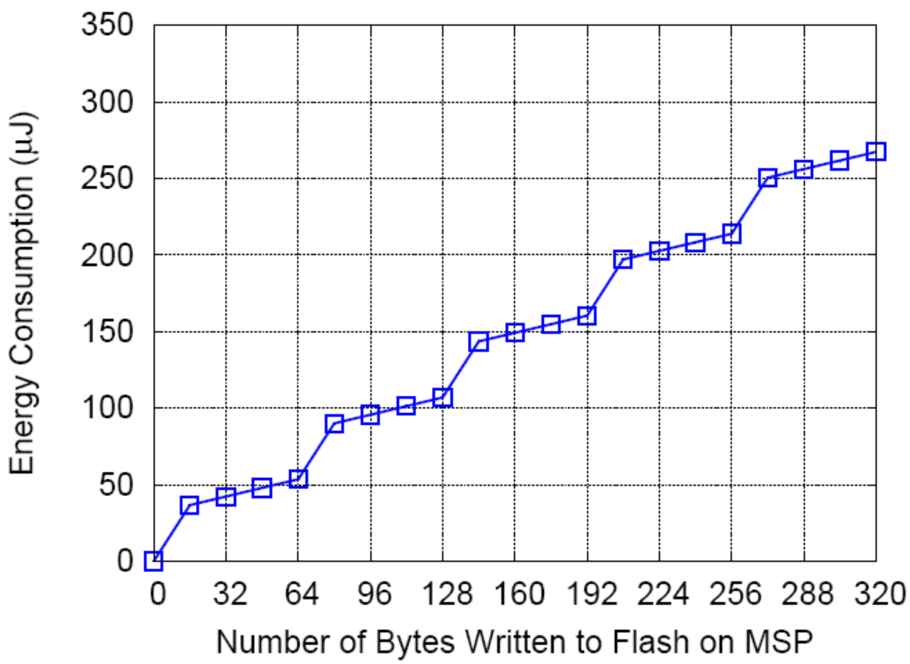
Task Set Generation

- Representative routines in sensor network and digital signal processing are selected:
  - e.g: Digital Filter, Fast Fourier Transform (FFT), Statistic, CRC, Checksum, Encryption/Decryption

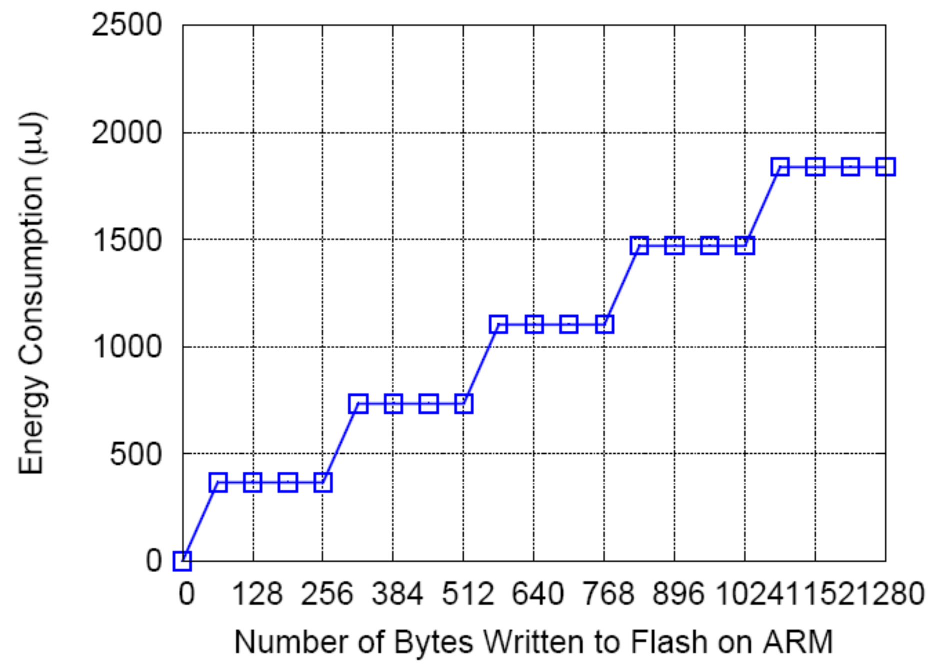
- Several task template that represent typical data processing and aggregation:



Flash Access

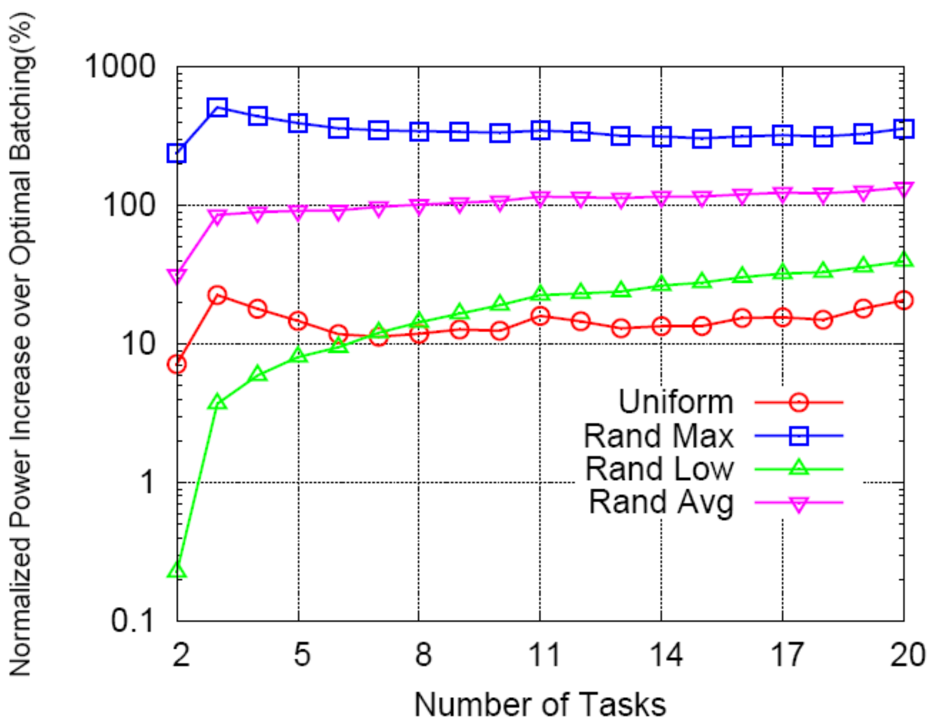


Flash Access Overhead for MSP

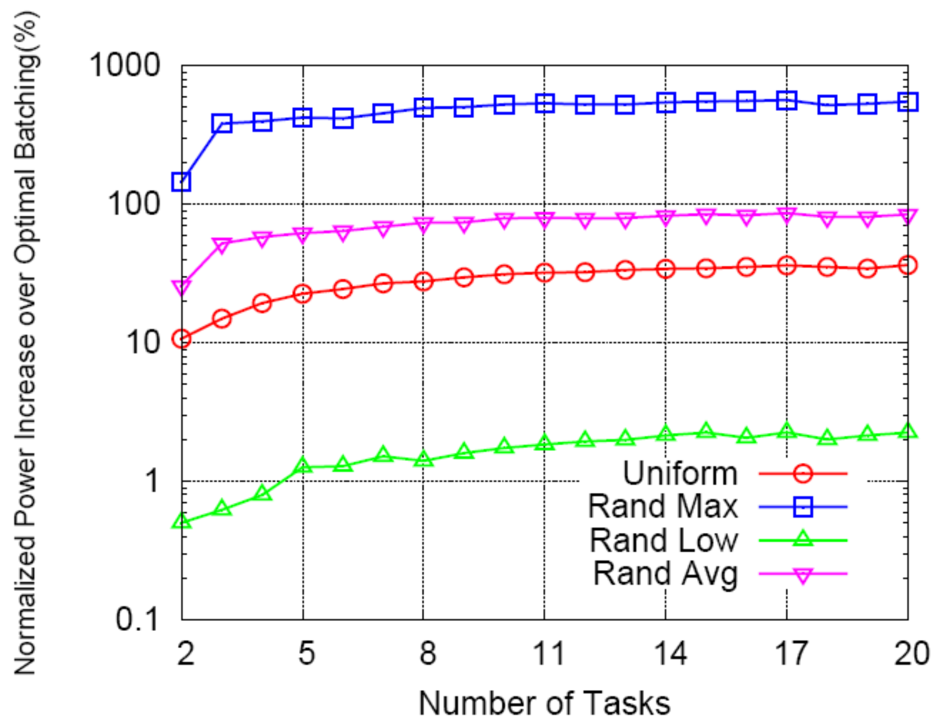


Flash Access Overhead for ARM

Experiment with Batching Period-1

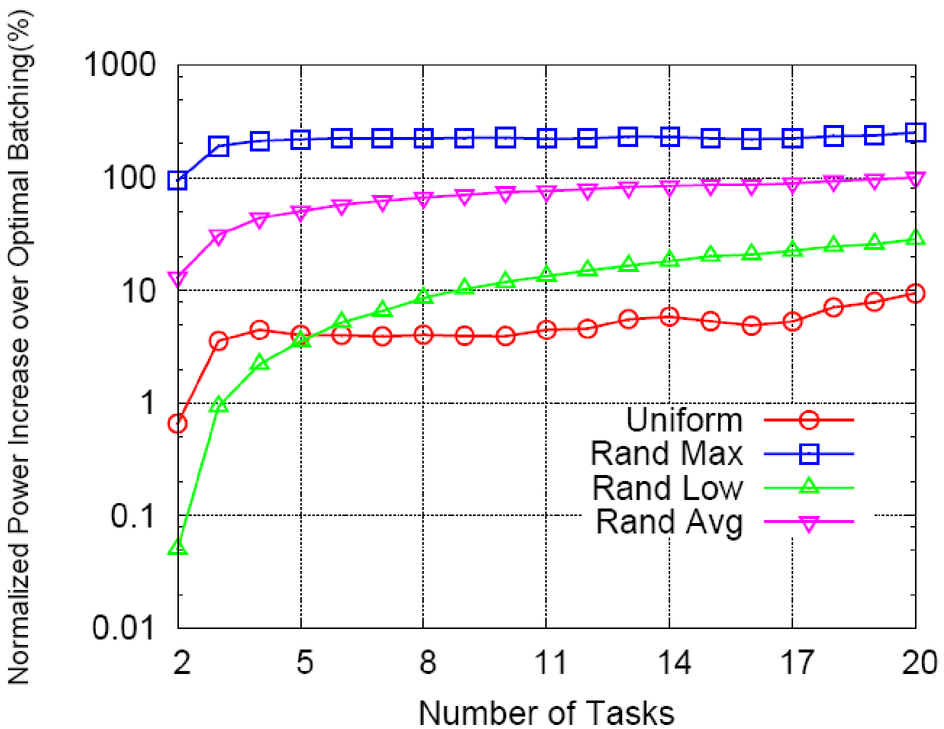


Comparison for Chain Topology on MSP

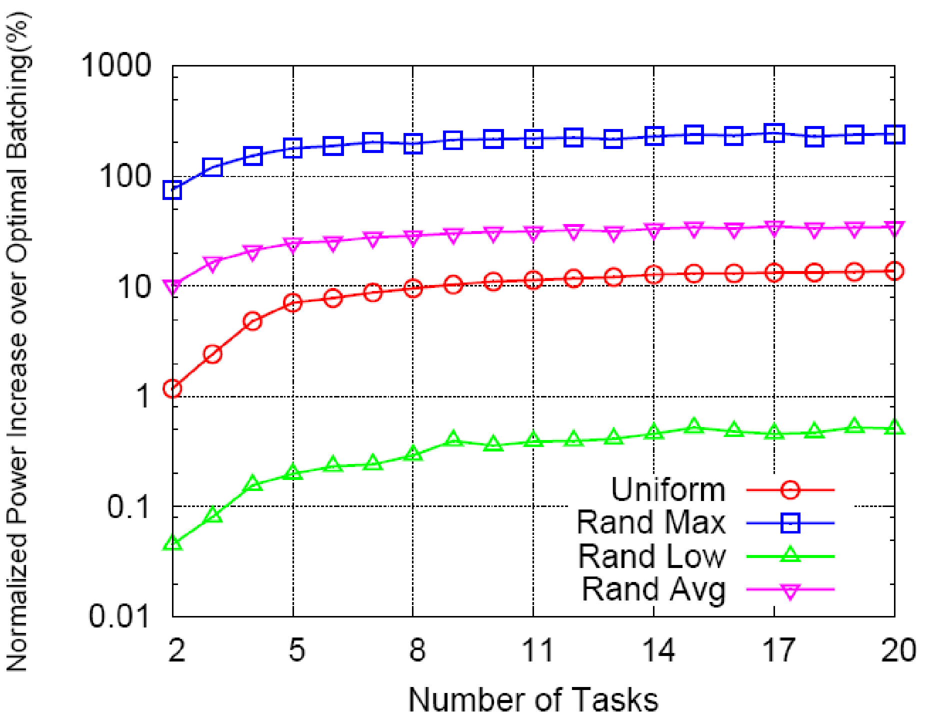


Comparison for Star Topology on MSP

Experiment with Batching Period-2

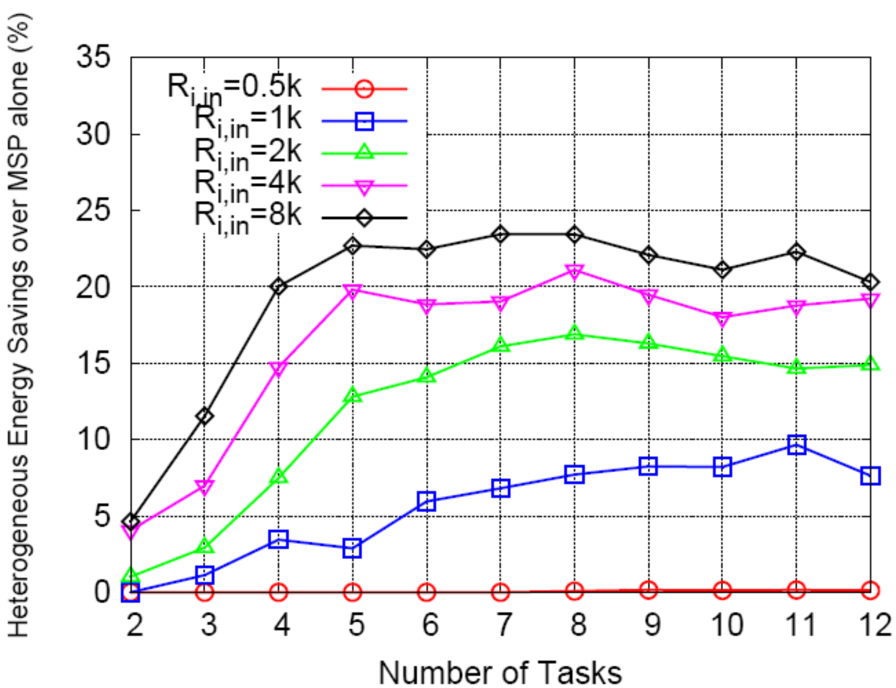


Comparison for Chain Topology on ARM

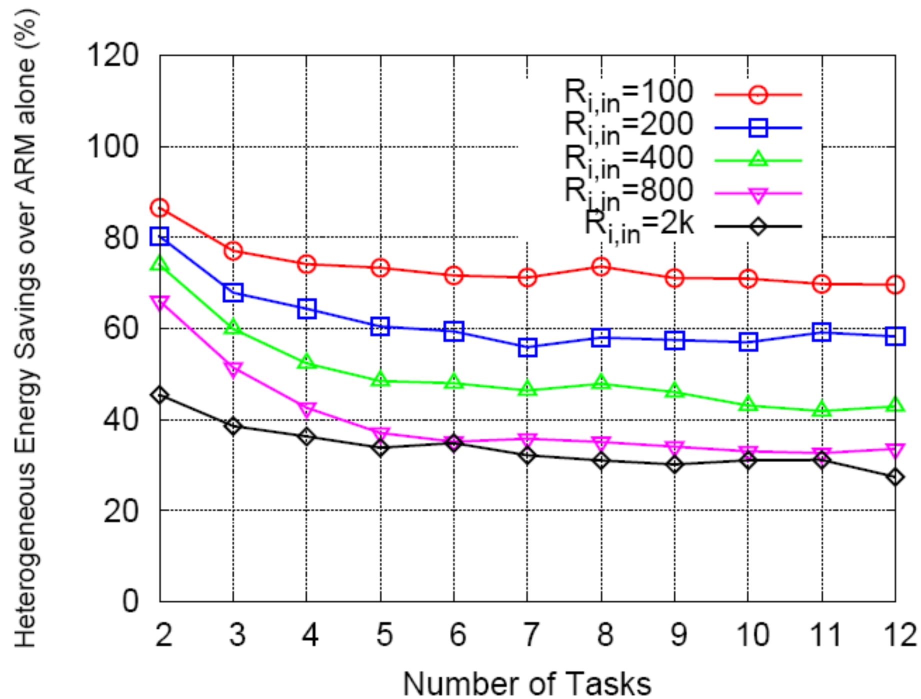


Comparison for Star Topology on ARM

Experiment with Optimal Task Assignment



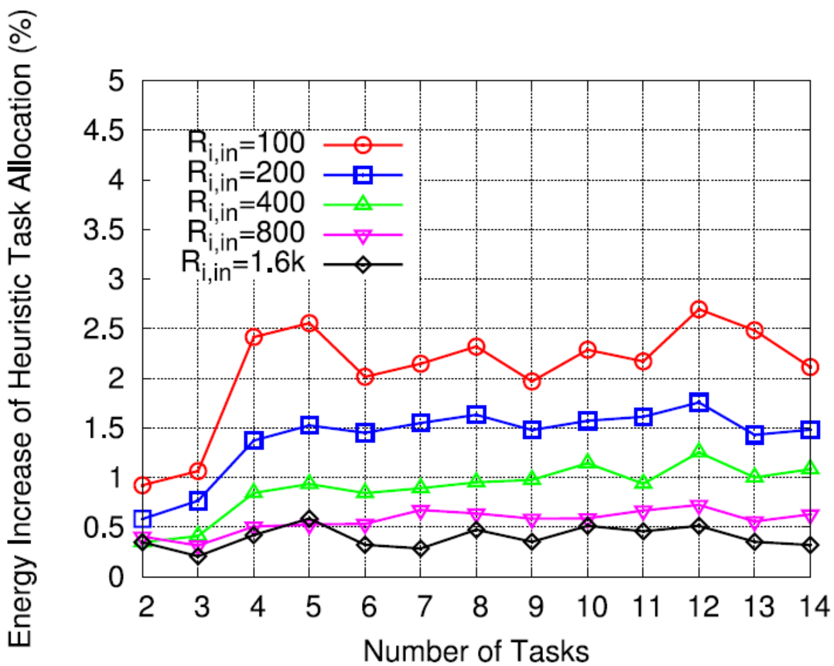
Heterogeneous Assignment vs MSP



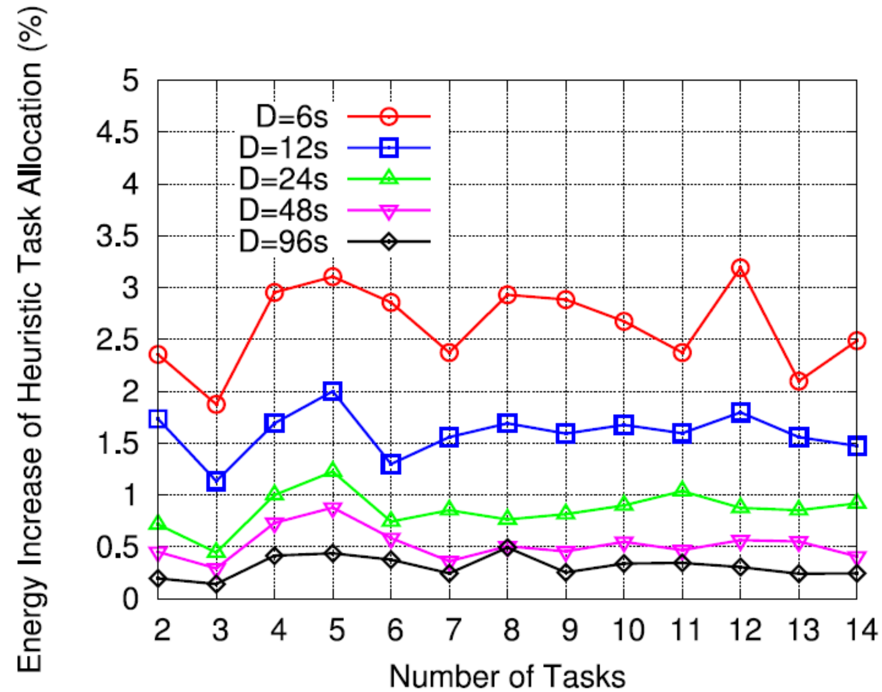
Heterogeneous Assignment vs ARM

Up to 25% energy is saved over MSP and upto 80% energy is saved over ARM

Task-Processor Heuristic Performance



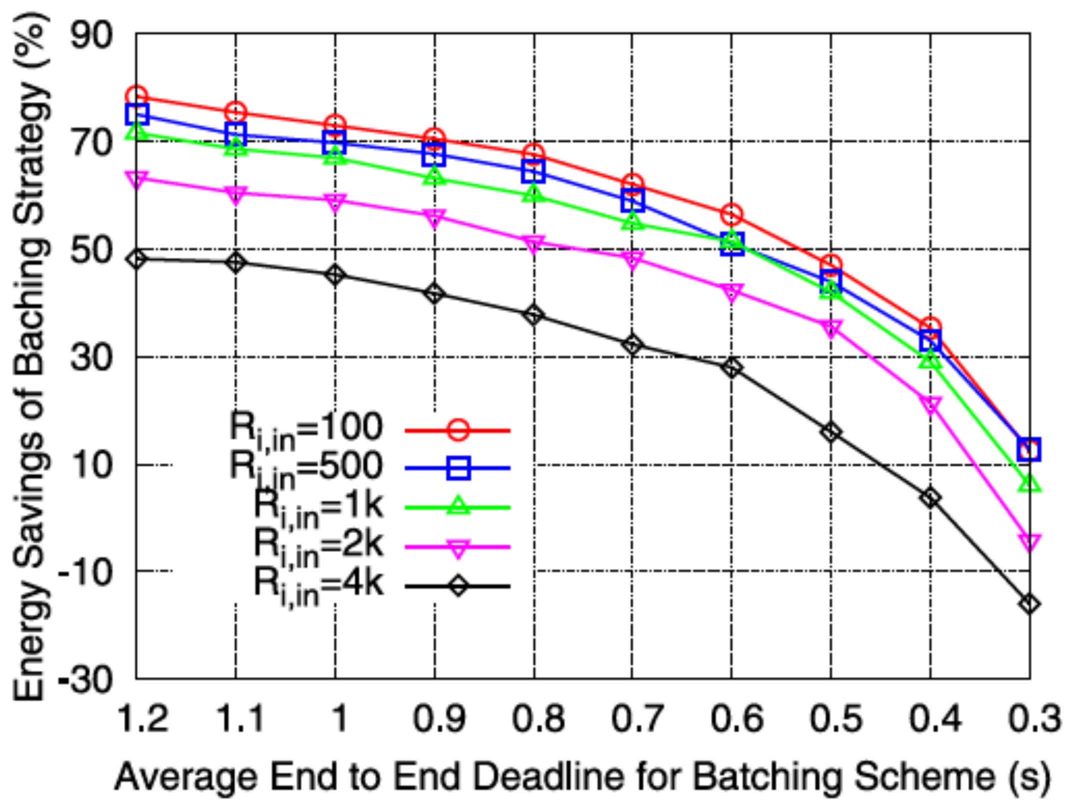
Energy Increase of Heuristic with varying Input Rate



Energy Increase of Heuristic with varying Deadline

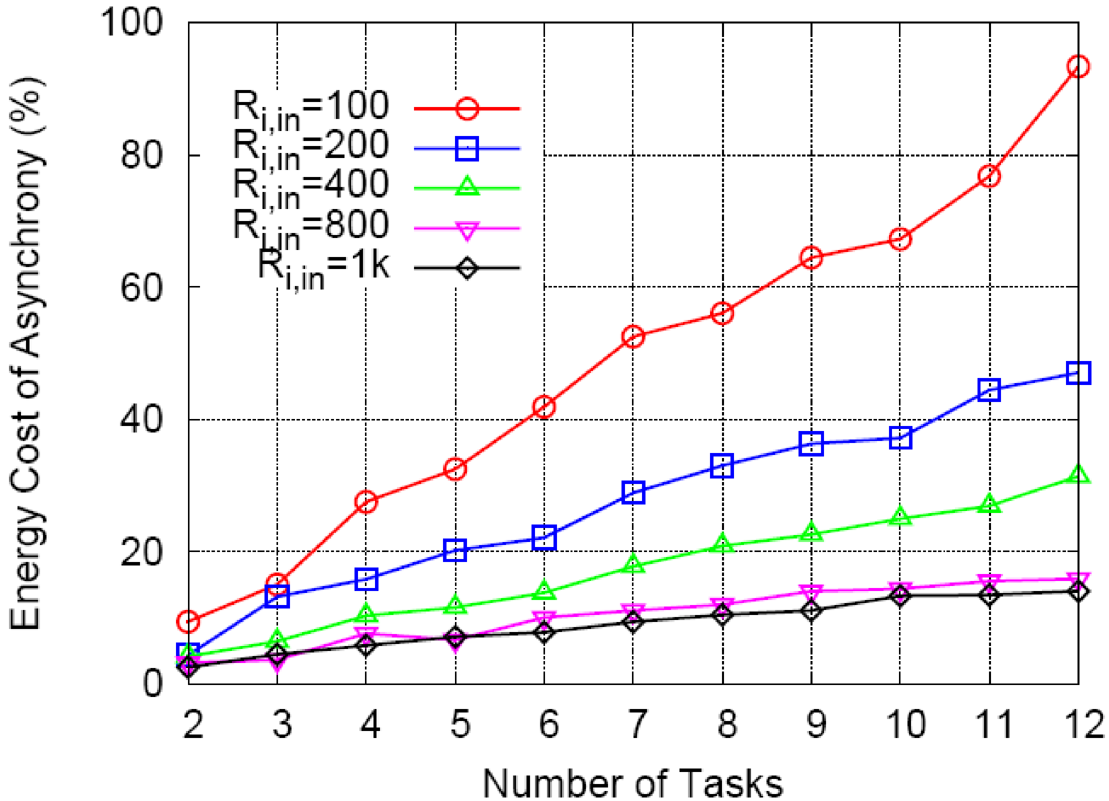
The energy increase of using heuristics is very small ☺

Tradeoffs between Energy Savings and Responsiveness



Energy savings of using batching periods decrease as end-end deadline decreases (sleep time of processors decrease)

Energy Cost of Asynchrony

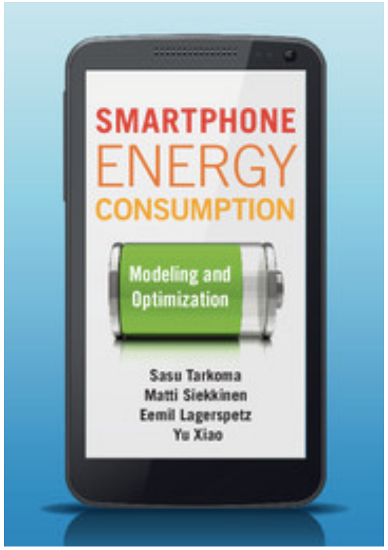


Conclusion

- Minimize energy of asynchronous multi-stage data processing with time constraints
- Optimal batching period allocation for data aggregation in sensor network
- Task to processor assignment is coupled with period allocation
- Evaluation on heterogeneous sensor platform-mPlatform

Papers

- Paper 2: ”ACE: exploiting correlation for energy-efficient and continuous context sensing." Nath, Suman. Proceedings of the 10th international conference on Mobile systems, applications, and services. ACM, 2012. (Best Paper)

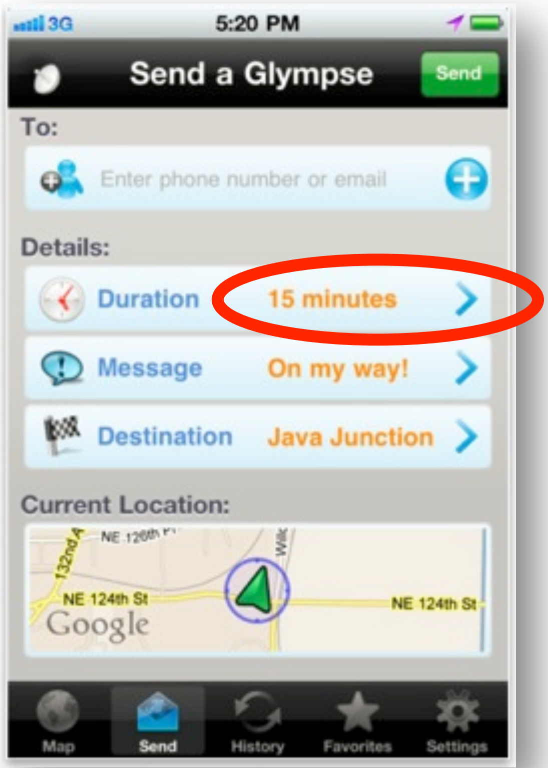


Continuous Context-Aware Apps

How much do I jog?	Mute phone in meeting	Alert when at grocery shop	Monitor indoor location	Custom message on driving
Jog Tracker	*Phone Buddy*	*Geo-Reminder*	*Batphone*	*Phone Buddy*

Continuous sensing of user's context

Sensing Context is Expensive



- Three orders of magnitude difference
  - Some apps limit how long to sense
- Our goal: push the limit

Sensing Context is Expensive

Context	Sensors	Sensing Energy (mJ)
IsWalking, IsDriving, IsJogging, IsSitting		
AtHome, AtOffice		
IsIndoor		
IsAlone		
InMeeting, IsWorking		

Q: What kind of sensors can be used to detect the above context?

Sensing Context is Expensive

Context	Sensors	Sensing Energy (mJ)
IsWalking, IsDriving, IsJogging, IsSitting	Accelerometer (10 sec)	259
AtHome, AtOffice	WiFi	605
IsIndoor	GPS + WiFi	1985
IsAlone	Mic (10 sec)	2995
InMeeting, IsWorking	WiFi + Mic (10 sec)	3505

- Three orders of magnitude difference

Q: How would you make the context-sensing more **energy-efficient**?

Potential Energy Savings by Inferring the Expensive Attributes from Cheaper ones

Sensing Context is Expensive

Context	Sensors	Sensing Energy (mJ)
IsWalking, IsDriving, IsJogging, IsSitting	Accelerometer (10 sec)	259
AtHome, AtOffice	WiFi	605
IsIndoor	GPS + WiFi	1985
IsAlone	Mic (10 sec)	2995
InMeeting, IsWorking	WiFi + Mic (10 sec)	3505

- Three orders of magnitude difference

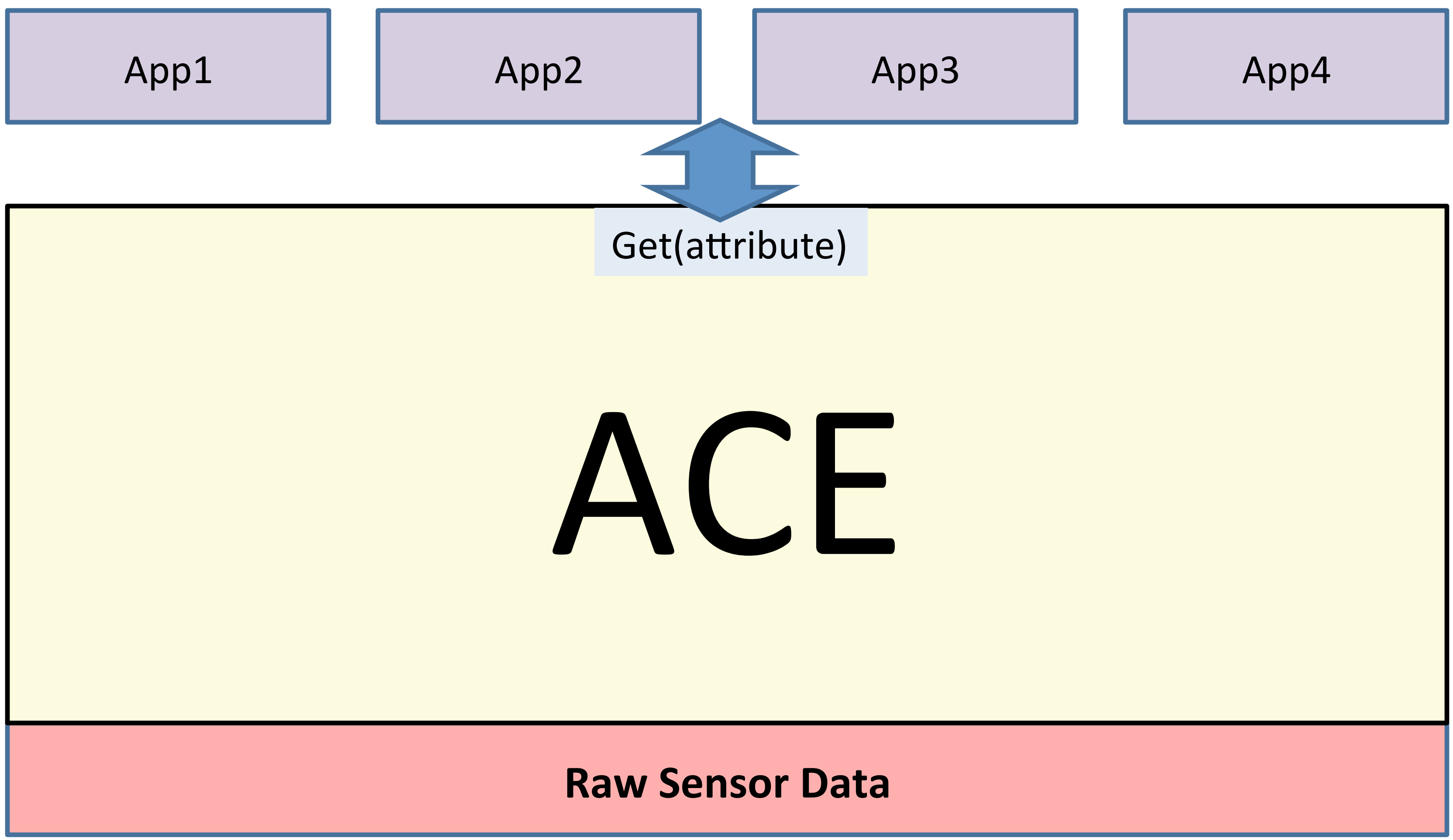
Q: How would you use cheap attributes to infer more expensive ones?

ACE: Acquisitional Context Engine

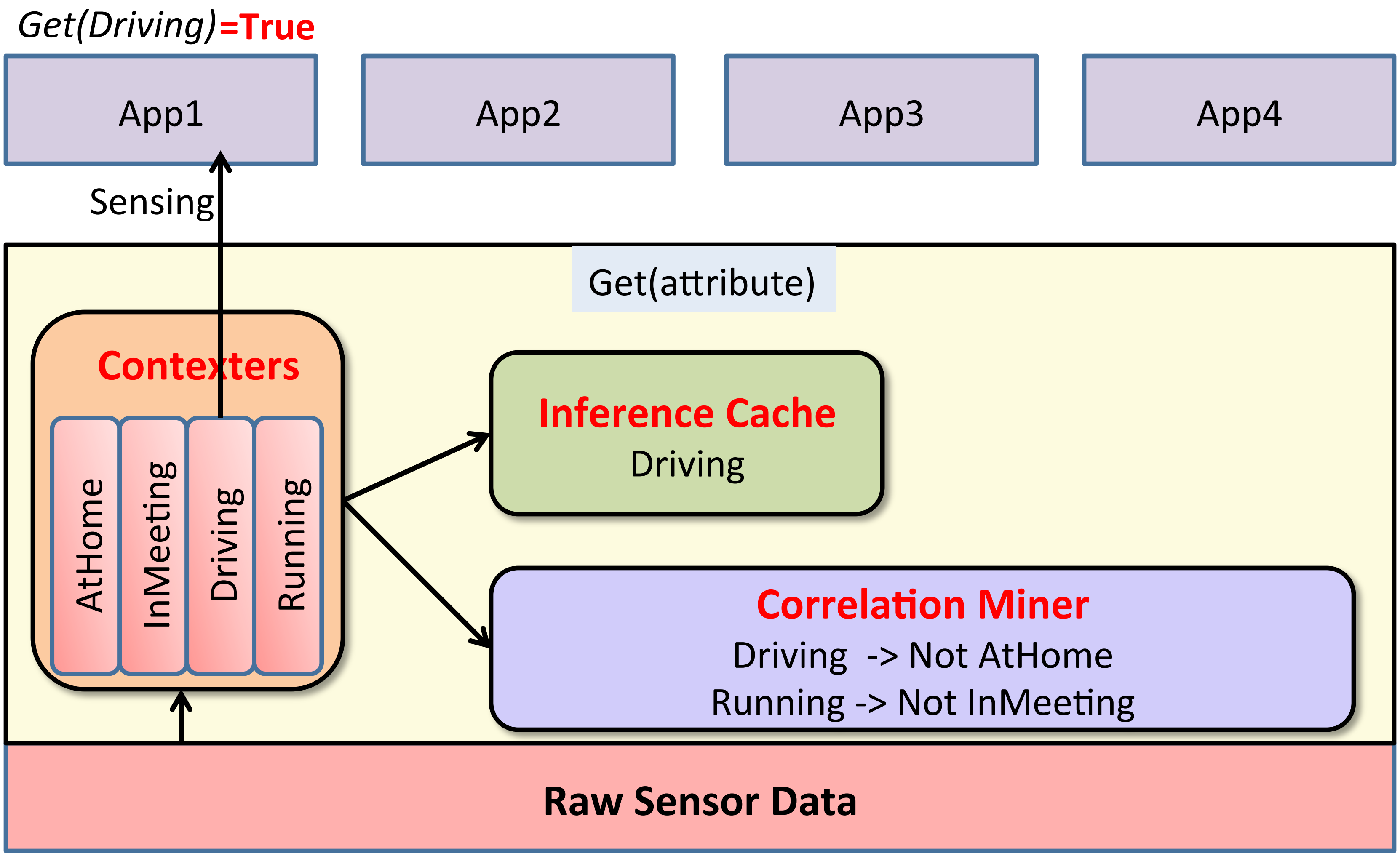
Low-energy continuous sensing middleware

- **Goal:** Reduce the Cost (Energy) of Context Sensing
- **Approach:** Opportunistically infer expensive attributes from cheap attributes
- **Conjecture:** Relationship of expensive and cheap attributes can be learnt automatically
- **Intuition:** Human activities constrained by physical constraints
  - **Behavior invariants**: Driving implies Not At Home

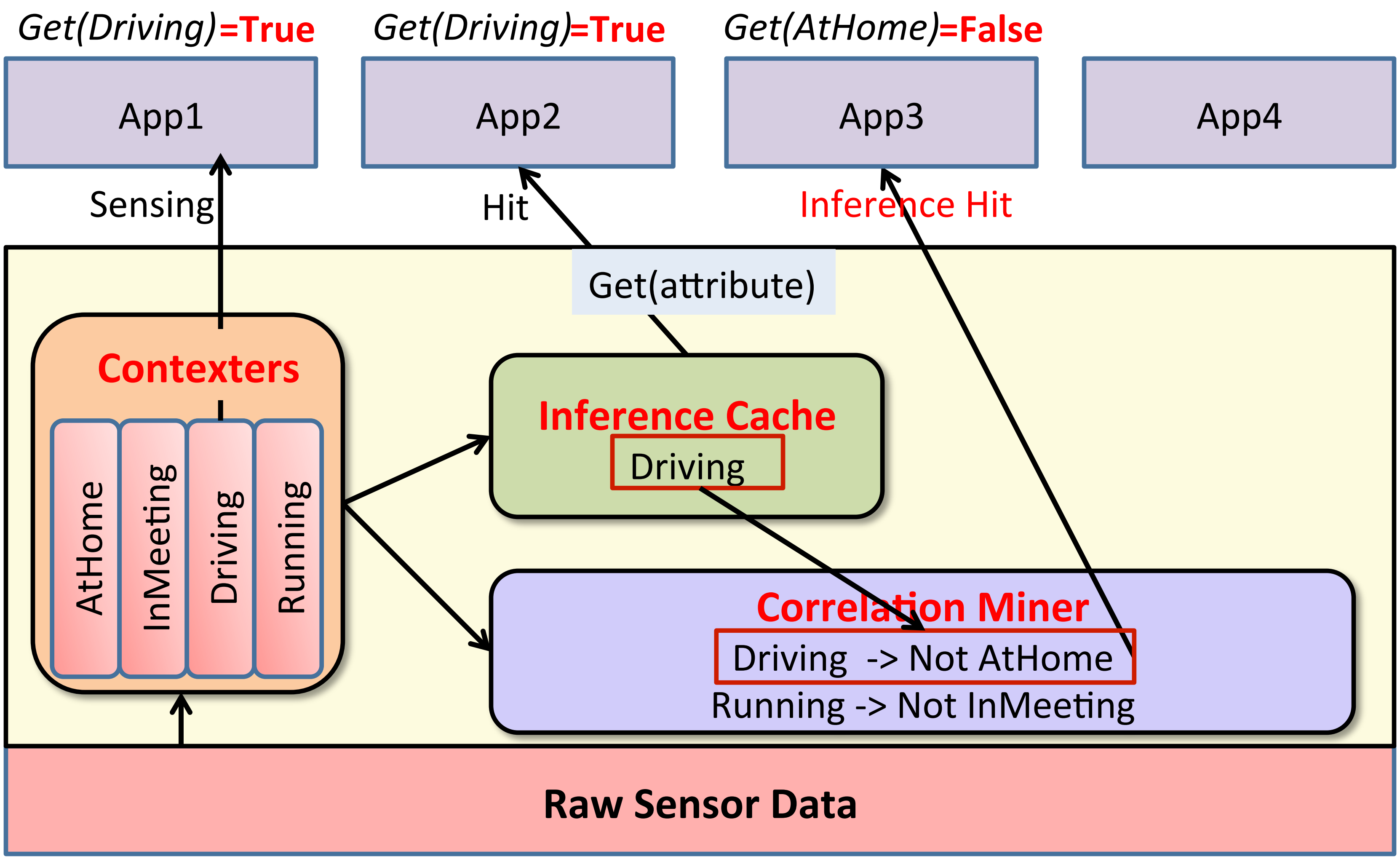
ACE Big Picture



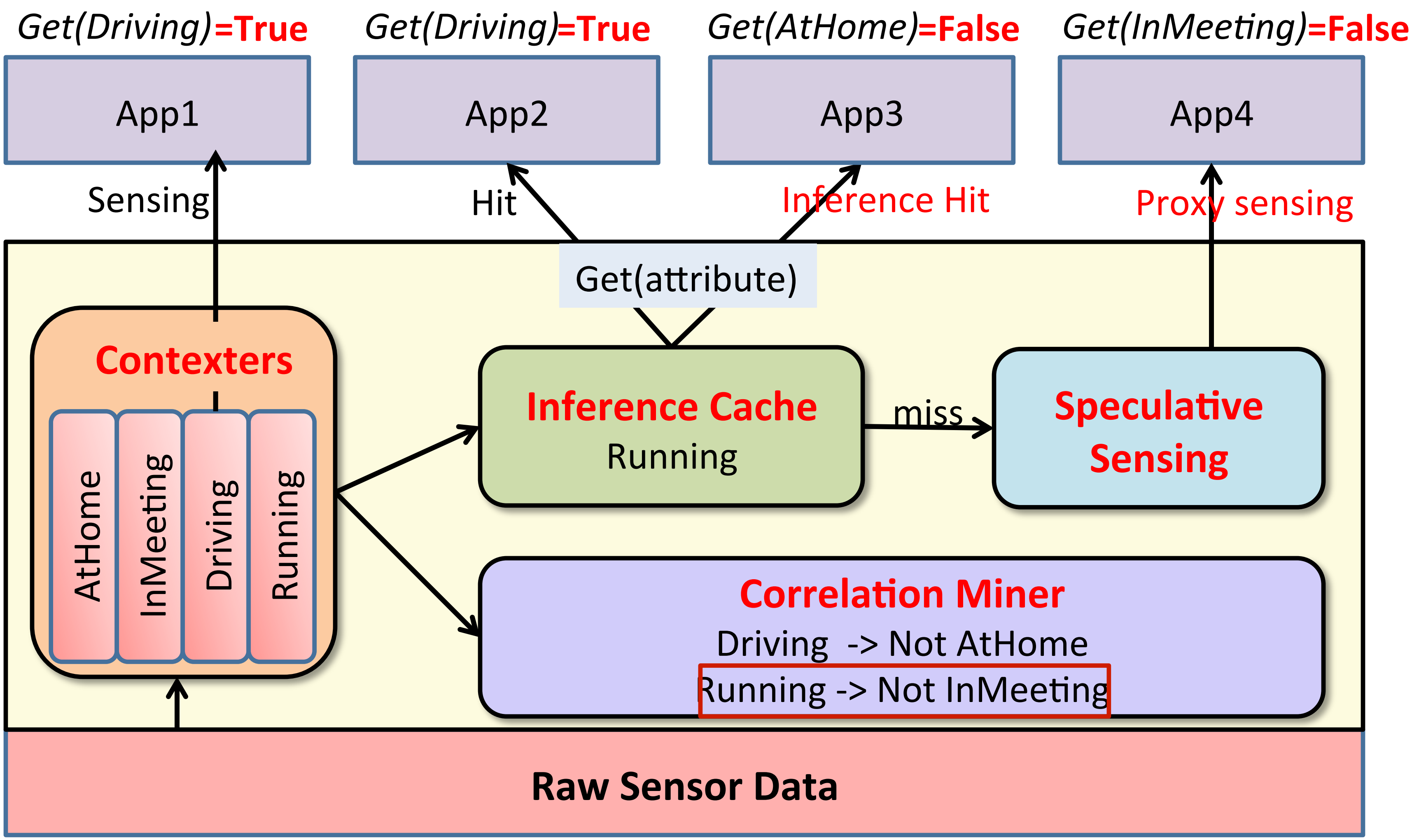
ACE Big Picture



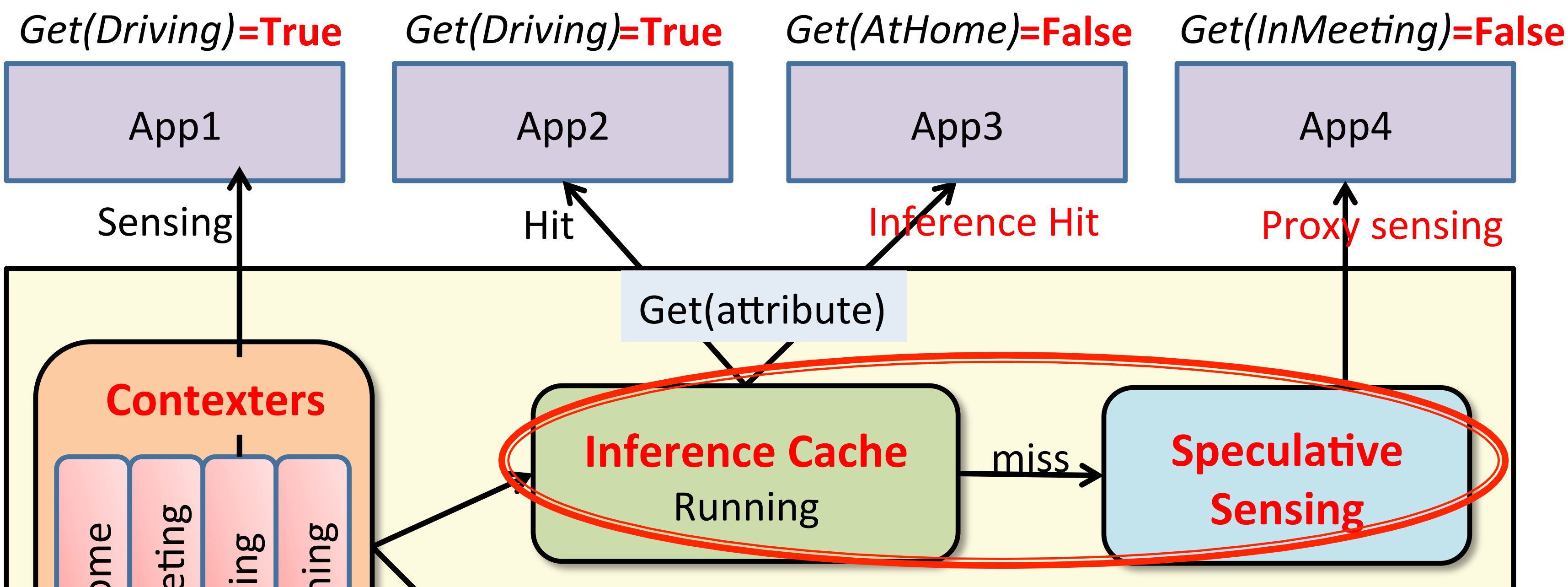
ACE Big Picture



ACE Big Picture



ACE Big Picture



Automatic process

No semantic meaning needed

Easy to extend with new Contexters

Disclaimers

- Not for apps requiring 100% accurate contexts
  - Experiments show ~4% inaccuracy
- Current prototype
  - Boolean attributes (categorical attributes)
  - Uses correlations at the same time
    - *E.g., Driving -> Not at home*
    - Ignores temporal aspects of rules

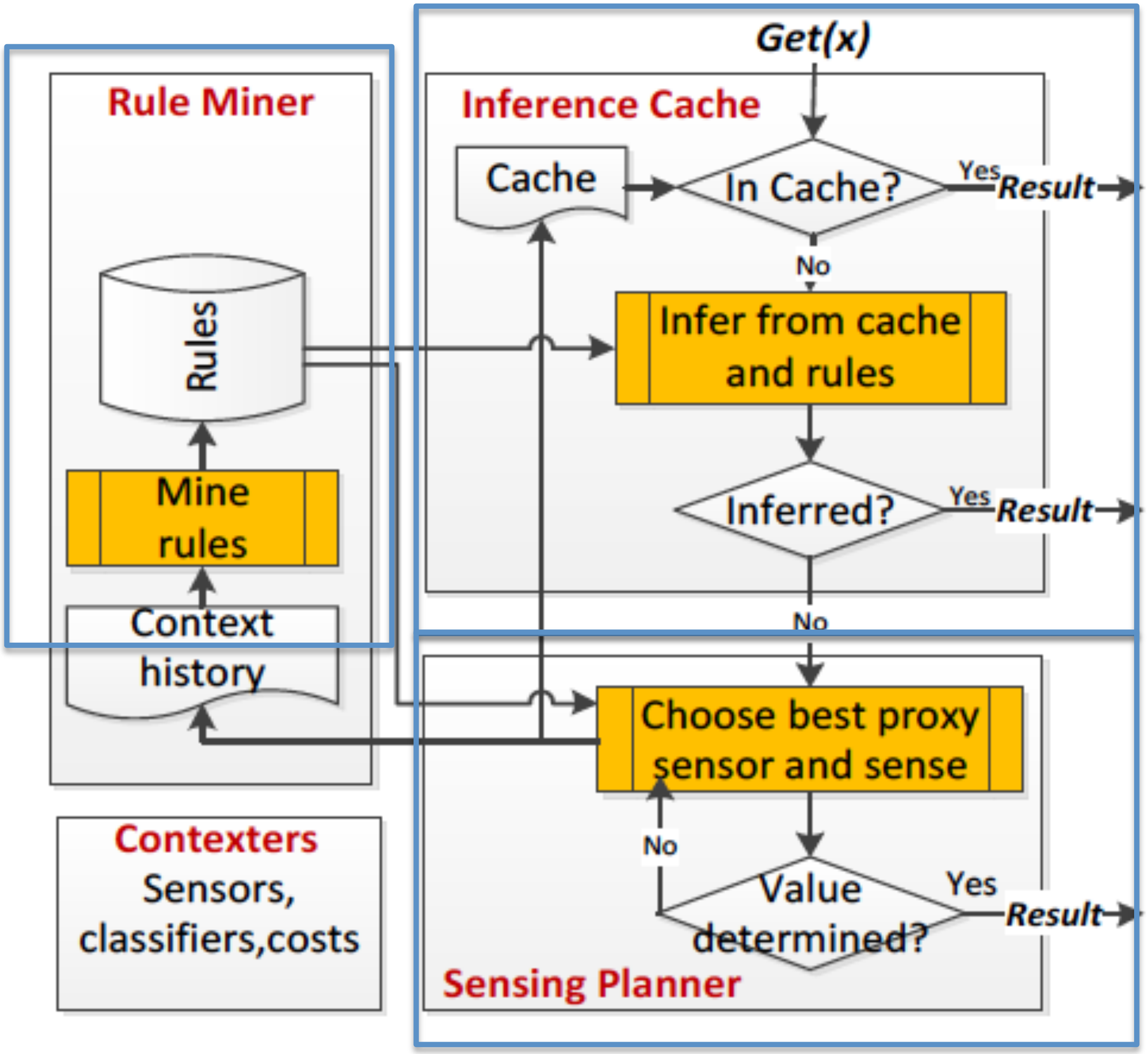
Contexters

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

Interface:

1. Attribute Name
2. Energy Cost

ACE Workflow



Key Questions

- **Feasibility:** Do useful correlations exist and can they be efficiently learnt?
- **System design:** How to systematically exploit the correlations?
- **Effectiveness:** How much energy savings?

Feasibility: Datasets

MIT Reality Mining Dataset	MSR Dataset
95 students and staffs at MIT Nokia 6600 phones, 2004-2005 min/avg/max: 14/122/269 days	10 interns and researchers Android phones min/avg/max: 5/14/30 days

```
                 ….
10:23:34 am   AtHome
10:23:35 am   Walking,Outdoor
10:23:36 am   Driving,Outdoor
10:23:55 am   Walking
10:23:59 am   InOffice
                 ….
```

Context Attributes

Location: AtHome, InOffice, IsIndoor,

Task: InMeeting, IsWorking, IsUsingApp, IsCalling,

Transportation mode: IsWalking, IsBiking, IsDriving, IsSitting,

Group: IsAlone

Apriori Algorithm to Find the Rules

- Example: 1000 transactions, 200 include both A and B, and 80 of the 200 also includes C.
- Association rule: (A,B) => C
  - Support: 80/1000=8%
  - Confidence: 80/200=40%
- Parameters
  - minSup (4% works well for ACE)
  - minConf (99% works well for ACE)

Mining Behavior Invariants

```
        ….
10:23:34 am   AtHome
10:23:35 am   Walking
10:23:36 am   Driving
10:23:50 am   Driving
10:23:55 am   Walking
10:23:59 am   InOffice
        ….
```

```
…
Driving -> Not AtHome

{Indoor, Alone, Not AtHome}
-> InOffice
…
```

Rules = Patterns that almost always hold

Rules may be person-specific

We use association rule mining algorithms

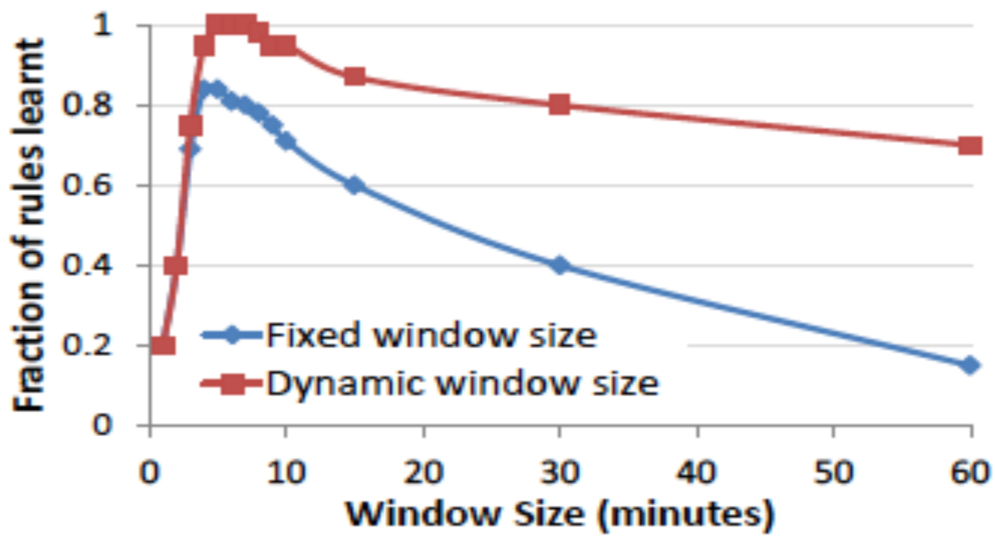
Challenges

Streaming data (Decide the right batch size)	**Capture rare rules** (Several hours on phone)	**Redundant rules** (~700 per person)	**Bootstrapping**

See the paper for details

Addressing Rule Mining Challenges

- Choose the right window size to batch context attributes to form transactions
- A user can change her context any time within a window, hence dynamic windowing is necessary



5 min window is optimal (from data)

Addressing Rule Mining Challenges

- Deal with low support
  - Offload rule mining to a powerful backend server
- Deal with inaccuracies
  - Do cross validation using ground truth results from occasional user annotations
- Suppress redundant rules
  - Use data mining algorithm to reduce the redundancy
- Bootstrapping
  - Start with universal rules and update them with personalized rules

Correlation Miner on Two Traces

- Useful correlations exist in our traces
  - Avg. ~44 non-redundant rules per person

{IsDri

{Indoo

{IsWal

Some rules can be specific to a single user and may not apply to all users

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

$\boxed{\{\mathtt{AtHome} = \mathrm{F}, \mathtt{IsDriving} = \mathrm{F}, \mathtt{IsUsingApp} = \mathrm{T}\} \Rightarrow \{\mathtt{InOffice} = \mathrm{T}\}}$

$\boxed{\{\mathtt{IsJogging} = \mathrm{T}\} \Rightarrow \{\mathtt{AtHome} = \mathrm{T}\}}$

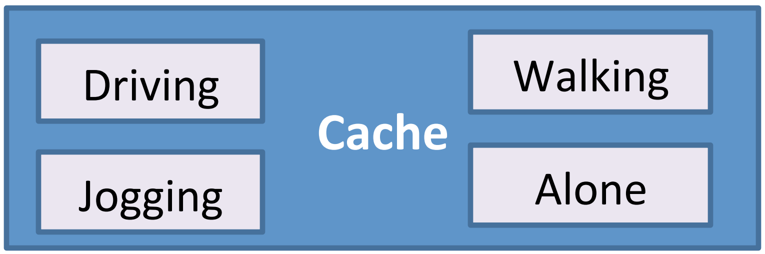
- Errors can be kept reasonably low (~ 4%)
  - Take only rules with high confidence (~ 99%)
  - Frequent cross-validation (1 in 20)

Key Questions

- **Feasibility:** Are there useful rules? Can we learn them?
- **System design:** Systematically exploiting correlation
  - Inference Cache
  - Speculative Sensing
- **Effectiveness:** How much energy savings?

Inference Caching

Get(Indoor)



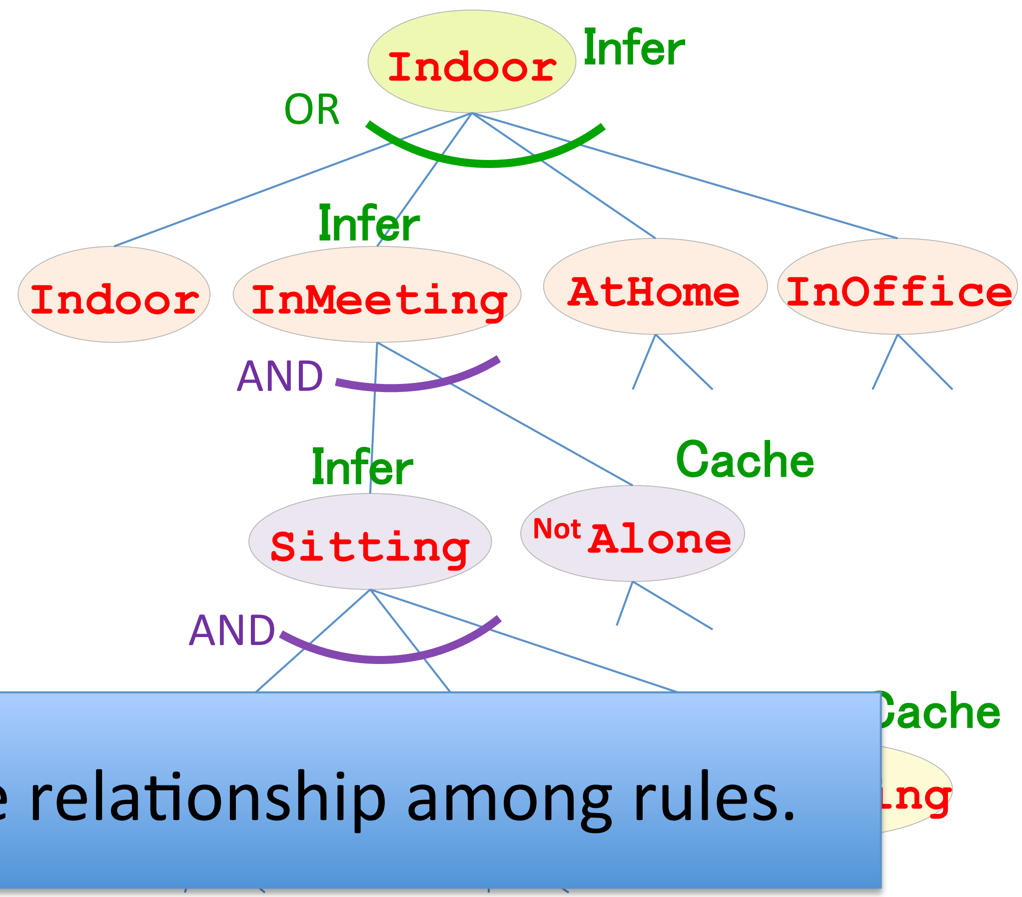
Indoor -> Indoor
InMeeting -> Indoor
InOffice -> Indoor
AtHome -> Indoor

S

N

AND Not Jogging -> Sitting

AND-OR Expression Tree

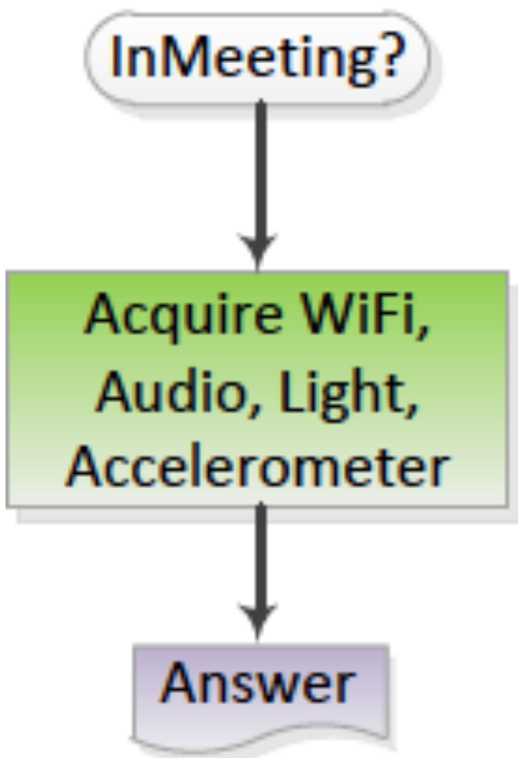


Capture the transitive relationship among rules.

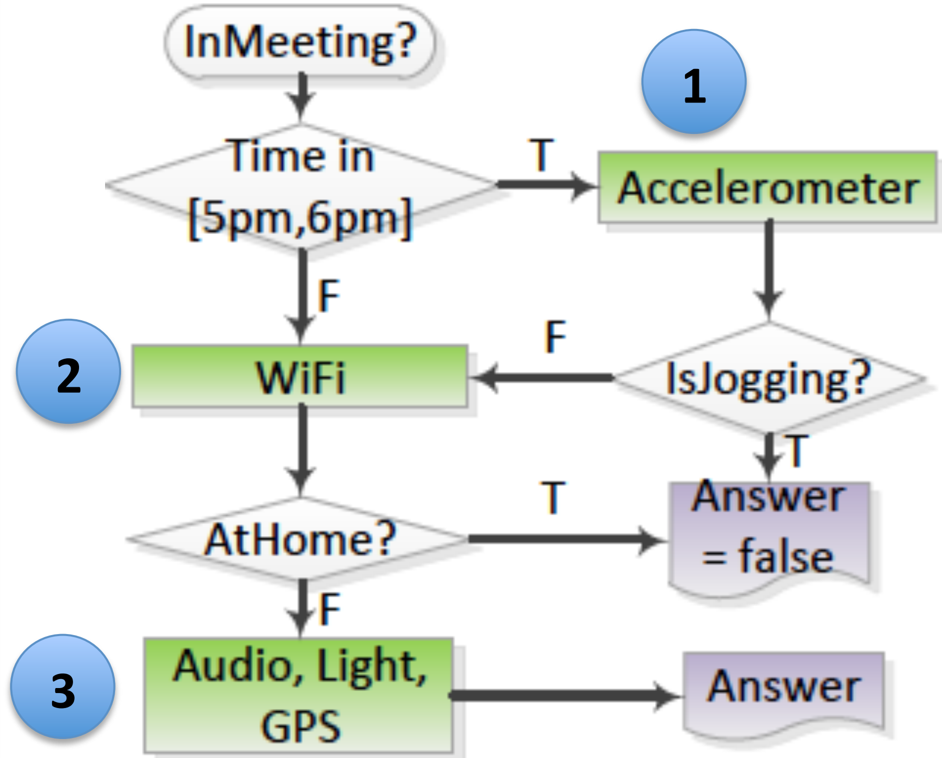
Speculative Sensing

- Goal: speculatively sense a <u>cheap attribute</u> to determine value of an <u>expensive attribute</u>
  - Infer InMeeting from IsRunning (e.g., 5pm)
- Challenge:
  - Choose the next attribute to sense Cost c
    - If infers target attributes, save energy Prob p
    - If not, waste energy
  - Goal: minimize expected cost
    - Choose attributes with low c and high p

Example: InMeeting?



Traditional plan



ACE plan

Speculative Sensing

- Problem: Select next attributes to sense that minimizes the expected total sensing cost

- NP Hard in general
  - Probabilistic And-Or Tree Resolution (PAOTOR)

- ACE provides: (*see paper for details*)
  - Dynamic programming : usable for <10 attributes
  - Heuristic: Fast, close to optimal

Key Questions

- **Feasibility:** Are there useful rules? Can we learn them?
- **System design:** Systematically exploiting correlation
  - Inference Cache
  - Speculative Sensing
- **Effectiveness:** How much energy savings?

Evaluation Setup

Prototype on Windows Phone

1G CPU
512MB RAM
Li-Ion 1500 mAh Battery

How much do I jog?

Mute phone in meeting

Alert when at grocery shop

Three apps

Jog Tracker | **Phone Buddy** | **Geo-Reminder**

IsWalking, isJogging, and IsBiking

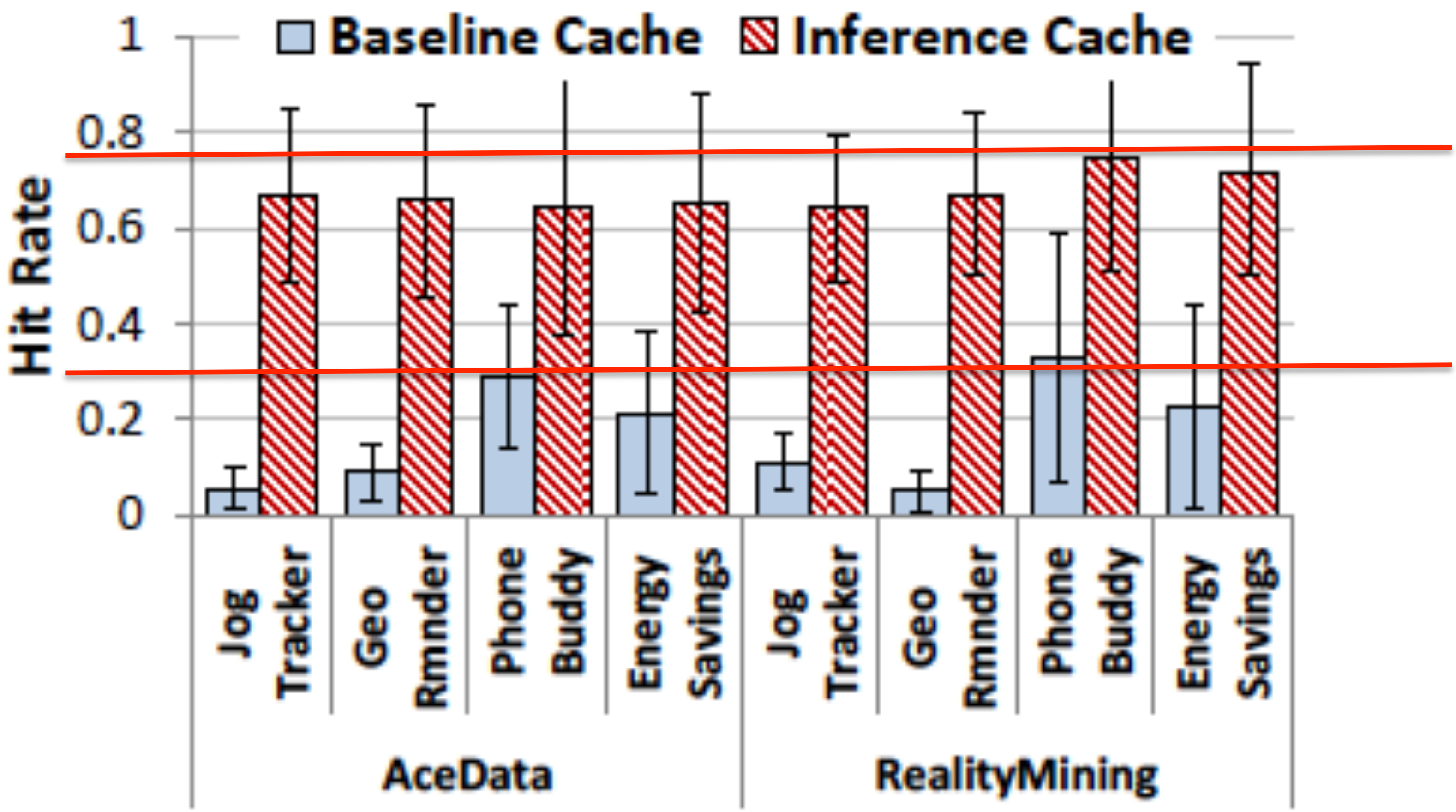
IsDriving,InMeeting, IsAlone, and InOffice.

AtHome, InOffice, Indoor

Effectiveness with MSR and Reality Mining traces

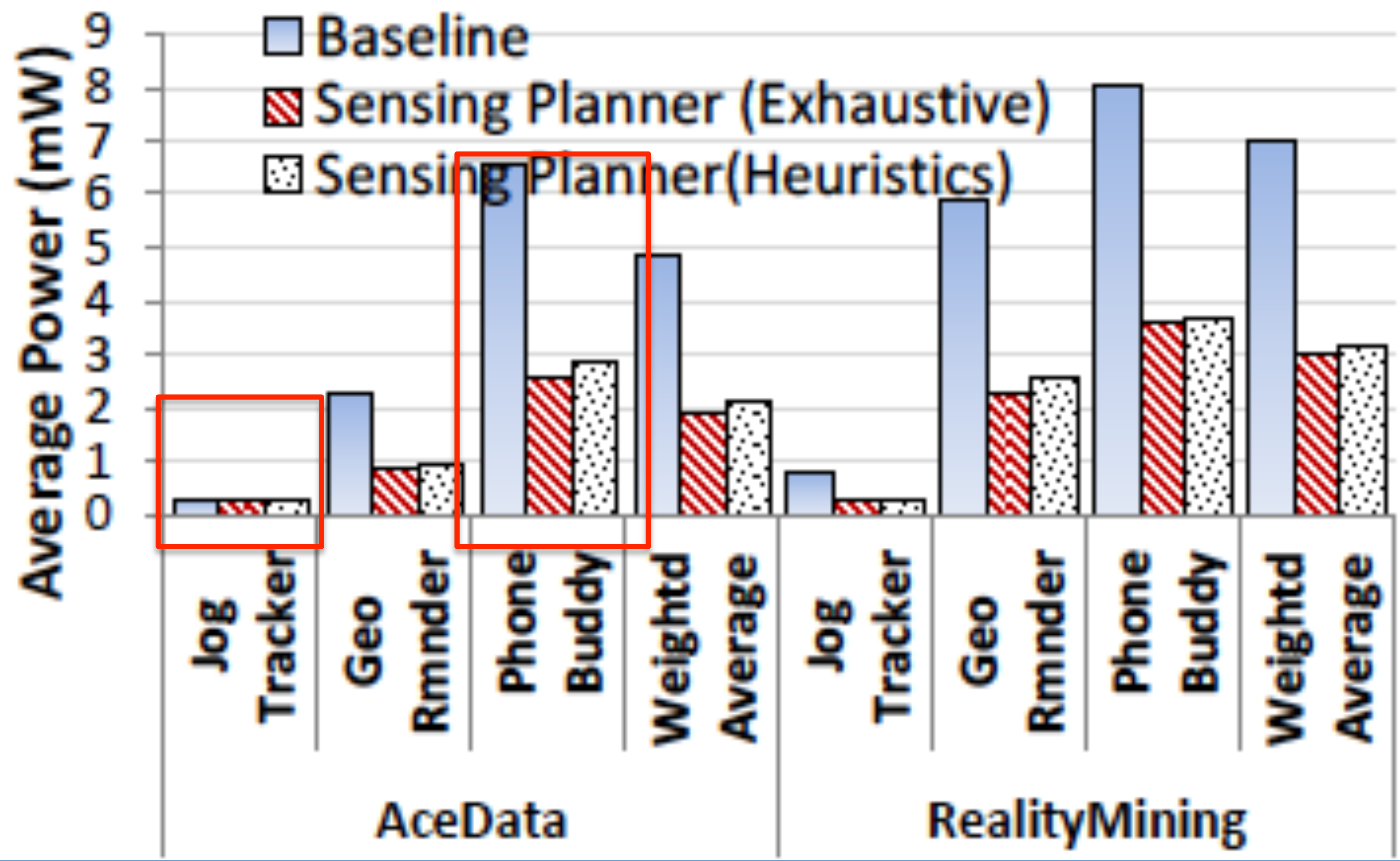
Performance on Samsung Focus Win 7 phone

Hit Rate of Inference Cache



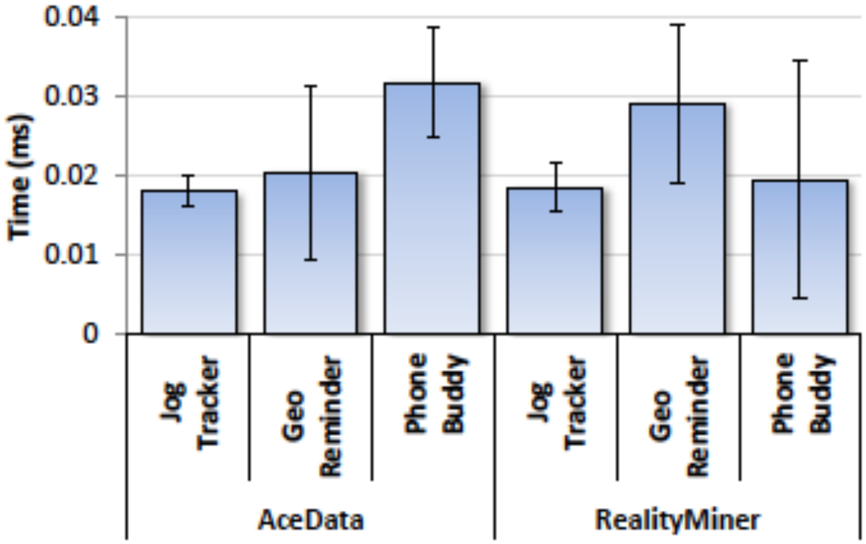
Inference Cache has a much higher hit rate: return an attribute even if it is not in cache!

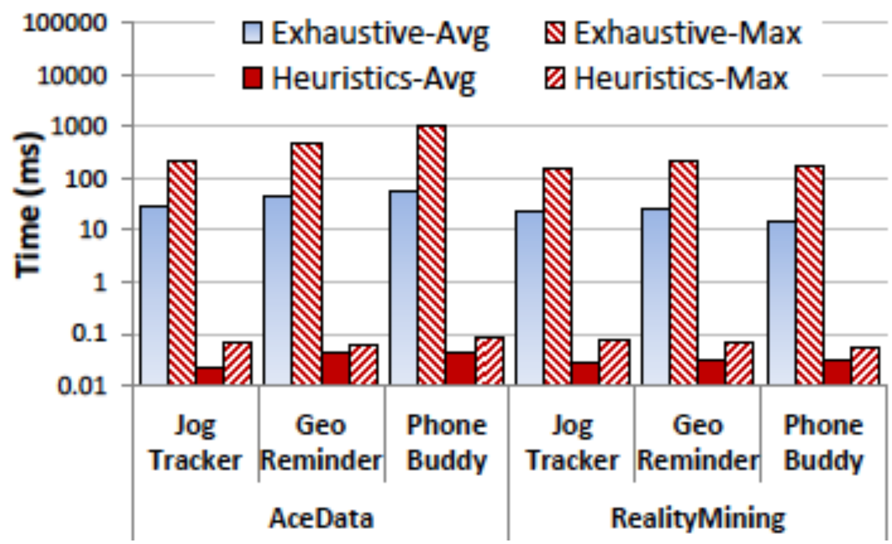
Energy Savings by Sensing Planner



Sensing Planner saves 5%-60% power compared to baseline (Heuristics are as good as Exhaustive Algorithm)

Overhead of ACE



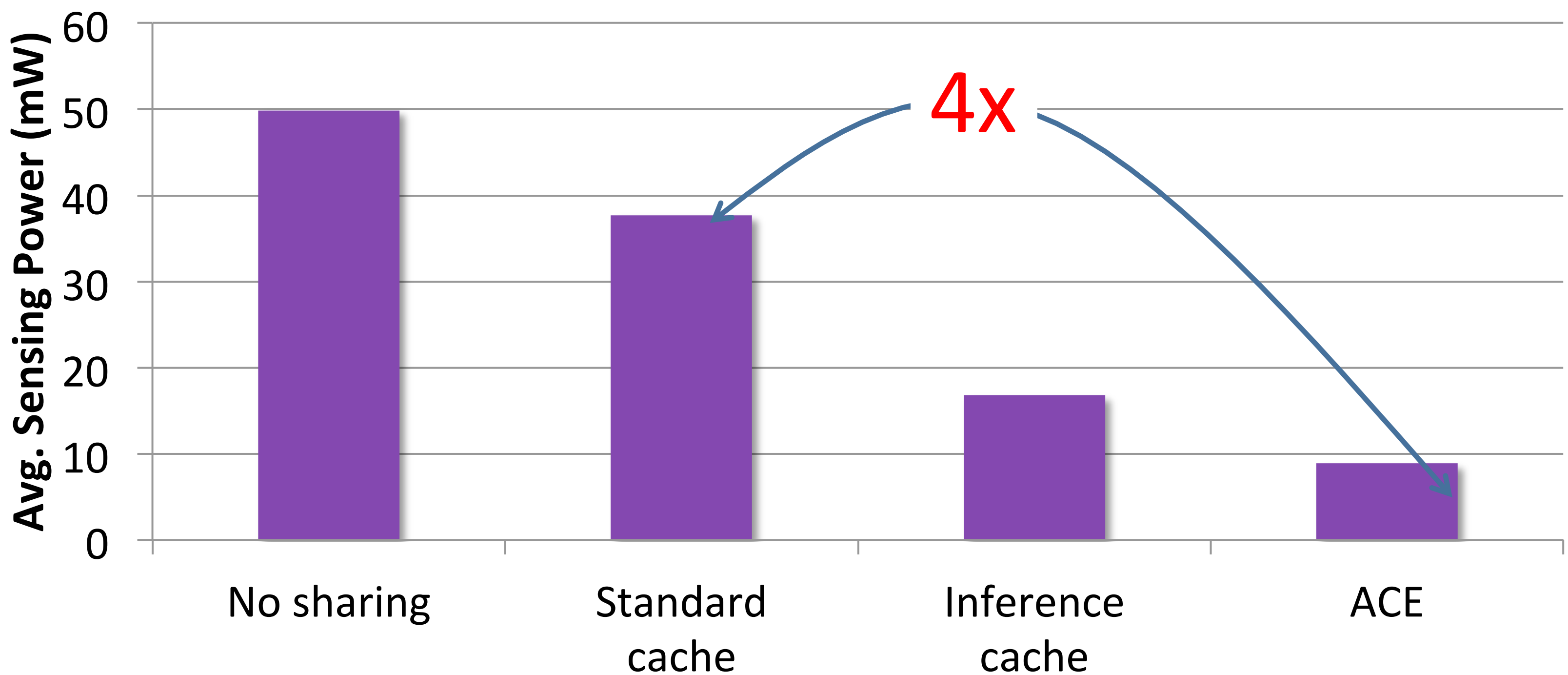


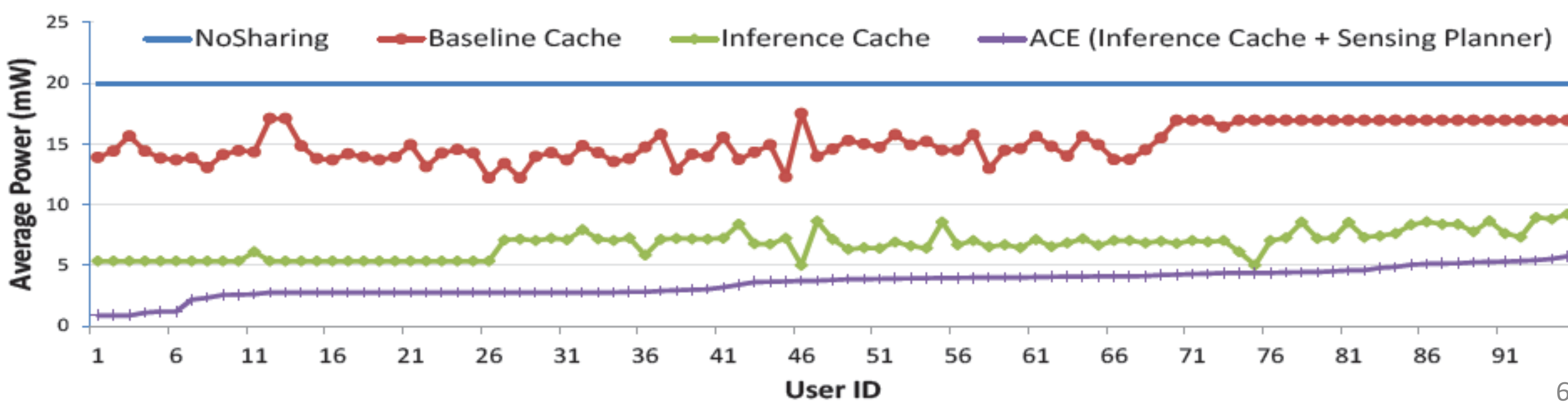
Time on Inference Cache (Cache Hit)

Time on Speculative Sensing (Cache Miss)

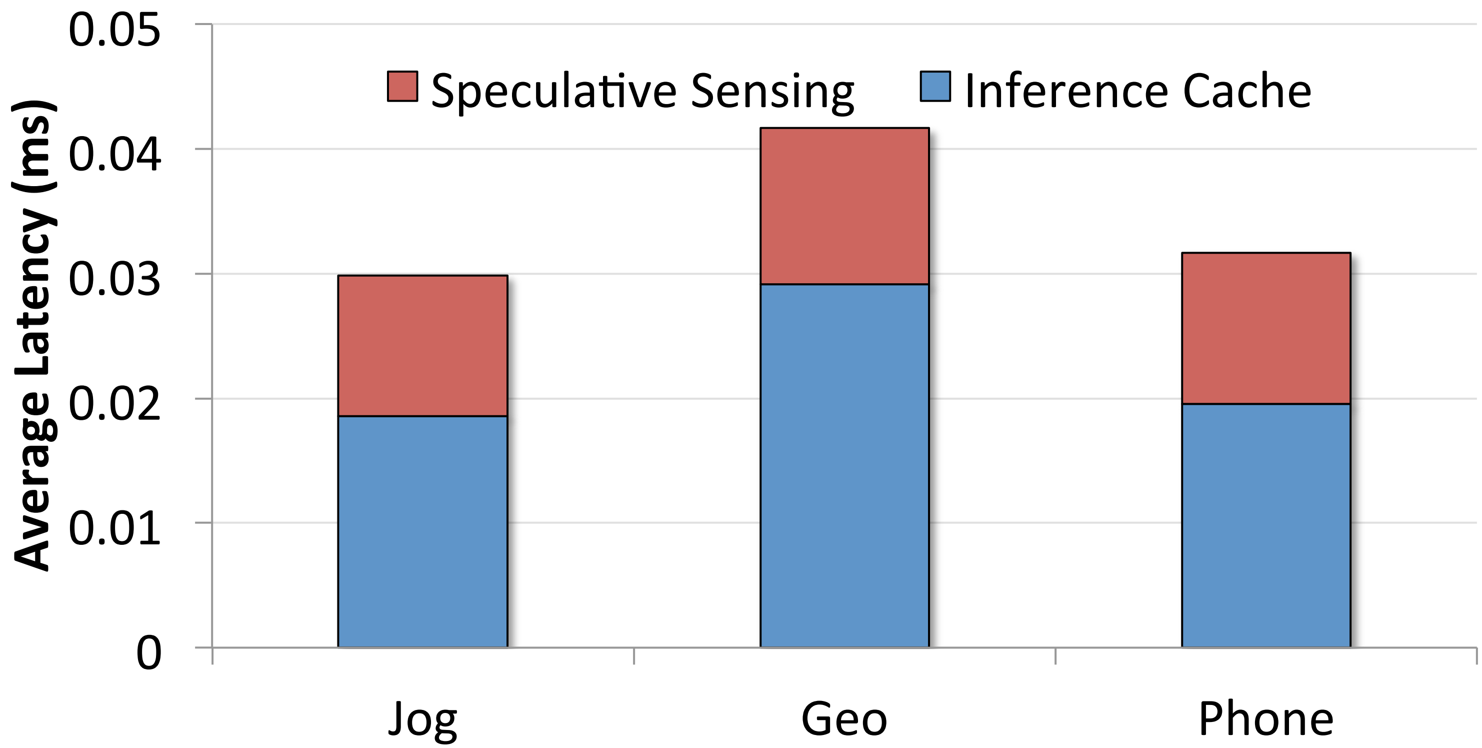
Time < 0.1 ms; Memory < 15 MB. Affordable on phones!

End-to-End Energy Savings





End-to-end Latency



End-to-end latency is < 0.1 ms, which is acceptable to all 3 apps!

What are the limitations you see?

Limitations claimed by author

- Fundamental: Occasionally inaccuracy in context inference
  - Rule mining parameters (support and confidence)
  - Cache expiration time
  - Cross validation frequency
- Non-fundamental
  - Boolean attributes only (E.g., cannot capture the user's moving speed, etc.)
  - Not exploit the temporal correlations between attributes (*E.g., InOffice => Not at home for next 10 mins*)
  - Inference cache only returns the value of an attribute not the confidence

Conclusion

- Useful correlations exist across context attributes
- ACE uses two key ideas to exploit correlation
  - Inference caching
  - Speculative sensing
- Automatically avoids sensing as much as possible, without requiring semantic information
- Significant sensing energy savings (4.2x) at the cost of small inaccuracies (~4%)