Cellular Networks and Mobile Computing COMS 6998-11, Fall 2012

Instructor: Li Erran Li
(lierranli@cs.columbia.edu)

http://www.cs.columbia.edu/~lierranli/coms6998-11Fall2012/

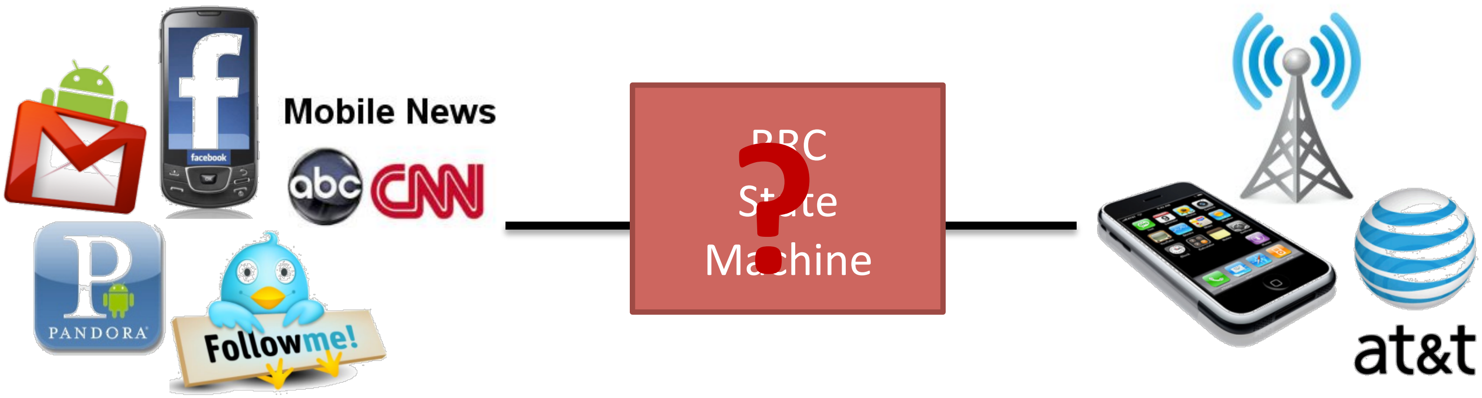
10/2/2012: Radio Resource Usage Profiling and Optimization

Outline

- Introduction
- Network Characteristics
- RRC State Inference
- Radio Resource Usage Profiling & Optimization
- Network RRC Parameters Optimization
- Conclusion

Introduction

- Typical testing and optimization in cellular data network



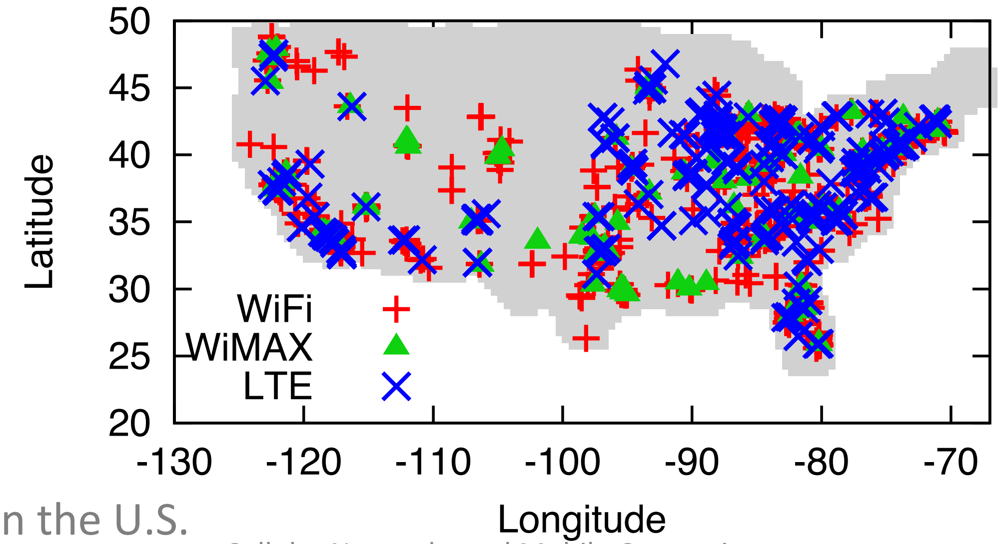
- Little focus has been put on their **cross-layer interactions**

  Many mobile applications are not cellular-friendly.
- The key coupling factor: the **RRC State Machine**
  - Application traffic patterns trigger state transitions
  - State transitions control radio resource utilization, end-user experience and device energy consumption (battery life)

Courtesy: Feng Qian et al.

Network characteristics

- ***4GTest*** on Android
  - *http://mobiperf.com/4g.html*
  - Measures network performance with the help of 46 **M-Lab** nodes across the world
  - **3,300** users and **14,000** runs in 2 months 10/15/2011 ~ 12/15/2011

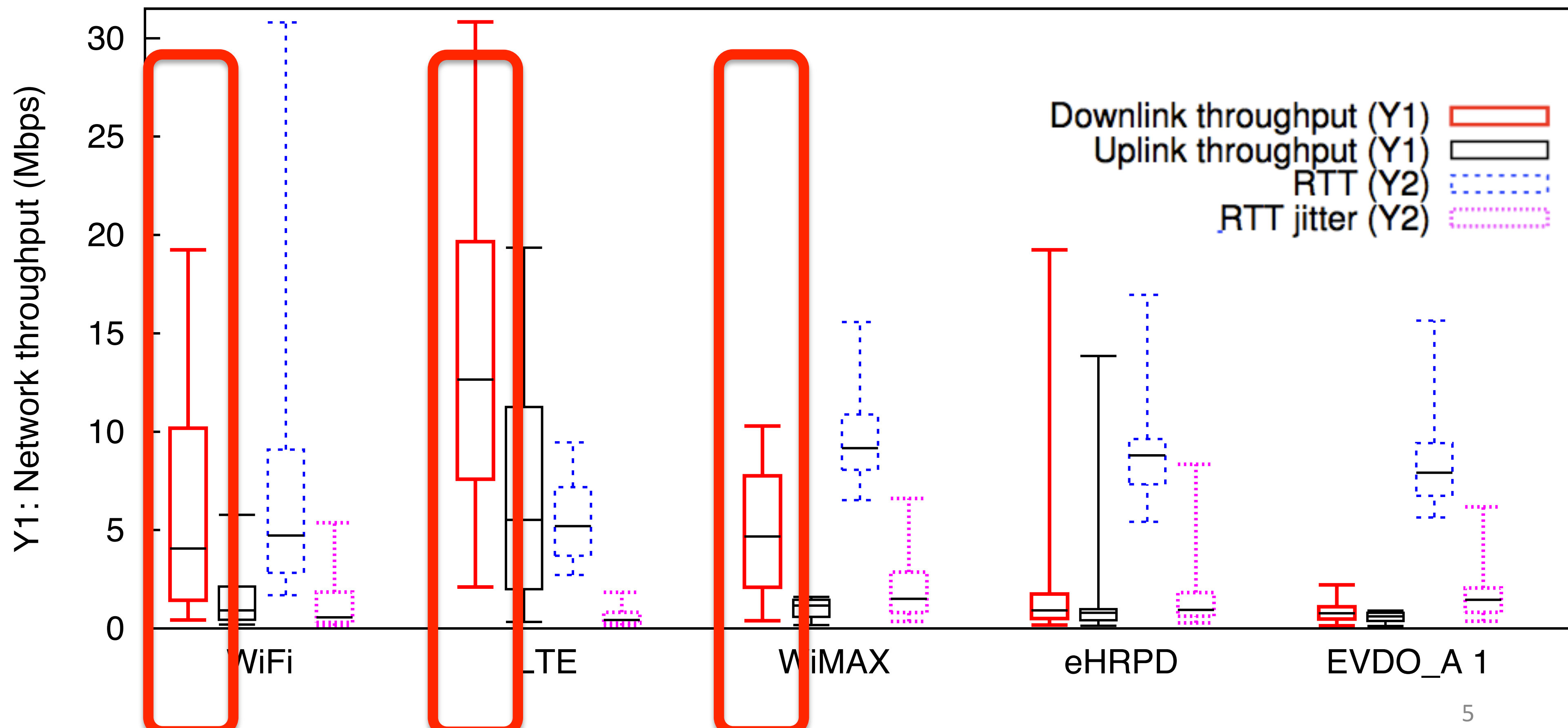


4GTest user coverage in the U.S.

Courtesy: Junxian Huang et al.

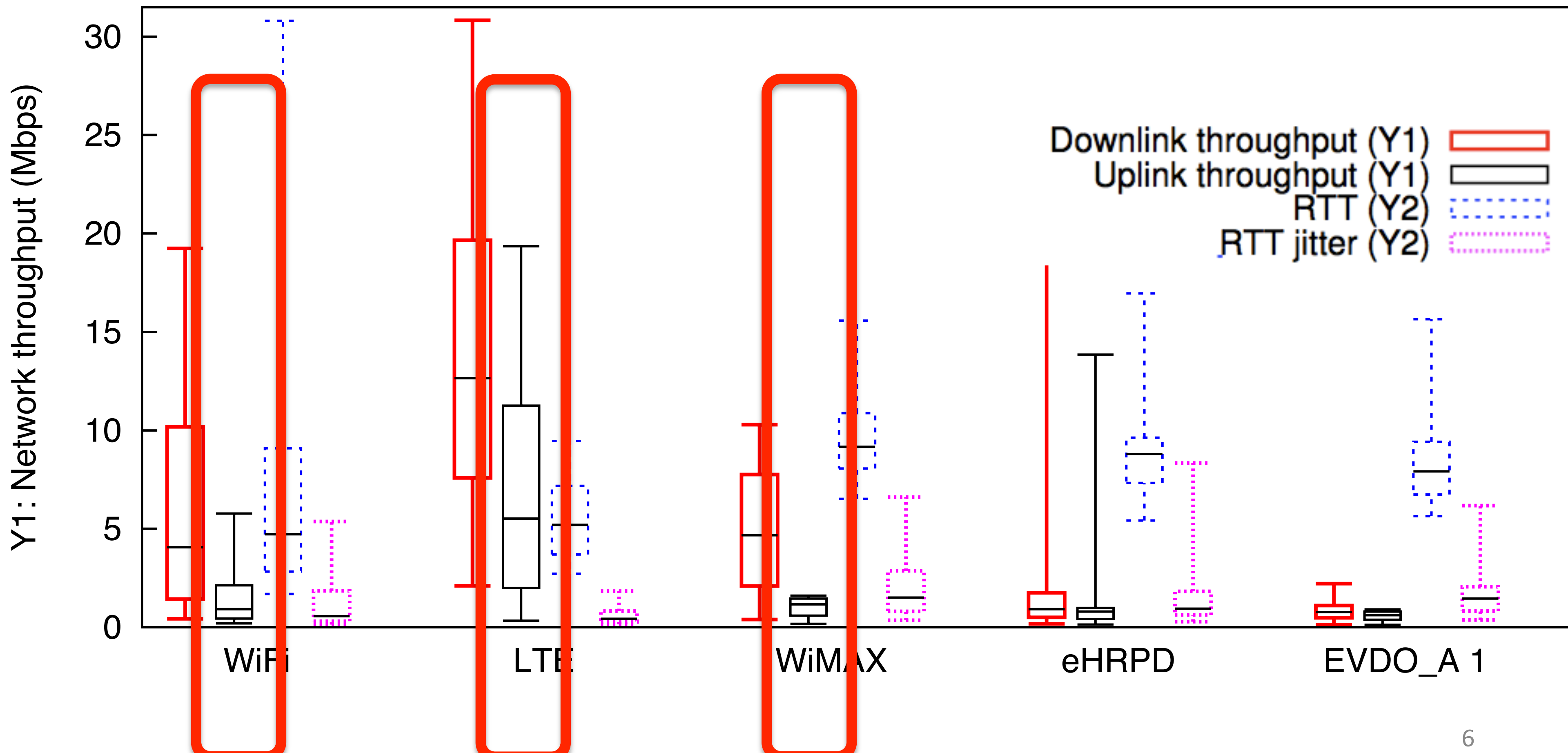
Downlink throughput

- LTE median is **13**Mbps, up to **30**Mbps
  - The LTE network is relatively unloaded
- WiFi, WiMAX **< 5**Mbps median



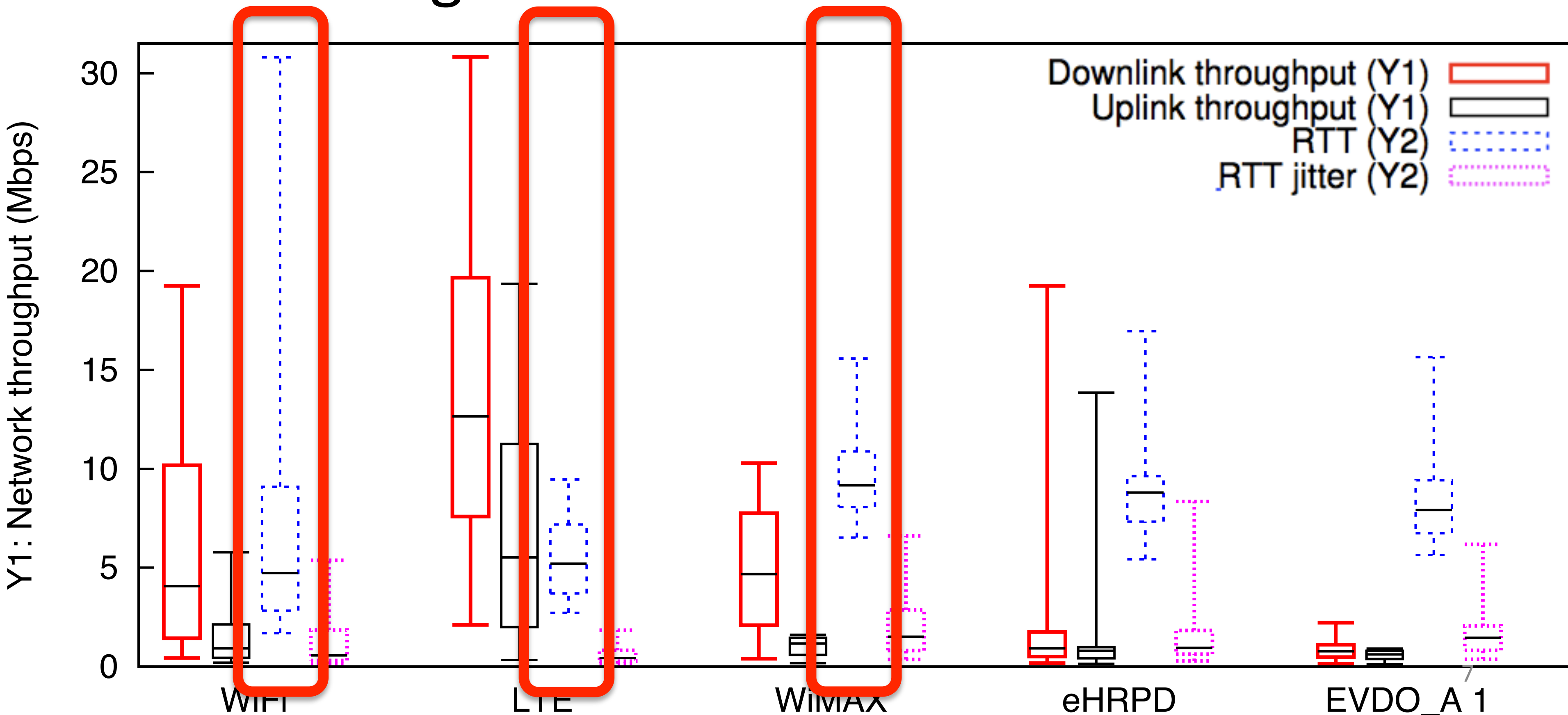
Uplink throughput

- LTE median is **5.6**Mbps, up to **20**Mbps
- WiFi, WiMAX **< 2**Mbps median



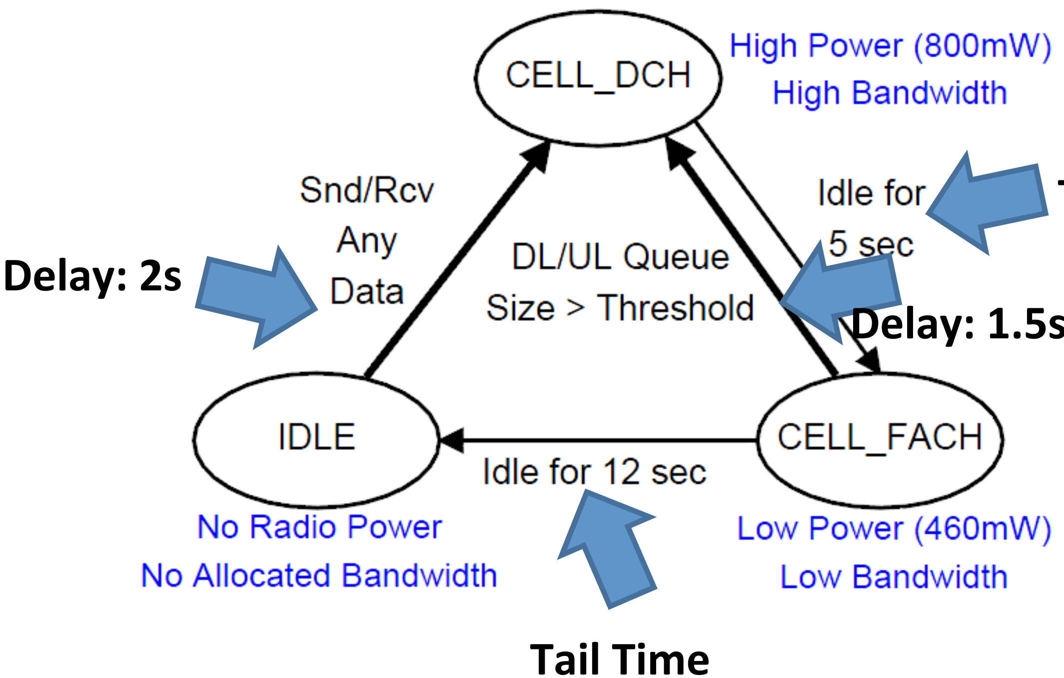
RTT

- LTE median **70**ms
- WiFi similar to LTE
- WiMAX higher



The RRC State Machine for UMTS Network

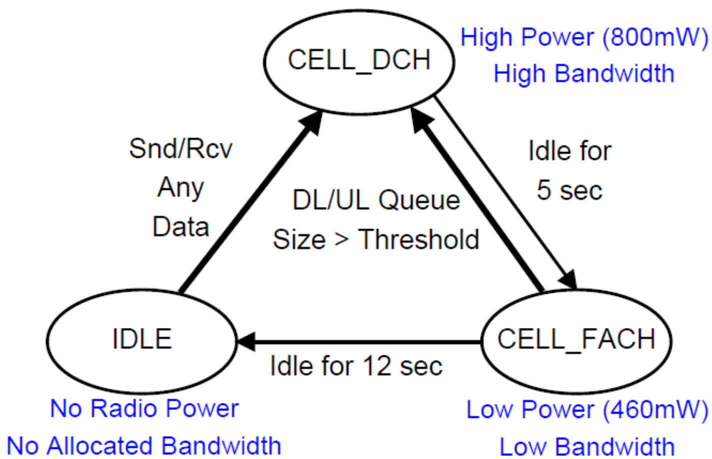
- State promotions have **promotion delay**
- State demotions incur **tail times**

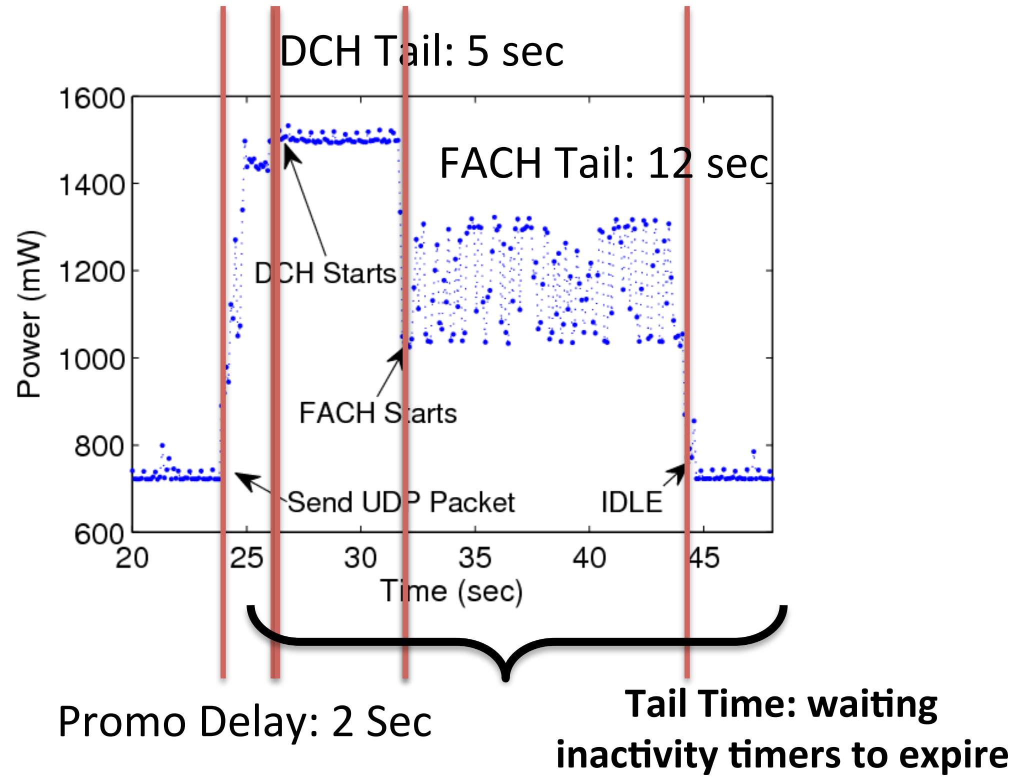


	Channel	Radio Power
IDLE	Not allocated	Almost zero
CELL_FACH	Shared, Low Speed	Low
CELL_DCH	Dedicated, High Speed	High

Courtesy: Feng Qian et al.

Example: RRC State Machine for a Large Commercial 3G Network





DCH: High Power State (high throughput and power consumption)

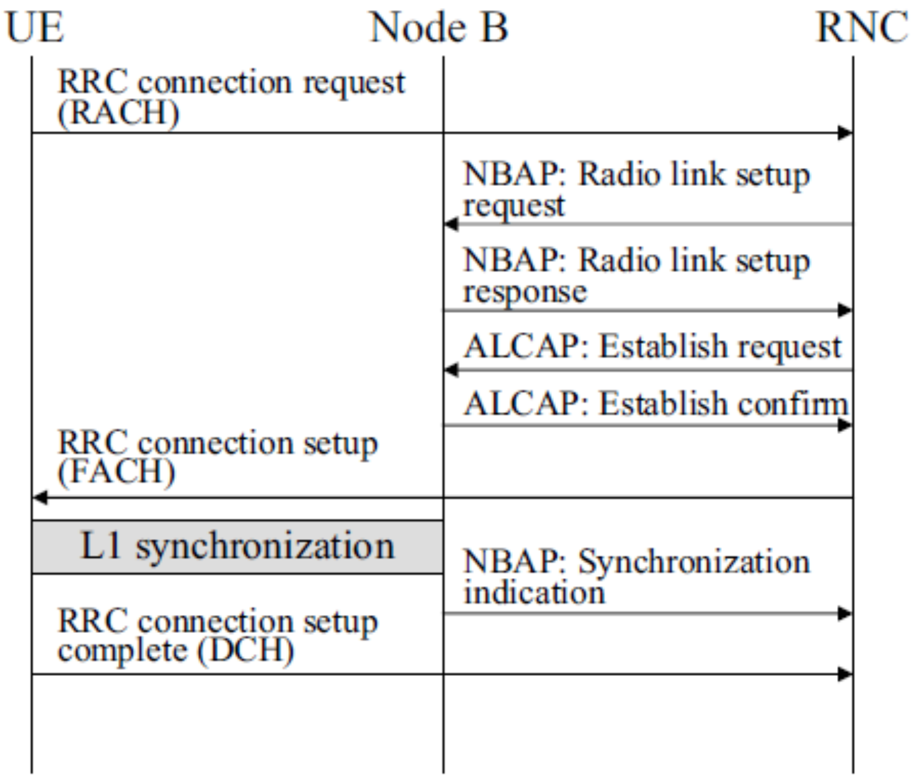
FACH: Low Power State (low throughput and power consumption)

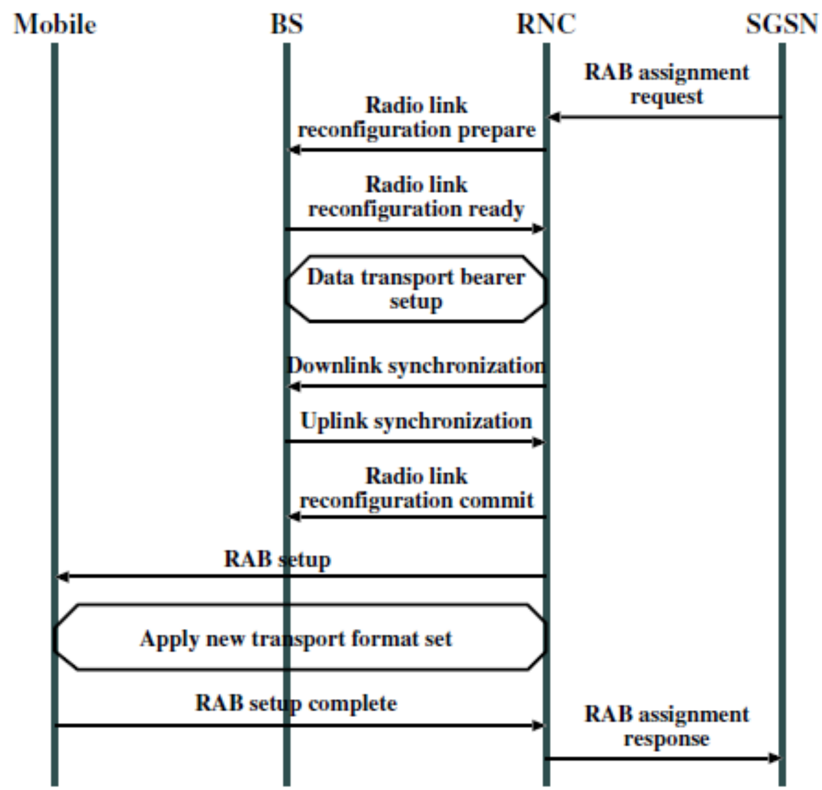
IDLE: No radio resource allocated

Courtesy: Feng Qian et al.

Why State Promotion Slow?

- Tens of control messages are exchanged during a state promotion.





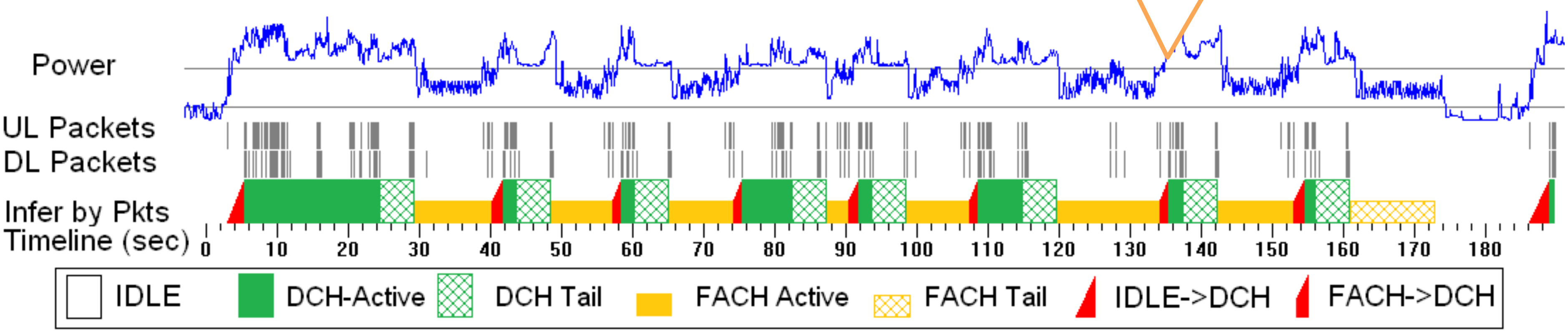
RRC connection setup: ~ 1sec + Radio Bearer Setup: ~ 1 sec

Figure source: HSDPA/HSUPA for UMTS: High Speed Radio Access for Mobile Communications. *John Wiley and Sons, Inc., 2006.*

Example of the State Machine Impact: Inefficient Resource Utilization

A significant amount of channel occupation time and battery life is wasted by **scattered bursts**.

State transitions impact end user experience and generate signaling load.

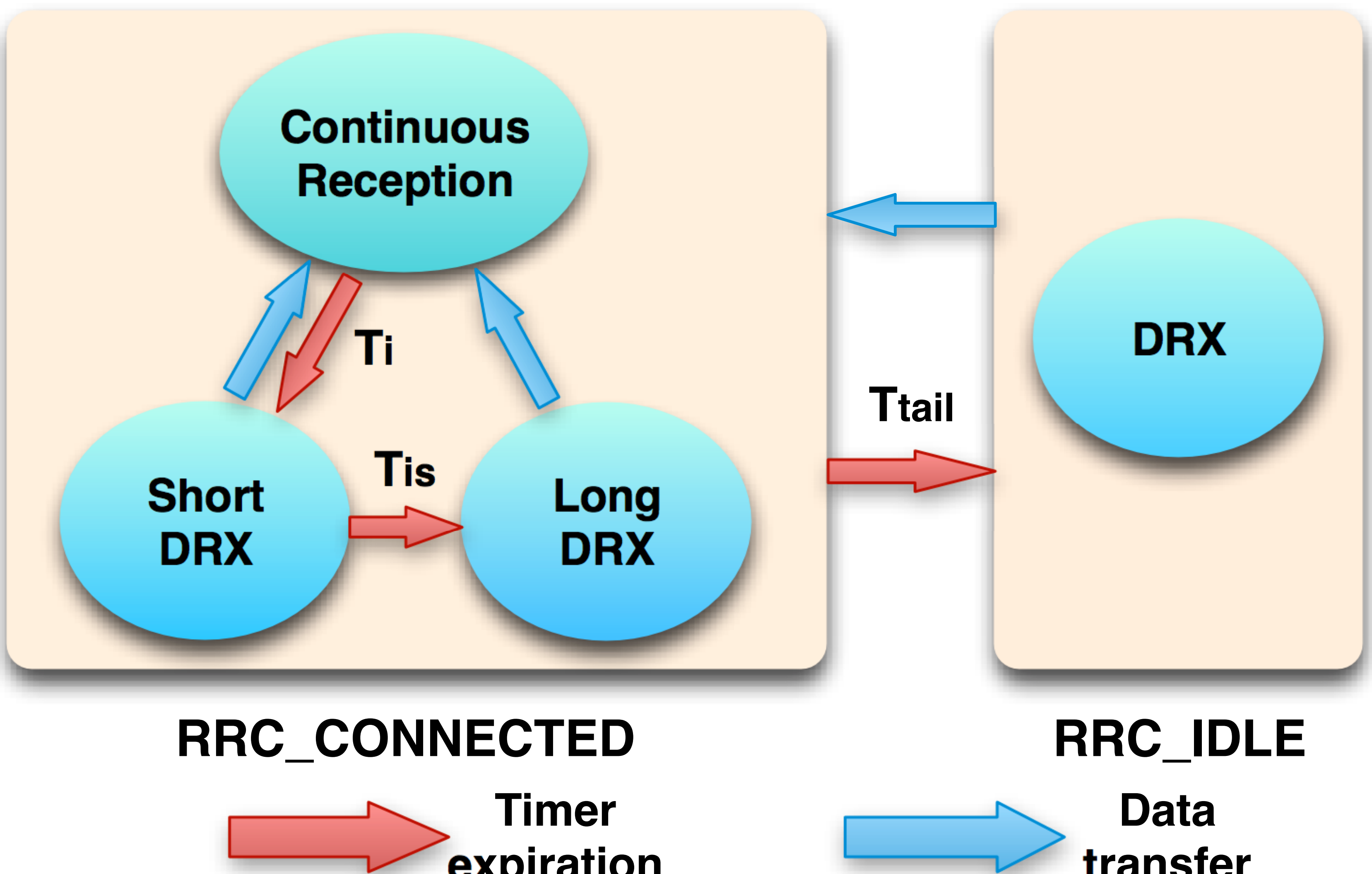


Analysis powered by the ARO tool

	FACH and DCH
Wasted Radio Energy	34%
Wasted Channel Occupation Time	33%

Courtesy: Feng Qian et al.

RRC state transitions in LTE

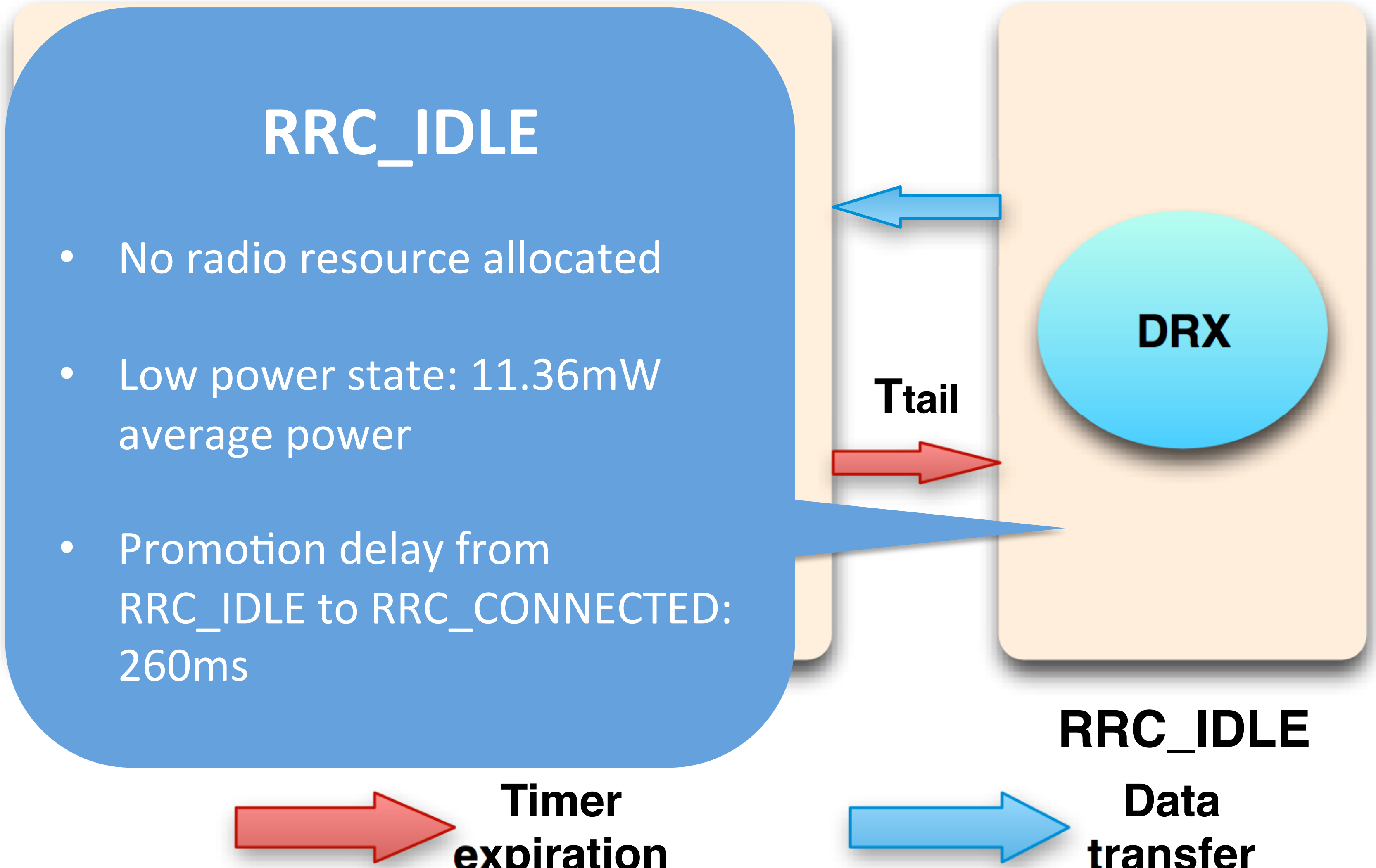


Courtesy: Junxian Huang et al.

RRC state transitions in LTE

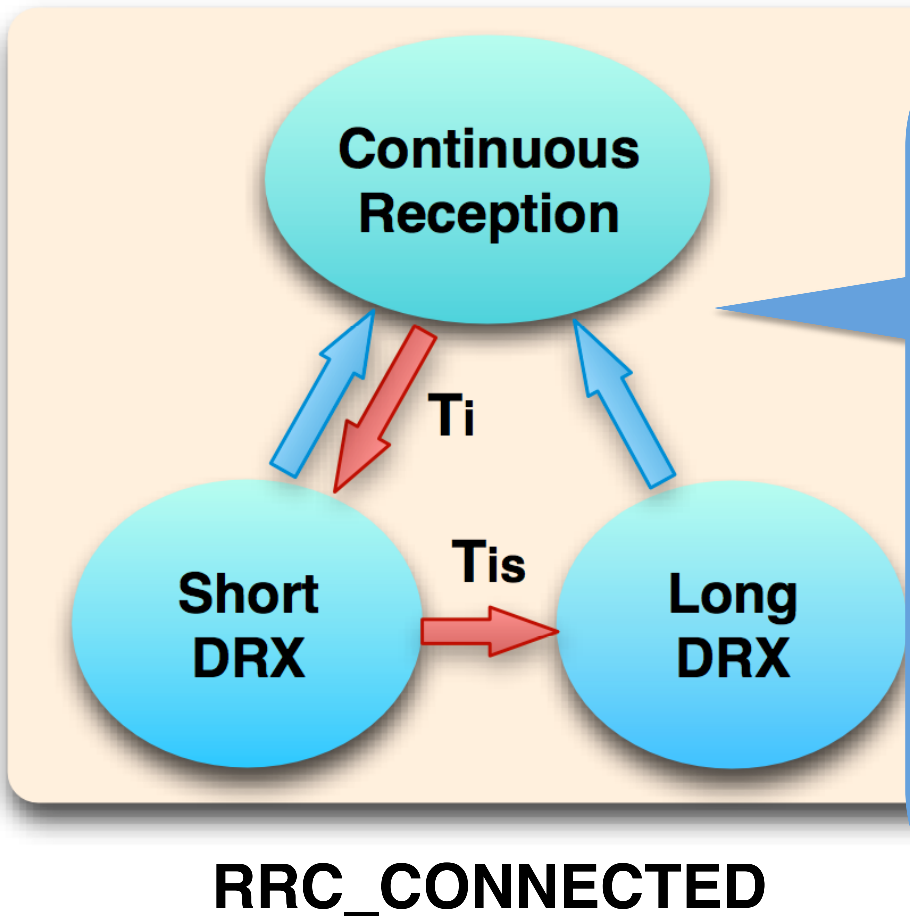
RRC_IDLE

- No radio resource allocated
- Low power state: 11.36mW average power
- Promotion delay from RRC_IDLE to RRC_CONNECTED: 260ms



Courtesy: Junxian Huang et al.

RRC state transitions in LTE

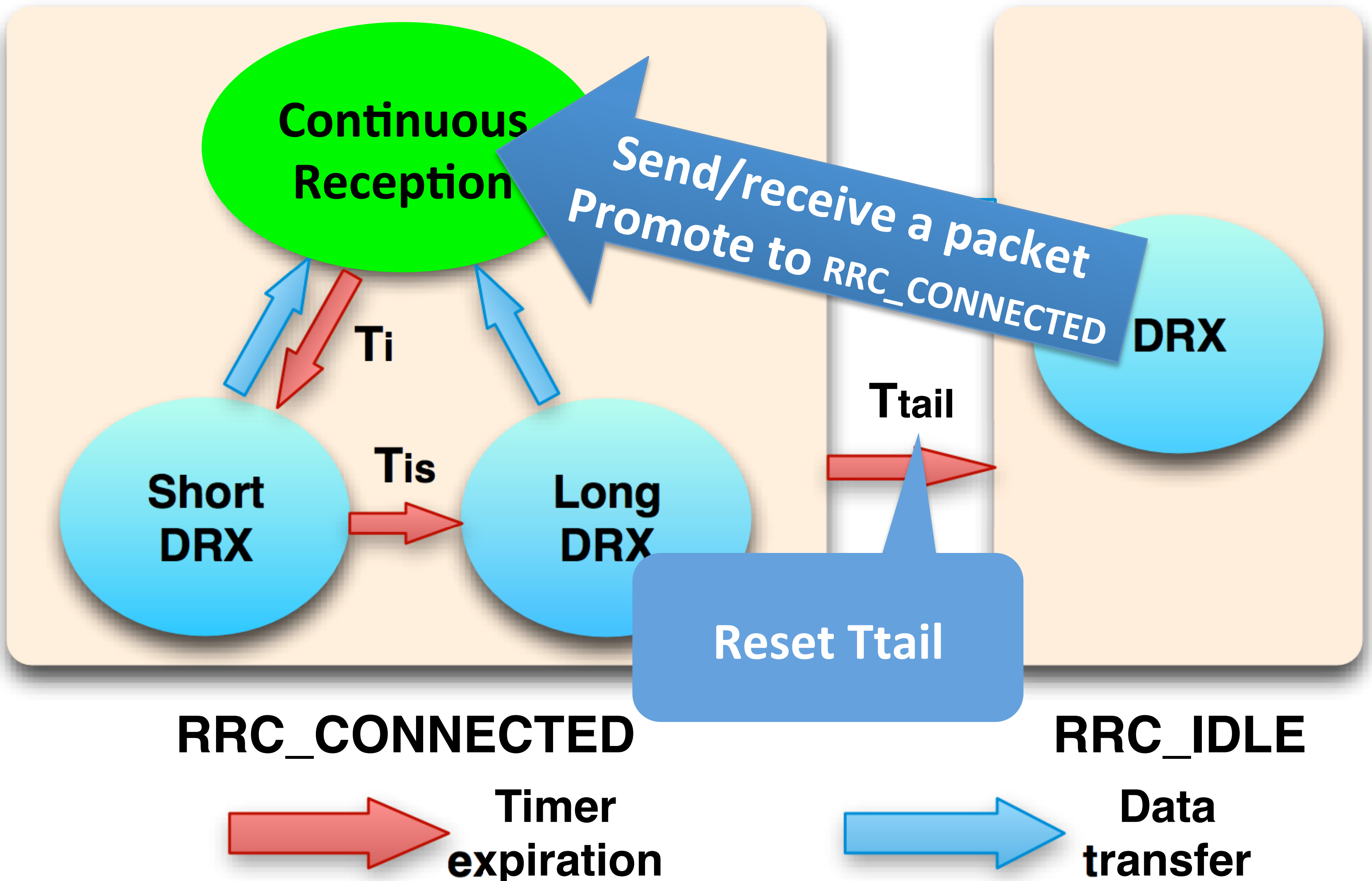


RRC_CONNECTED

- Radio resource allocated
- Power state is a function of data rate:
    - 1060mW is the base power consumption
    - Up to 3300mW transmitting at full speed

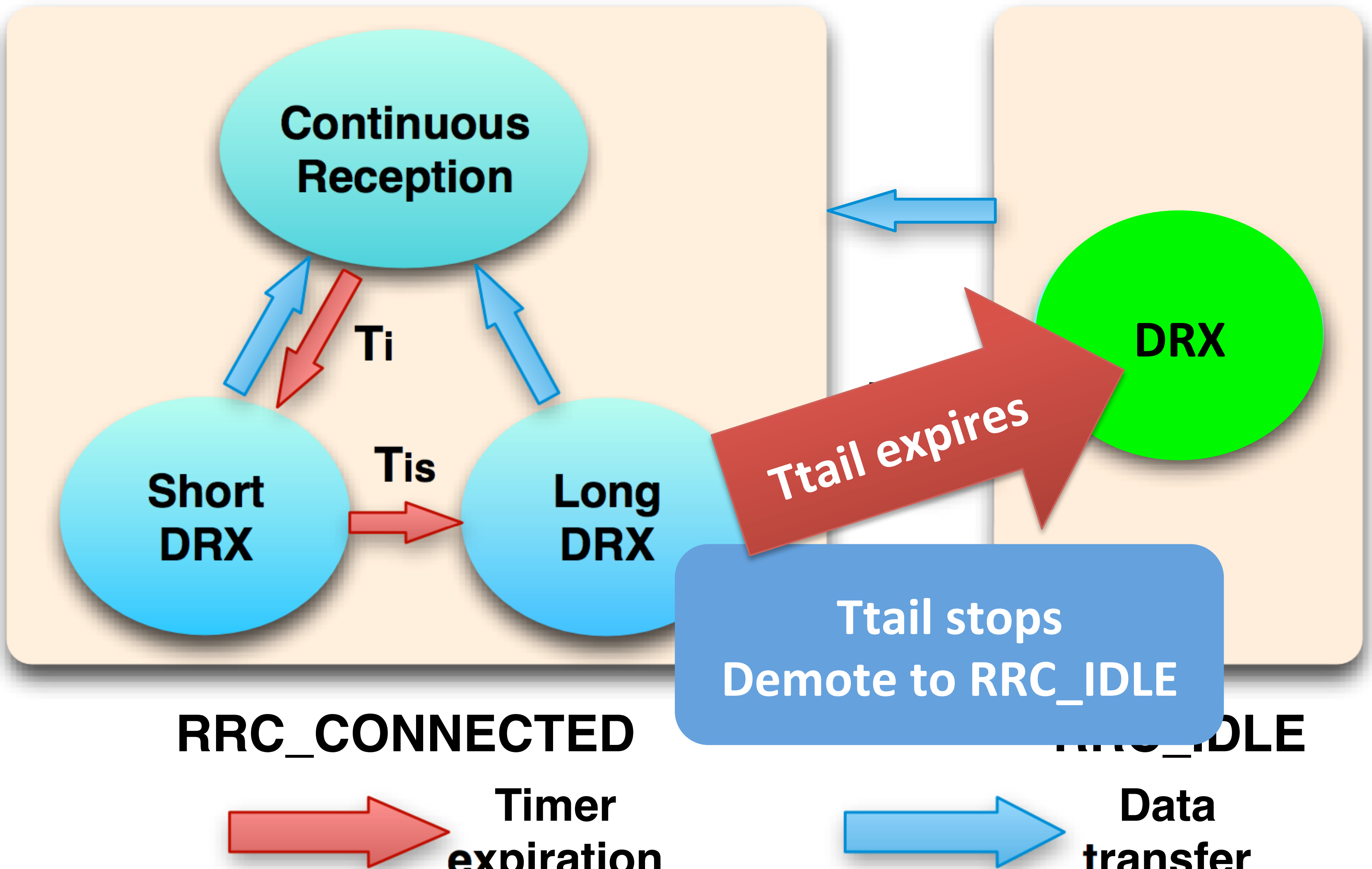
Courtesy: Junxian Huang et al.

RRC state transitions in LTE



Courtesy: Junxian Huang et al.

RRC state transitions in LTE



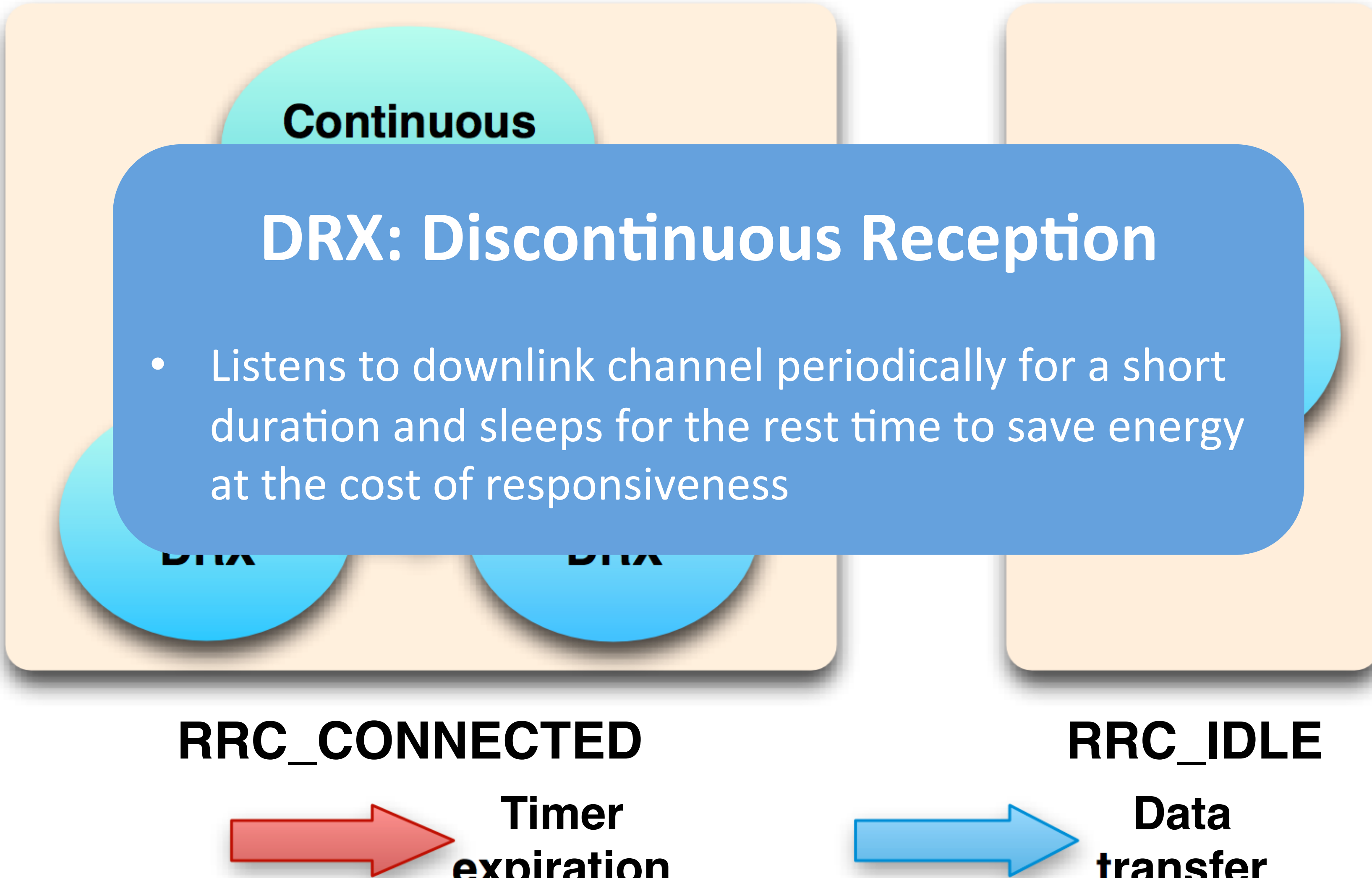
Courtesy: Junxian Huang et al.

Tradeoffs of *Ttail* settings

Ttail setting	Energy Consumption	# of state transitions	Responsiveness
Long	High	Small	Fast
Short	Low	Large	Slow

Courtesy: Junxian Huang et al.

RRC state transitions in LTE



DRX: Discontinuous Reception

- Listens to downlink channel periodically for a short duration and sleeps for the rest time to save energy at the cost of responsiveness

Courtesy: Junxian Huang et al.

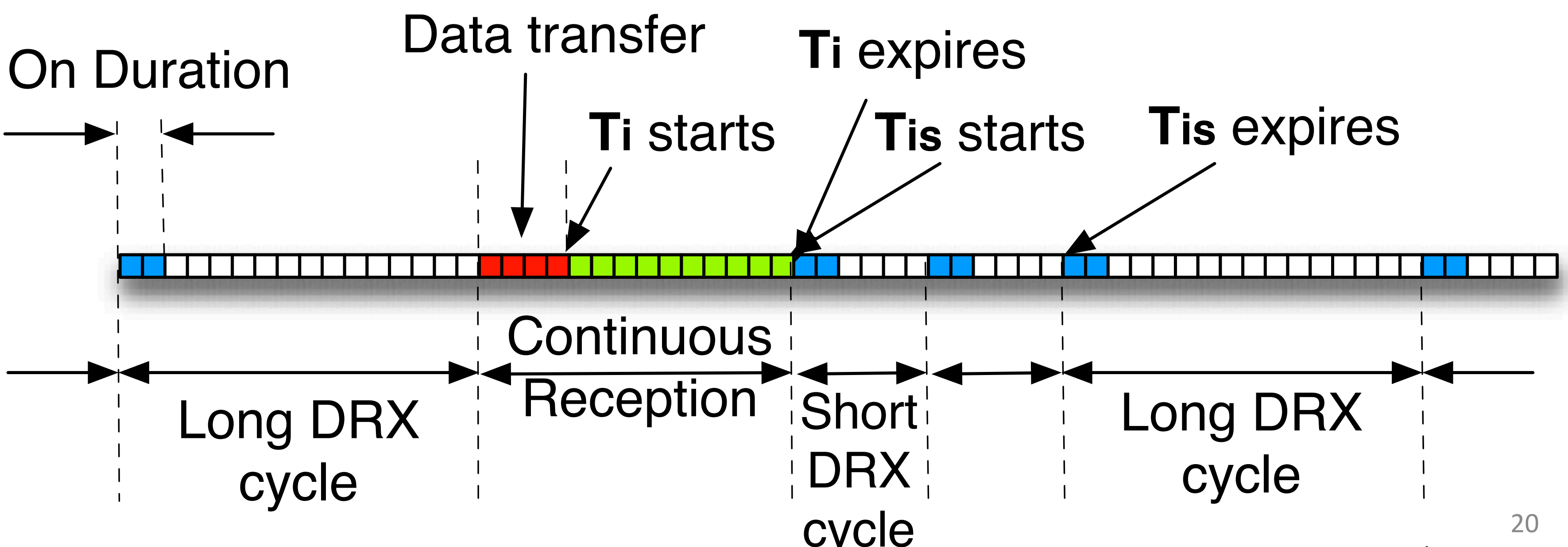
Discontinuous Reception (DRX): micro-sleeps for energy saving

- In LTE 4G, DRX makes UE ***micro-sleep periodically*** in the RRC_CONNECTED state
  - Short DRX
  - Long DRX
- DRX incurs tradeoffs between energy usage and latency
  - Short DRX – *sleep less and respond faster*
  - Long DRX – *sleep more and respond slower*
- In contrast, in UMTS 3G, UE is always listening to the downlink control channel in the data transmission states

Courtesy: Junxian Huang et al.

DRX in LTE

- A DRX cycle consists of
  - ‘On Duration’ - UE monitors the downlink control channel (PDCCH)
  - ‘Off Duration’ - skip reception of downlink channel
- T_i: Continuous reception inactivity timer
  - When to start Short DRX
- T_{is}: Short DRX inactivity timer
  - When to start Long DRX



Courtesy: Junxian Huang et al.

LTE power model

- Measured with a LTE phone and Monsoon power meter, averaged with repeated samples

	Power* (mW)	Duration (ms)	Periodicity (ms)
Screen off (base)	11.4±0.4	N/A	N/A
Screen 100% on	847.2±2.7	N/A	N/A
LTE promotion	1210.7±85.6	T_{pro}: 260.1±15.8	N/A
LTE Short DRX On in **RRC_CONNECTED**	1680.2±15.7	T_{on}: 1.0±0.1	T_{ps}: 20.0±0.1
LTE Long DRX On in **RRC_CONNECTED**	1680.1±14.3	T_{on}: 1.0±0.1	T_{pl}: 40.1±0.1
LTE Off Duration in **RRC_CONNECTED**	1060.0±3.3	T_{tail}: 11576.0±26.1	N/A
LTE DRX On in **RRC_IDLE**	594.3±8.7	T_{oni}: 43.2±1.5	T_{pi}: 1280.2±7.1

LTE power model

- Measured with a LTE phone and Monsoon power meter, averaged with repeated samples

	Power* (mW)	Duration (ms)	Periodicity (ms)
Screen off (base)	11.4±0.4	N/A	N/A
Screen 100% on	847.2±2.7	N/A	N/A
LTE promotion	1210.7±85.6	T_{pro}: 260.1±15.8	N/A
LTE Short DRX On in **RRC_CONNECTED**	1680.2±15.7	T_{on}: 1.0±0.1	T_{ps}: 20.0±0.1
LTE Long DRX On in **RRC_CONNECTED**	1680.1±14.3	T_{on}: 1.0±0.1	T_{pl}: 40.1±0.1
LTE Off Duration in **RRC_CONNECTED**	1060.0±3.3	T_{tail}: 11576.0±26.1	N/A
LTE DRX On in **RRC_IDLE**	594.3±8.7	T_{oni}: 43.2±1.5	T_{pi}: 1280.2±7.1

LTE power model

- Measured with a LTE phone and Monsoon power meter, averaged with repeated samples

	Power* (mW)	Duration (ms)	Periodicity (ms)
Screen off (base)	11.4±0.4	N/A	N/A
Screen 100% on	847.2±2.7	N/A	N/A
LTE promotion	1210.7±85.6	T_{pro}: 260.1±15.8	N/A
LTE Short DRX On in **RRC_CONNECTED**	1680.2±15.7	T_{on}: 1.0±0.1	T_{ps}: 20.0±0.1
LTE Long DRX On in **RRC_CONNECTED**	1680.1±14.3	T_{on}: 1.0±0.1	T_{pl}: 40.1±0.1
LTE Off Duration in **RRC_CONNECTED**	1060.0±3.3	T_{tail}: 11576.0±26.1	N/A
LTE DRX On in **RRC_IDLE**	594.3±8.7	T_{oni}: 43.2±1.5	T_{pi}: 1280.2±7.1

LTE power model

- Measured with a LTE phone and Monsoon power meter, averaged with repeated samples

	Power* (mW)	Duration (ms)	Periodicity (ms)
Screen off (base)	11.4±0.4	N/A	N/A
Screen 100% on	847.2±2.7	N/A	N/A
LTE promotion	1210.7±85.6	T_{pro}: 260.1±15.8	N/A
LTE Short DRX On in **RRC_CONNECTED**	1680.2±15.7	T_{on}: 1.0±0.1	T_{ps}: 20.0±0.1
LTE Long DRX On in **RRC_CONNECTED**	1680.1±14.3	T_{on}: 1.0±0.1	T_{pl}: 40.1±0.1
LTE Off Duration in **RRC_CONNECTED**	1060.0±3.3	T_{tail}: 11576.0±26.1	N/A
LTE DRX On in **RRC_IDLE**	594.3±8.7	T_{oni}: 43.2±1.5	T_{pi}: 1280.2±7.1

LTE power model

- Measured with a LTE phone and Monsoon power meter, averaged with repeated samples

	Power⋆	Duration	Periodicity (ms)
Scre[illegible]	[illegible]	[illegible]	N/A
Scre[illegible]	[illegible]	[illegible]	N/A
LTE[illegible]	[illegible]	[illegible]	N/A
LTE S[illegible] in **RRC**[illegible]	[illegible]	[illegible]	T_{ps}: [illegible]±0.1
LTE L[illegible] in **RRC**[illegible]	[illegible]	[illegible]	T_{pl}: [illegible]±0.1
LTE C[illegible] in **RRC_CONNECTED**	[illegible]	[illegible]	N/A
LTE DRX On in **RRC_IDLE**	594.3±8.7	T_{oni}: 43.2±1.5	T_{pi}: 1280.2±7.1

- **P(on) – P(off) = 620mW, DRX saves 36% energy in RRC_CONNECTED**
- **High power levels in *both* On and Off durations in the DRX cycle of RRC_CONNECTED**

LTE consumes more instant power than 3G/WiFi in the high-power tail

- Average power for WiFi tail
  - **120** mW
- Average power for 3G tail
  - **800** mW
- Average power for LTE tail
  - **1080** mW

Courtesy: Junxian Huang et al.

Power model for data transfer

- A linear model is used to quantify instant power level:
  - Downlink throughput t_d Mbps
  - Uplink throughput t_u Mbps

$$P = \alpha_u t_u + \alpha_d t_d + \beta$$

Data transfer power model

< 6% error rate in evaluations with real applications

Courtesy: Junxian Huang et al.

Energy per bit comparison

- LTE's high throughput compensates for the promotion energy and tail energy

Transfer Size	LTE μ J / bit	WiFi μ J / bit	3G μ J / bit
10KB	170	6	100
10MB	0.3	0.1	4

Total energy per bit for downlink bulk data transfer

Courtesy: Junxian Huang et al.

Energy per bit comparison

- LTE's high throughput compensates for the promotion energy and tail energy

Small data transfer, LTE wastes energy
Large data transfer, LTE is energy efficient

10MB	**0.3**	0.1	4

Total energy per bit for downlink bulk data transfer

Courtesy: Junxian Huang et al.

Example of the State Machine Impact: DNS timeout in UMTS networks

Start from CELL_DCH STATE (1 request / response) – Keep in DCH

240	11.806360	10.0.192.152	172.18.145.103	DNS	Standard query A www.eecs.umich.edu
241	11.991680	172.18.145.103	10.0.192.152	DNS	Standard query response A 141.212.113.110

Start from CELL_FACH STATE (1 request / response) – Keep in FACH

237	18.304116	10.0.192.152	172.18.145.103	DNS	Standard query A www.eecs.umich.edu
238	18.627700	172.18.145.103	10.0.192.152	DNS	Standard query response A 141.212.113.110

Start from IDLE STATE (2~3 requests / responses) – IDLE → DCH

1884	221.958559	10.0.192.152	172.18.145.103	DNS	Standard query A www.eecs.umich.edu
1885	222.947858	10.0.192.152	172.18.145.103	DNS	Standard query A www.eecs.umich.edu
1886	223.947892	10.0.192.152	172.18.145.103	DNS	Standard query A www.eecs.umich.edu
1887	224.069453	172.18.145.103	10.0.192.152	DNS	Standard query response A 141.212.113.110
1888	224.070465	172.18.145.103	10.0.192.152	DNS	Standard query response A 141.212.113.110
1889	224.079463	172.18.145.103	10.0.192.152	DNS	Standard query response A 141.212.113.110

Starting from IDLE triggers at least one DNS timeout (default is 1 sec in WinXP)

2 second promotion delay because of the wireless state machine (see previous slide), but DNS timeout is 1 second!

=> Triple the volume of DNS requests…

Courtesy: Feng Qian et al.

State Machine Inference

- State Promotion Inference
  - Determine one of the two promotion procedures
  - P1: IDLE→FACH→DCH; P2:IDLE→DCH

Algorithm 1 State promotion inference

1: Keep UE on IDLE.
2: UE sends *min* bytes. Server echoes *min* bytes. ← P1: IDLE→FACH, P2:IDLE→DCH
3: UE sends *max* bytes. Server echoes *min* bytes. ← P1: FACH→DCH, P2:Keep on DCH
4: UE records the RTT Δt for Step 3. ← Normal RTT < 300ms
5: Report $P1$ iff $\Delta t \gg$ normal RTT. Otherwise report $P2$. RTT w/ Promo > 1500ms

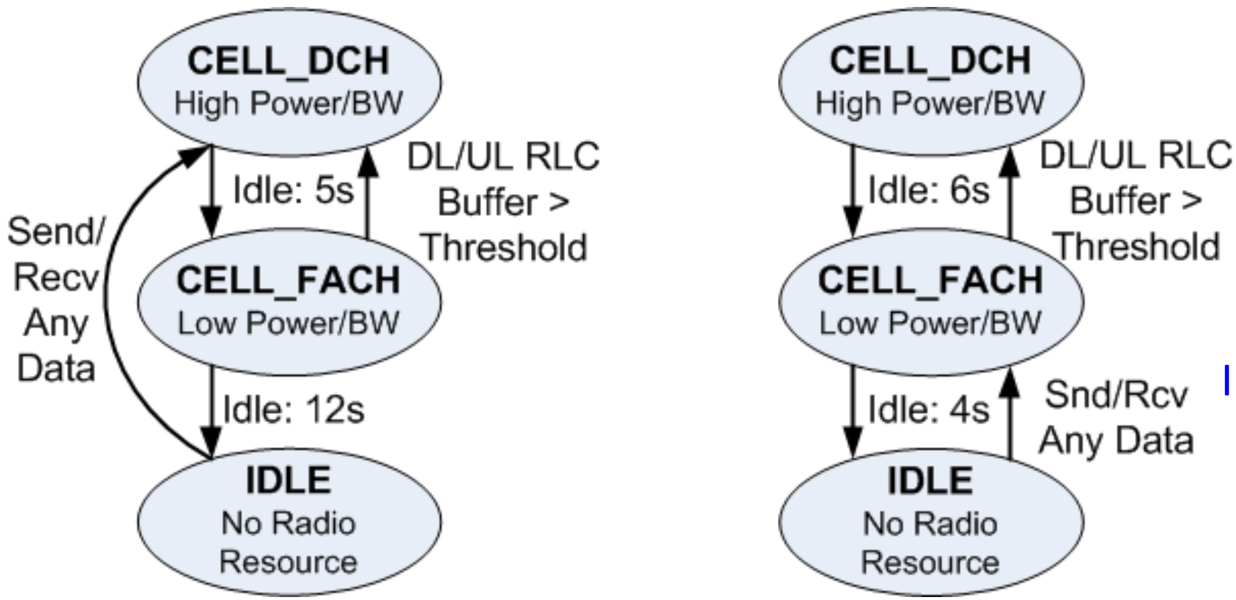
A packet of **min** bytes **never** triggers FACH→DCH promotion (we use 28B)
A packet of **max** bytes **always** triggers FACH→DCH promotion (we use 1KB)

- State demotion and inactivity timer inference
  - See paper for details

Courtesy: Feng Qian et al.

RRC State Machines of Two Commercial UMTS Carriers

Promotion Inference Reports **P2** IDLE→DCH



Carrier 1

Carrier 2

Promotion Inference Reports **P1** IDLE→FACH→DCH

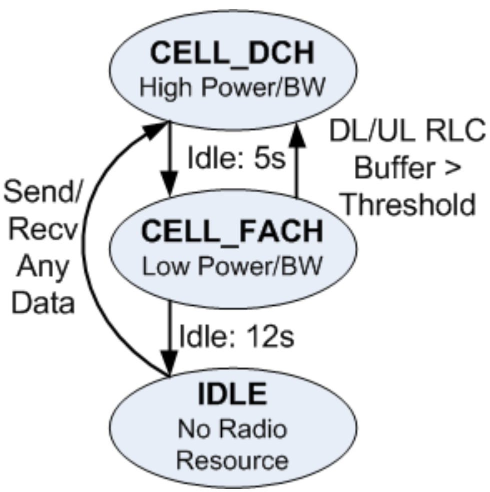
Timer	Carrier 1	Carrier 2
DCH→FACH (α timer)	5 sec	6 sec
FACH→IDLE (β timer)	12 sec	4 sec

What are the optimal inactivity timer values?

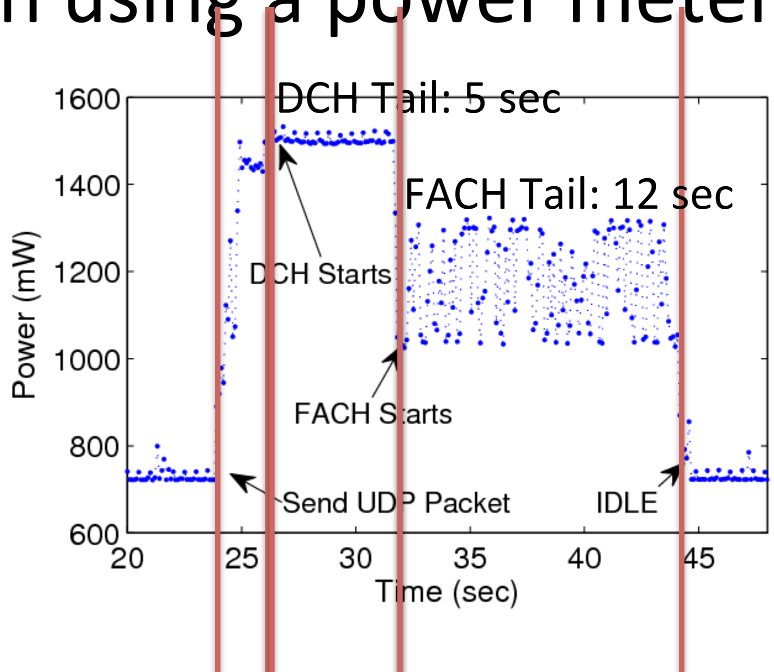
Courtesy: Feng Qian et al.

State Machine Inference

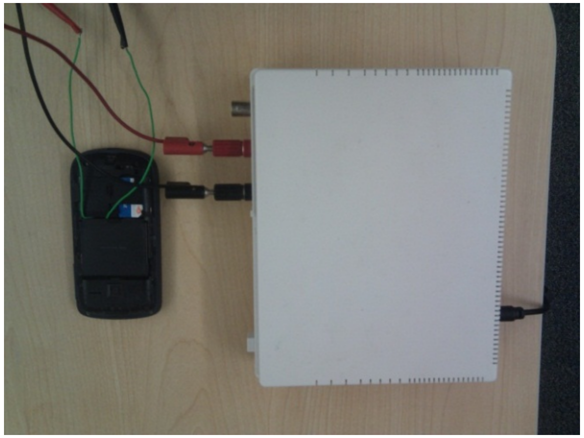
- Validation using a power meter



Carrier 1



Promo Delay: 2 Sec



RRC State	Avg Radio Power
IDLE	0
FACH	460 mW
DCH	800 mW
FACH→DCH	700 mW
IDLE→DCH	550 mW

Outline

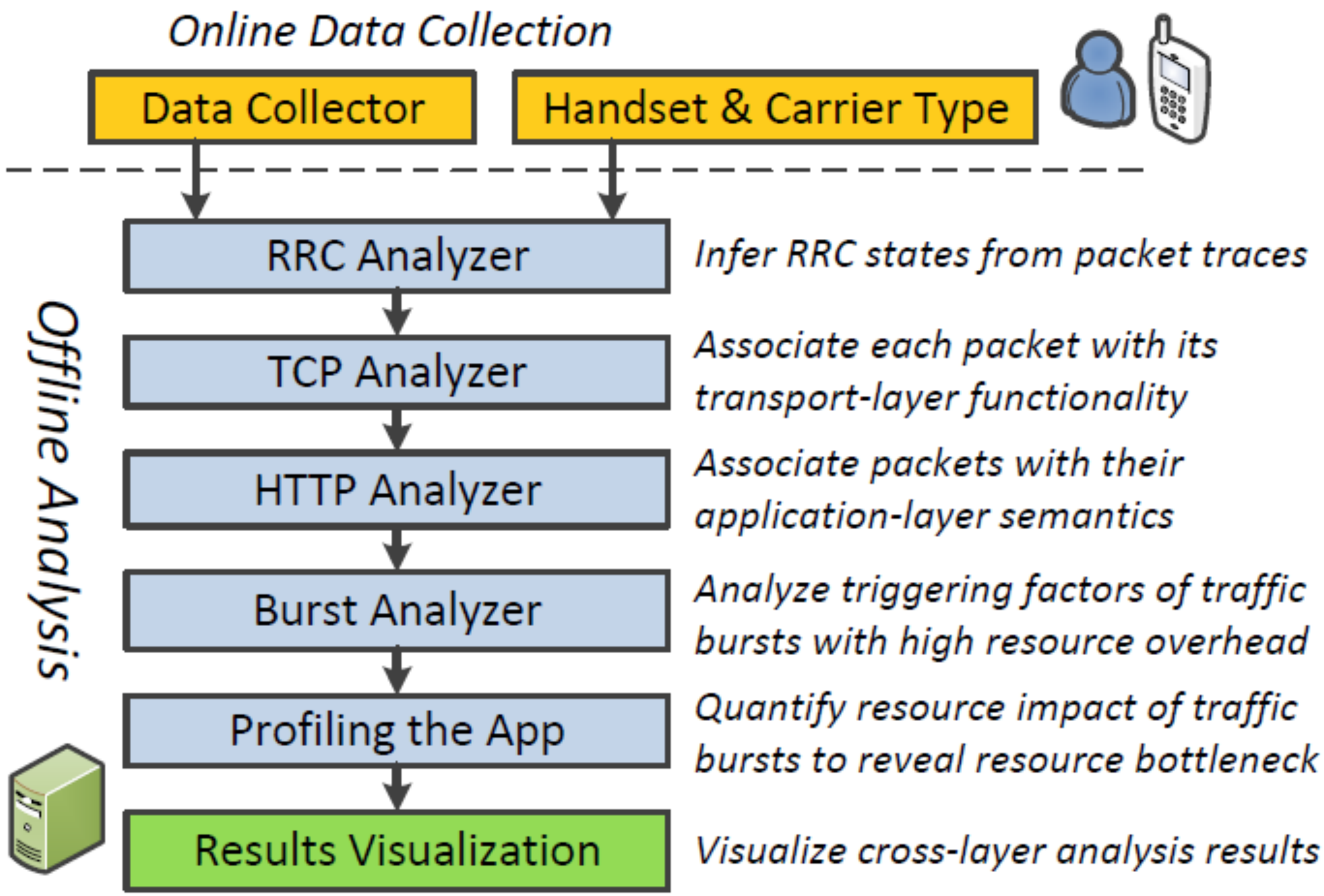
- Introduction
- RRC State Inference
- Radio Resource Usage Profiling & Optimization
- Network RRC Parameters Optimization
- Conclusion

ARO: Mobile Application Resource Optimizer

- **Motivations:**
  - Are developers aware of the RRC state machine and its implications on radio resource / energy? **NO.**
  - Do they need a tool for automatically profiling their prototype applications? **YES.**
  - If we provide that visibility, would developers optimize their applications and reduce the network impact? **Hopefully YES.**
- **ARO: Mobile Application Resource Optimizer**
  - Provide visibility of radio resource and energy utilization.
  - Benchmark efficiencies of cellular radio resource and battery life for a specific application

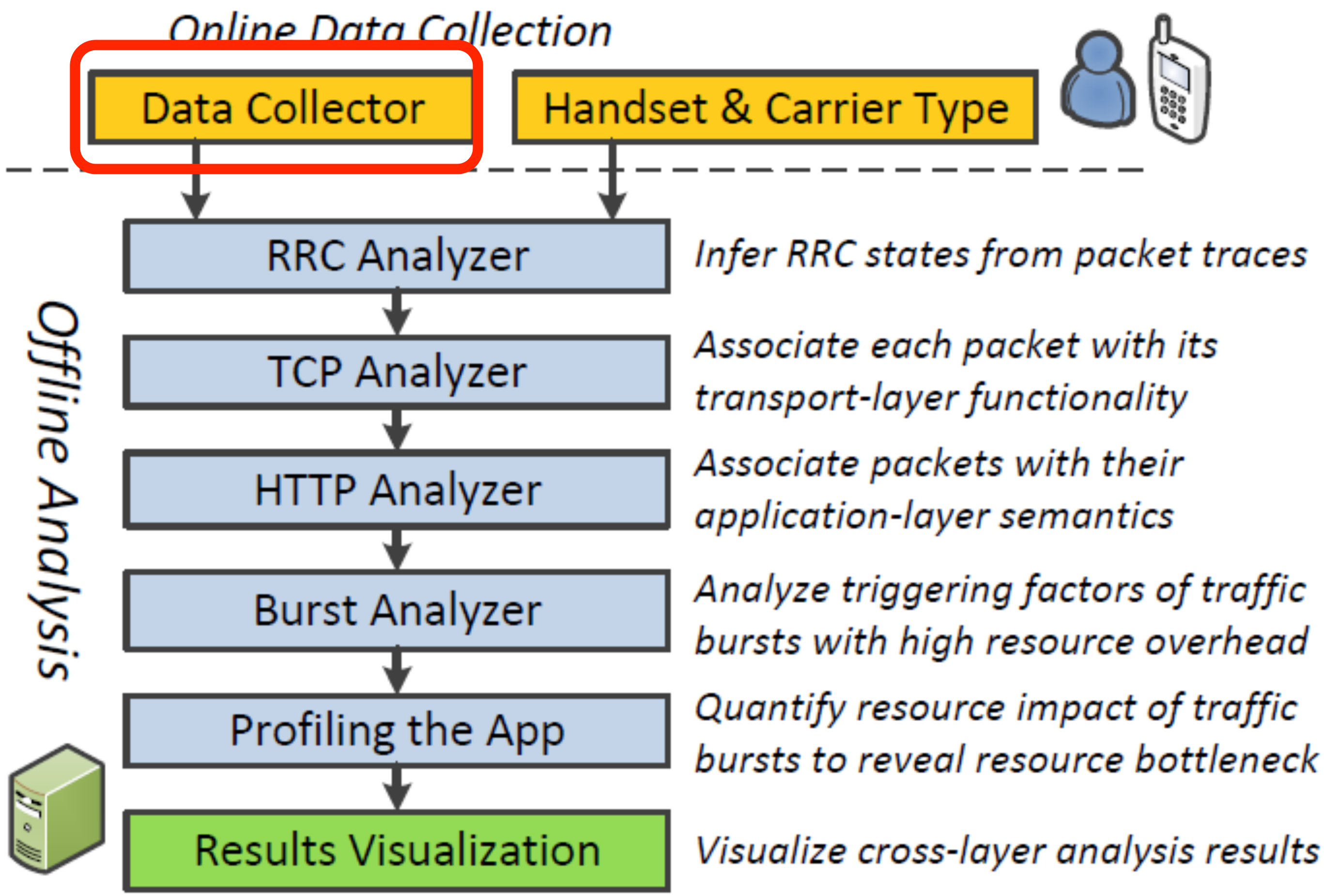
Courtesy: Feng Qian et al.

ARO System Architecture



Courtesy: Feng Qian et al.

ARO System Architecture



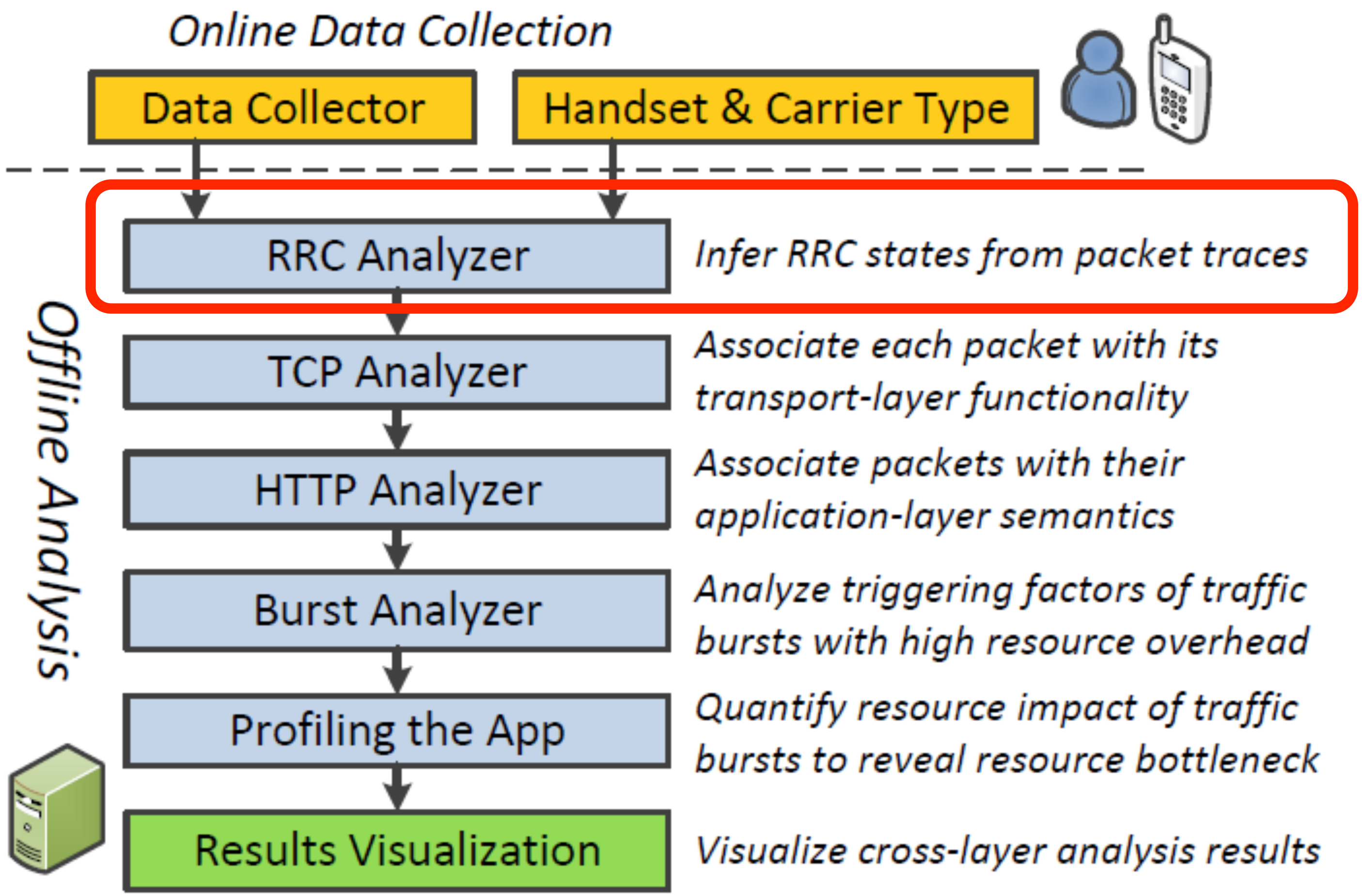
Courtesy: Feng Qian et al.

The Data Collector

- Collects three pieces of information
  - The packet trace
  - User input (e.g., touching the screen)
  - Packet-process correspondence
    - The RRC state transition is triggered by the **aggregated** traffic of all concurrent applications
    - But we are only interested in our **target** application.
- Less than 15% runtime overhead when the throughput is as high as 600kbps

Courtesy: Feng Qian et al.

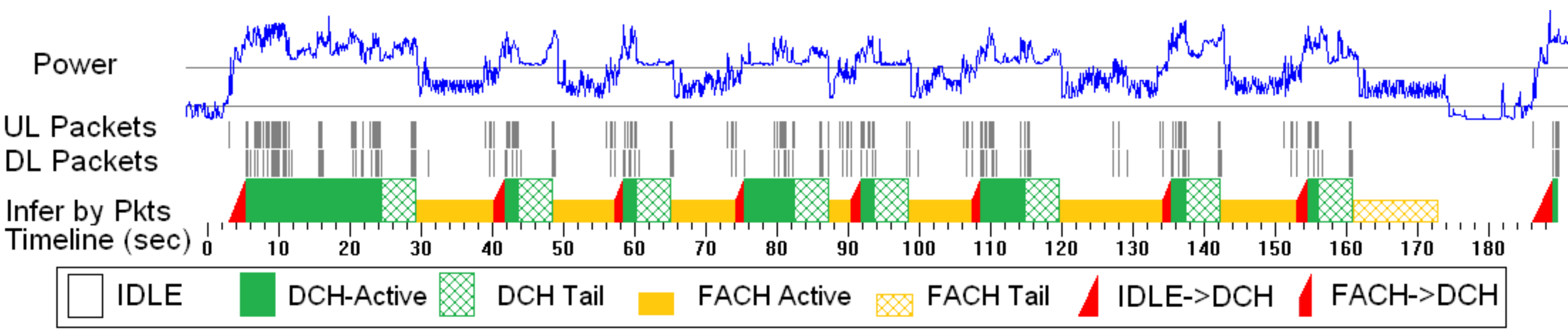
ARO System Architecture



Courtesy: Feng Qian et al.

RRC Analyzer: State Inference

- RRC state inference
  - Taking the packet trace as input, **simulate** the RRC state machine to infer the RRC states
    - Iterative packet driven simulation: given RRC state known for pkt_i, infer state for pkt_{i+1} based on inter-arrival time, packet size and UL/DL
  - Evaluated by measuring the device power



Example: Web Browsing Traffic on HTC TyTn II Smartphone

Courtesy: Feng Qian et al.

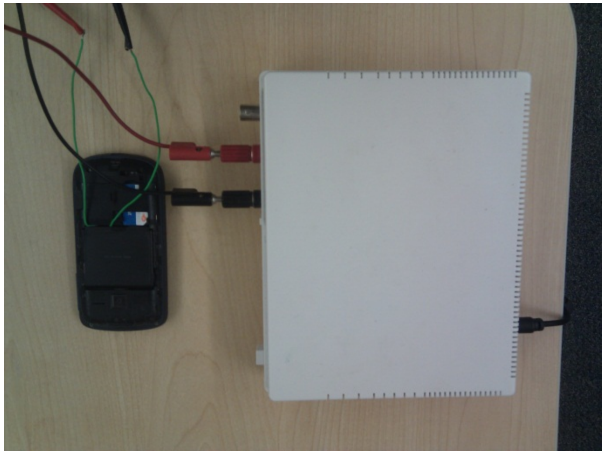
RRC Analyzer: Applying the Energy Model

- Apply the energy model
  - Associate each state with a constant power value
  - Based on our measurement using a power-meter

Table 3: Measured average radio power consumption

	TyTn Carrier 1	NexusOne Carrier 1	ADP1 * T-Mobile
P(IDLE)	0	0	10mW
P(FACH)	460mW	450mW	401mW
P(DCH)	800mW	600mW	570mW
P(FACH→DCH)	700mW	550mW	N/A
P(IDLE→DCH)	550mW	530mW	N/A

* Reported by [27] for Android HTC Dream phone



Courtesy: Feng Qian et al.

RRC Analyzer: Applying the Energy Model (Cont'd)

- 3G radio interface power consumption
  - at DCH, the radio power (800 mW) contributes 1/3 to 1/2 of total device power (1600 mW to 2400 mW)

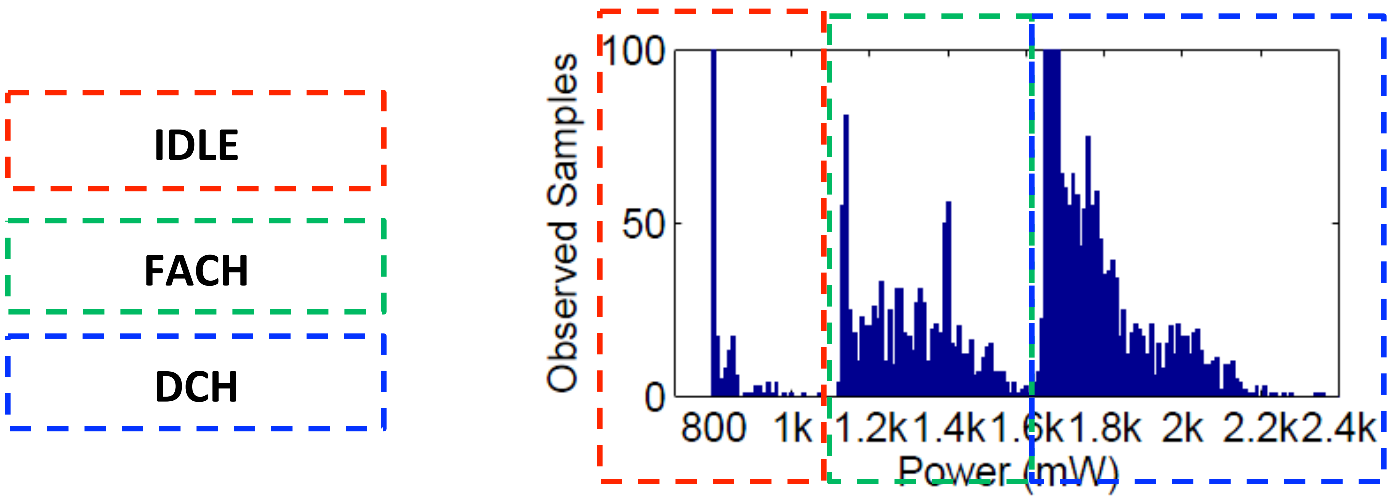
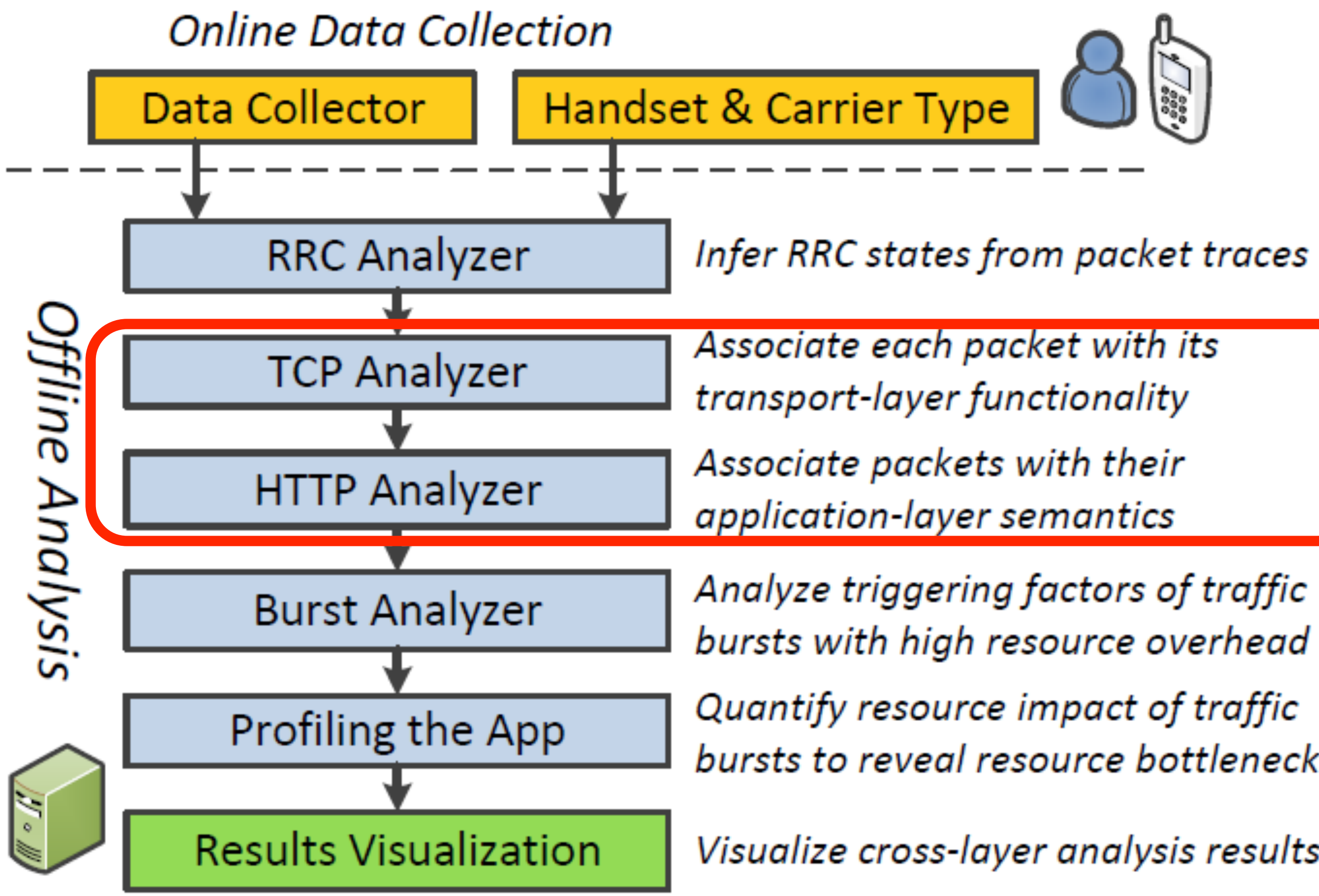


Figure 8: Histogram of measured power values for the News1 trace collected at an HTC TyTn II phone

Courtesy: Feng Qian et al.

ARO System Architecture



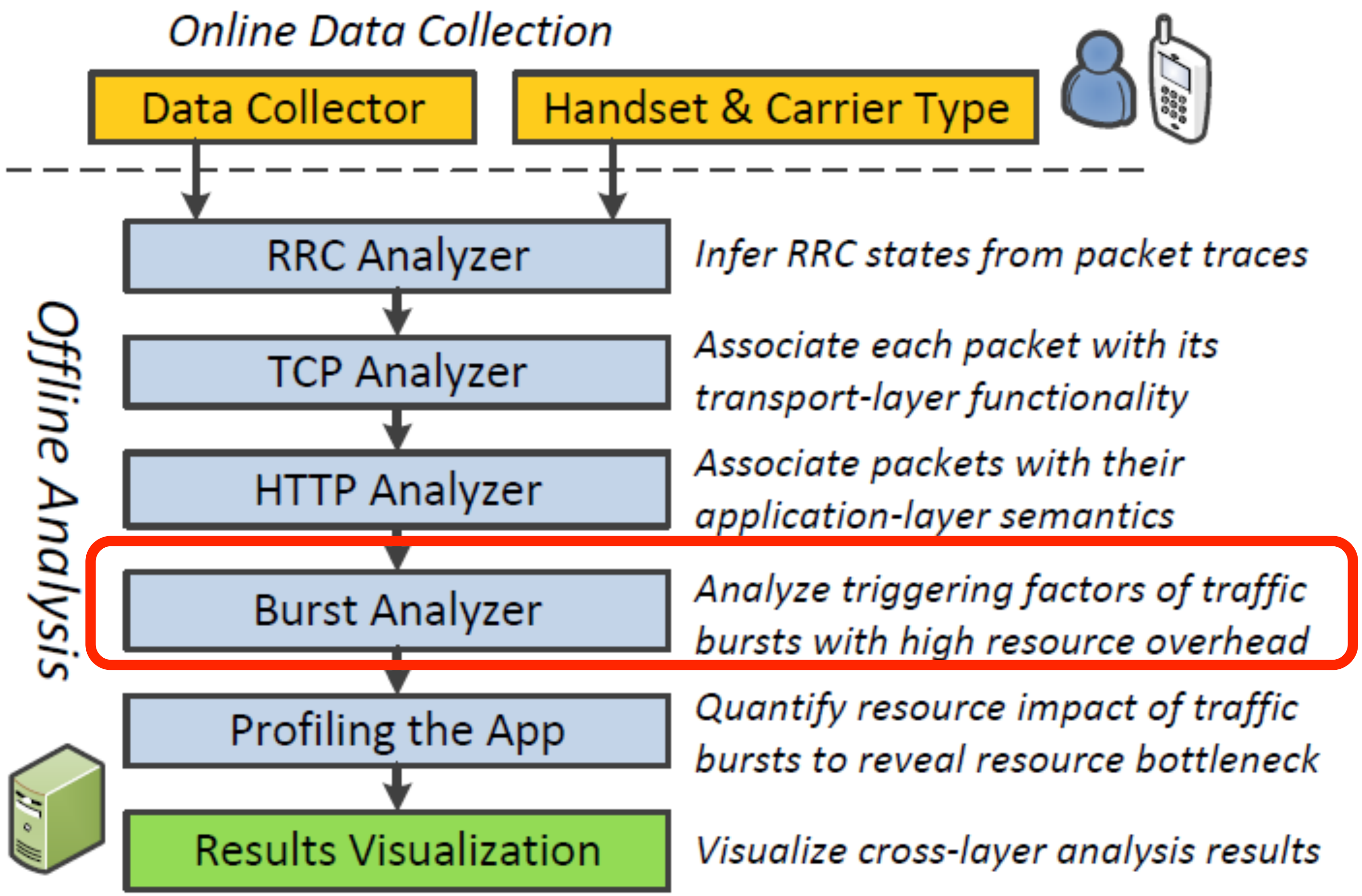
Courtesy: Feng Qian et al.

TCP / HTTP Analysis

- TCP Analysis
  - Infer transport-layer properties for each TCP packet
    - SYN, FIN, or RESET?
    - Related to loss? (e.g., duplicated ACK / recovery ACK)
    - …
- HTTP Analysis:
  - HTTP is the dominant app-layer protocol for mobile apps.
  - Model HTTP behaviors

Courtesy: Feng Qian et al.

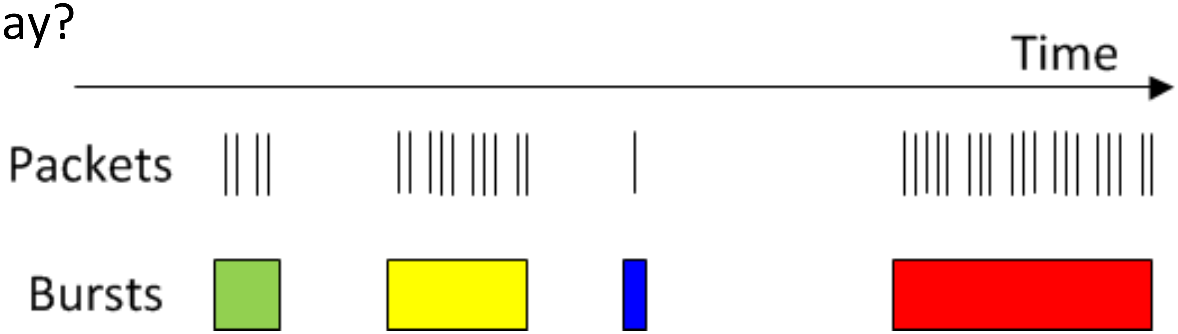
ARO System Architecture



Courtesy: Feng Qian et al.

Burst Analysis

- A burst consists of consecutive packets transferred in a batch (i.e., their IAT is less than a threshold)
- We are interested in **short bursts** that incur energy / radio resource inefficiencies
- ARO finds the **triggering factor** of each short burst
    - Triggered by user interaction?
    - By server / network delay?
    - By application delay?
    - By TCP protocol?



Courtesy: Feng Qian et al.

Burst Analysis Algorithm

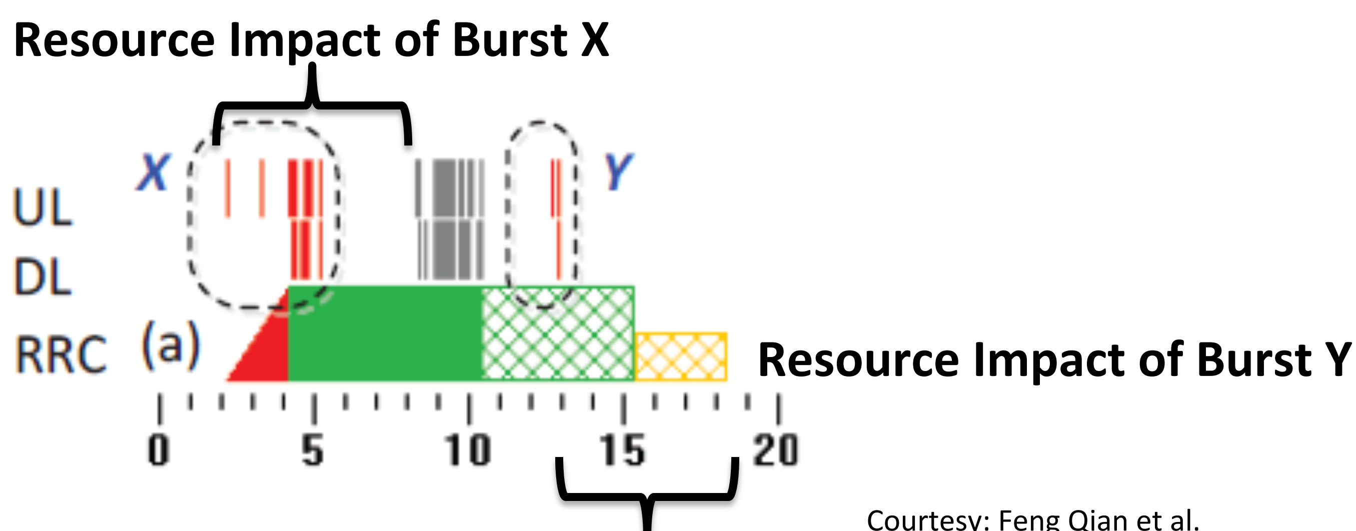
```
Burst_Analysis (Burst b) {
    Remove packets of non-target apps;
    if (no packet left) {return NON_TARGET;}                 Test 1
    if (b.payload > th_s && b.duration > th_d)               Test 2
      {return LARGE_BURST;}
    if (b.payload == 0) {                                    Test 3
      if (b contains any of ESTABLISH, CLOSE, RESET,
      TCP_OTHER packets)
        {return TCP_CONTROL;}
    }
    d0 ← direction of the first packet of b;
    i0 ← TCP label of the first packet of b;
    if (d0 == DL && (i0 == DATA || i0 == ACK))               Test 4
      {return SVR_NET_DELAY;}
    if (i0 == ACK_DUP && i0 == ACK_RECOVER &&
    i0 == DATA_DUP && i0 == DATA_RECOVER)                    Test 5
      {return TCP_LOSS_RECOVER;}
    if (b.payload > 0 && find user input before b)           Test 6
      {return USER_INPUT;}
    if (b.payload > 0) {return APP;}                         Test 7
    else {return UNKNOWN;}
}
```

Label	The burst is triggered by ...
USER_INPUT	User interaction
LARGE_BURST	(The large burst is resource efficient)
TCP_CONTROL	TCP control packets (*e.g.*, FIN and RST)
SVR_NET_DELAY	Server or network delay
TCP_LOSS_RECOVER	TCP congestion / loss control
NON_TARGET	Other applications not to be profiled
APP	The application itself
APP_PERIOD	Periodic data transfers (One special type of APP)

Courtesy: Feng Qian et al.

Compute Resource Consumption of a Burst

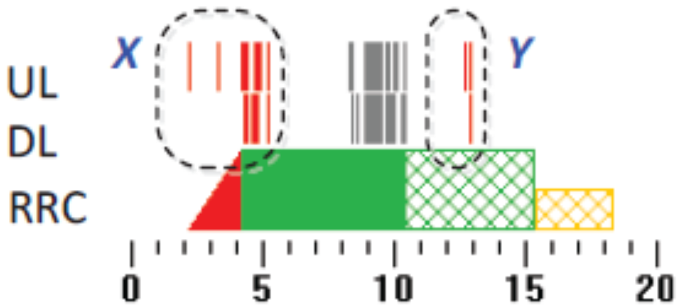
- **Upperbound** of resource utilization
  - The resource impact of a burst B_i is from the beginning of B_i to the beginning of the next burst B_{i+1}
  - May **overestimate** resource consumption, as one burst may already be covered by the tail of the previous burst



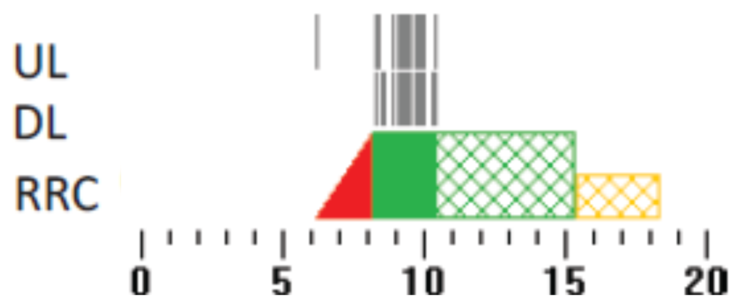
Courtesy: Feng Qian et al.

Compute Resource Consumption of a Burst

- **Lowerbound** of resource utilization
  - Compute the total resource utilization of the original trace
  - Remove the interested burst, then compute the resource utilization again
  - Take the delta



The original trace
Resource Utilization is E_1

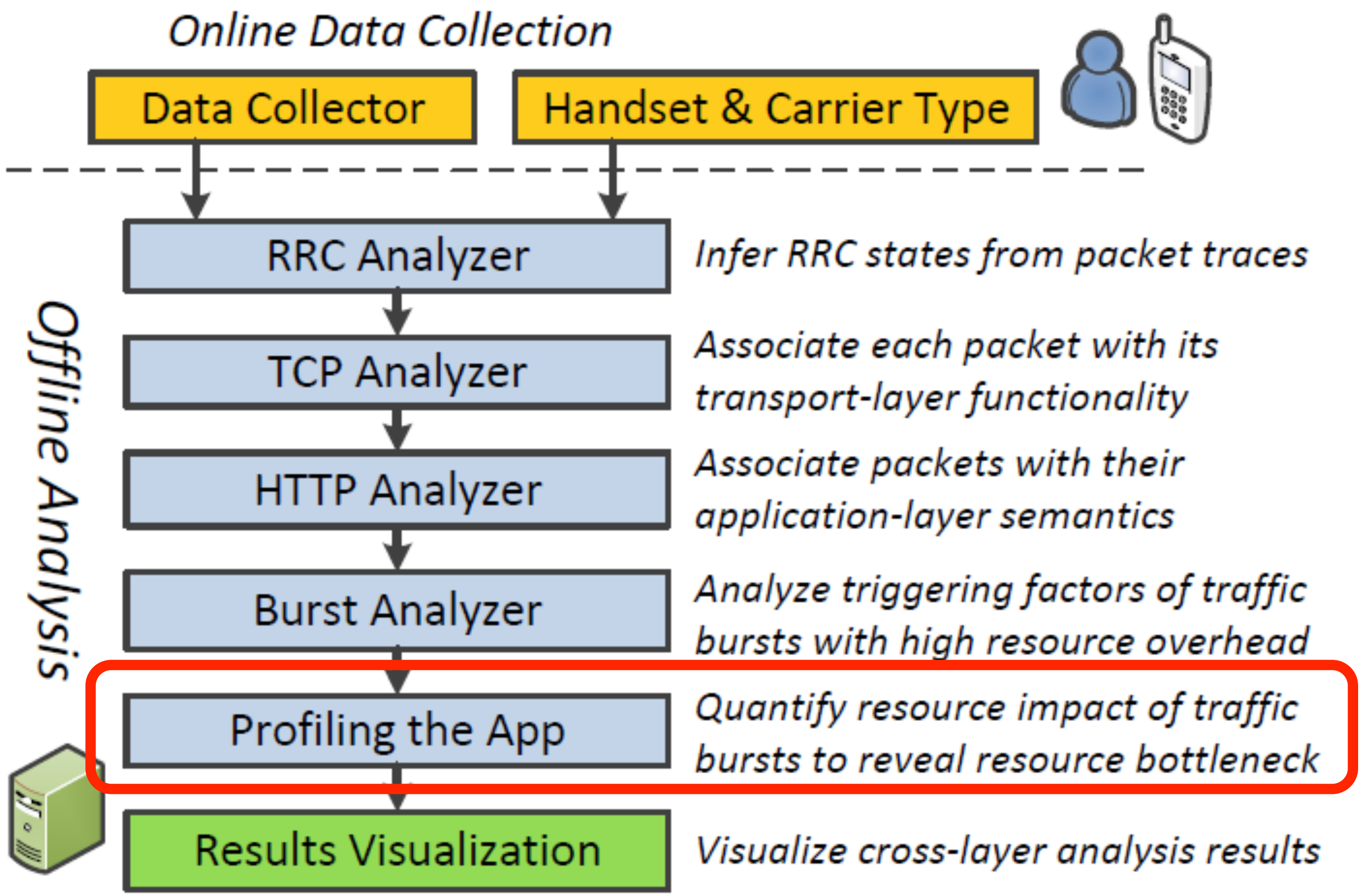


Remove X and Y
Resource utilization is E_2

The resource impact of X and Y is E_1-E_2

Courtesy: Feng Qian et al.

ARO System Architecture



Courtesy: Feng Qian et al.

Profiling Applications

- From **RRC Analysis**
  - We know the **radio resource state** and the **radio power** at any given time
- From **Burst analysis**
  - We know the **triggering factor** of each burst
  - We know the **transport-layer** and **application-layer** behavior of each burst
- By "**profiling applications**", we mean
  - Compute resource consumption of each **burst**
  - Therefore identify the root cause of resource inefficiency.

Courtesy: Feng Qian et al.

Metrics for Quantifying Resource Utilization Efficiency

- Handset radio energy consumption
- DCH occupation time
  - Quantifies **radio resource utilization**
- Total state promotion time (IDLE→DCH, FACH→DCH)
  - Quantifies **signaling overhead**
- Details of computing the three metrics (upperbound and lowerbound) in the paper

Courtesy: Feng Qian et al.

Implementation

- Data collector built on Android: modified tcpdump with two new features (1K lines of code)
  - logging user inputs: reads /dev/input/event*
    - captures all user input events such as touching the screen, pressing buttons
  - finding packet-to-application association
    - /proc/PID/fd containing mappings from process ID (PID) to inode of each TCP/UDP socket
    - /proc/net/tcp(udp) maintaining socket to inode mappings,
    - /proc/PID/cmdline that has the process name of each PID
- The analyzers were implemented in C++ on Windows 7 (7.5K lines of code)

Case Studies

- Fully implemented for Android platform (7K LoC)
- Study 17 popular Android applications
  - All in the "TOP Free" Section of Android Market
  - Each has 250,000+ downloads as of Dec 2010
- ARO pinpoints resource inefficiency for many popular applications. For example,
  - **Pandora Streaming**
    High radio energy overhead (50%) of periodic measurements
  - **Fox News**
    High radio energy overhead (15%) due to users' scrolling
  - **Google Search**
    High radio energy overhead (78%) due to real-time query suggestions

Courtesy: Feng Qian et al.

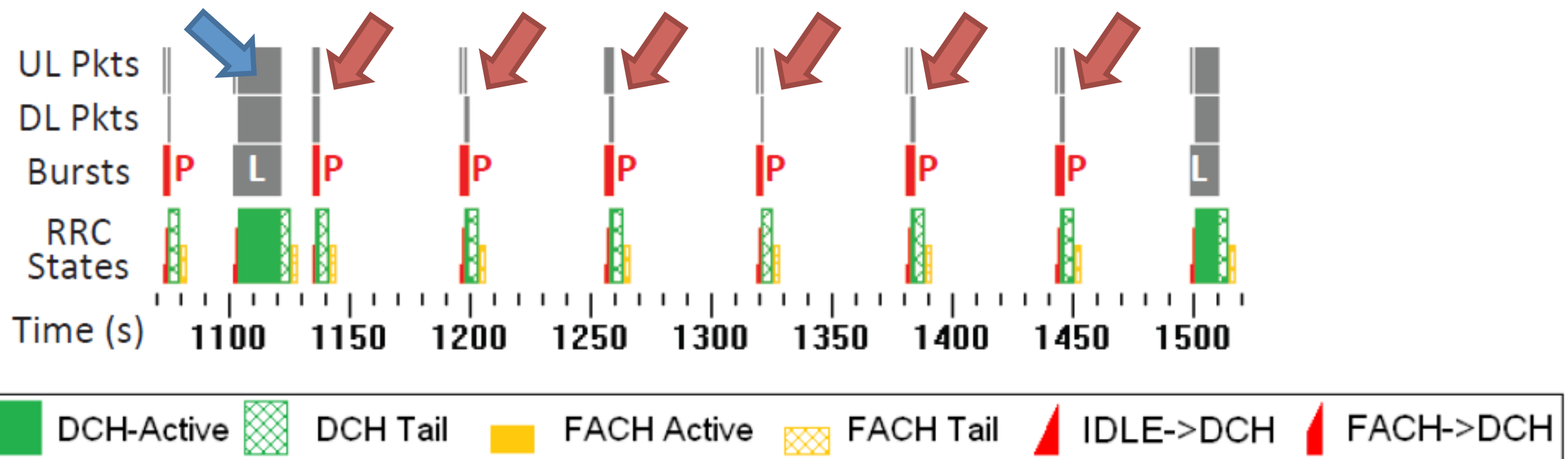
Case Study: Pandora Music

Pandora profiling results (Trace len: 1.45 hours)

Burst type	Payloads	Energy		DCH	
		LB	UB	LB	UB
LARGE_BURST	96.4%	35.6%	35.9%	42.4%	42.5%
APP_PERIOD	0.2%	45.9%	46.7%	40.4%	40.9%
APP	3.2%	12.8%	13.4%	12.4%	12.8%
TCP_CONTROL	0.0%	1.2%	1.6%	1.1%	1.5%
TCP_LOSS_RECOVER	0.2%	0.2%	0.6%	0.3%	0.7%
NON_TARGET	0.0%	1.8%	1.8%	1.7%	1.7%
Total	23.6 MB	846 J		895 sec	

Problem: High resource overhead of periodic audience measurements (every 1 min)

Recommendation: Delay transfers and batch them with delay-sensitive transfers



Courtesy: Feng Qian et al.

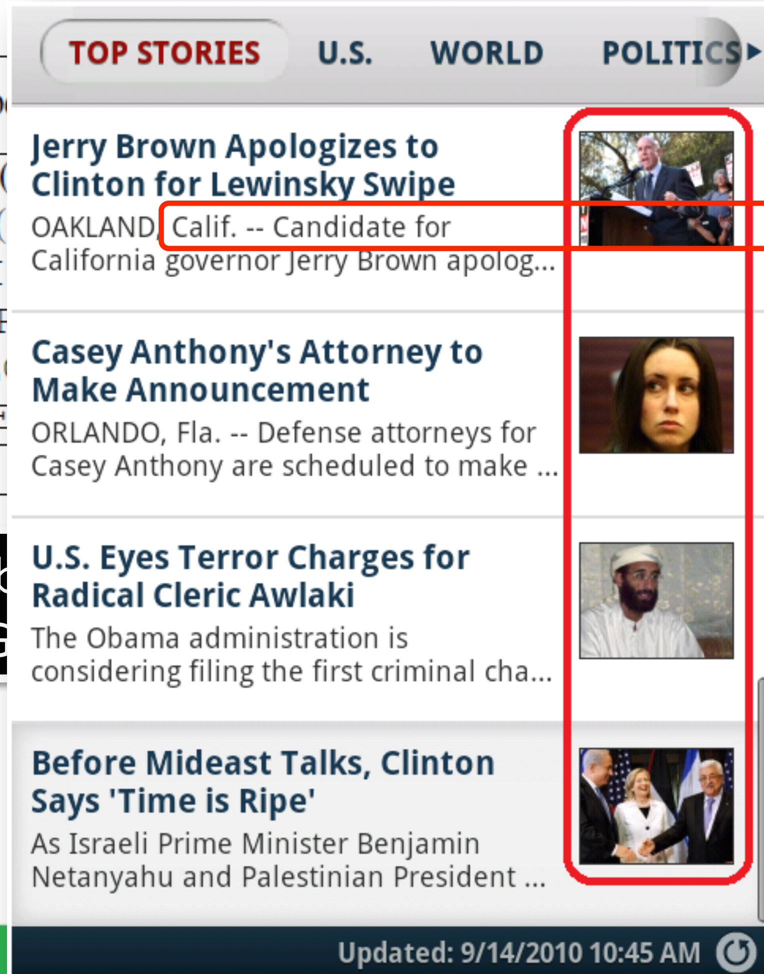
Case Study: Fox News

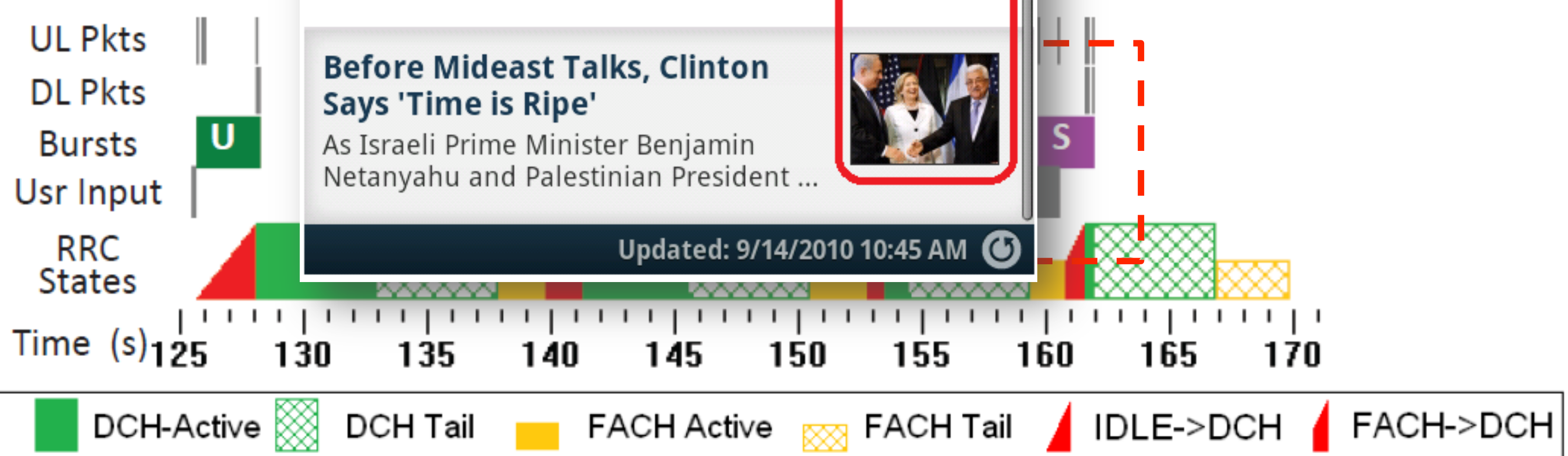
…nins)

Burst typ…	…	DCH	
		…B	UB
USER_INPUT(		…2%	70.4%
USER_INPUT(		…7%	16.7%
APP_PERI…		…%	7.4%
TCP_CONTR…		…%	2.3%
TCP_LOSS_RE…		…%	3.2%
SVR_NET_DE…		…%	0.0%
Total		284 sec	

Problem: Scattered b…

Recommendation: G… …mages in one burst





Courtesy: Feng Qian et al.

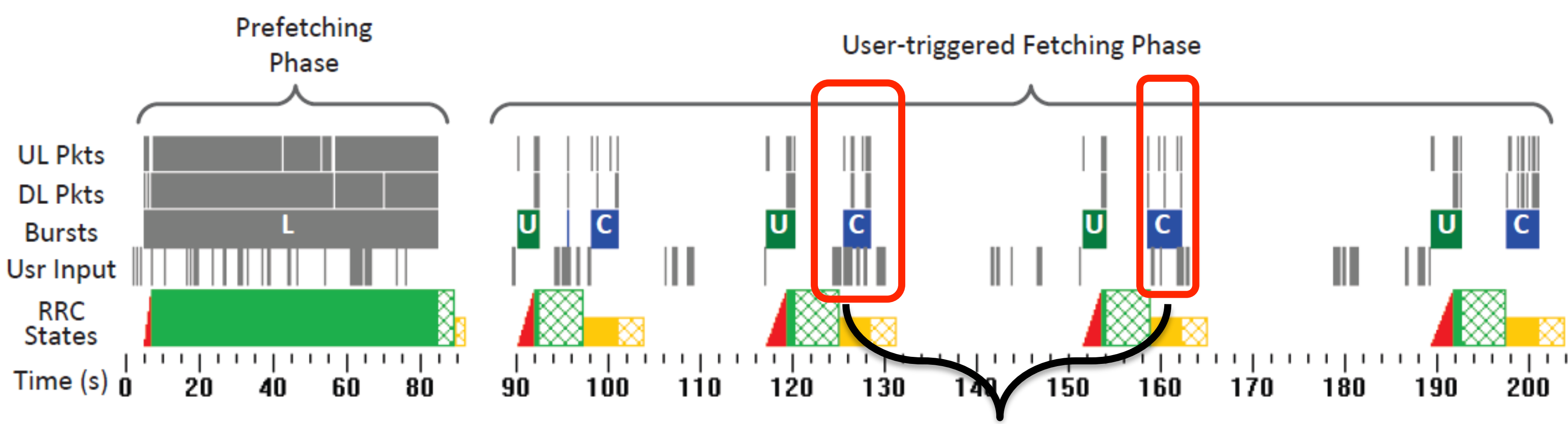
Case Study: BBC News

BBC News profiling results

User-triggered Fetching Phase (8 mins)					
Burst type	Payloads	Energy		DCH	
		LB	UB	LB	UB
TCP_CONTROL	0	11.3%	24.2%	0.0%	5.7%
USER_INPUT	98.7%	42.5%	73.1%	37.9%	90.0%
SVR_NET_DELAY	1%	0.0%	2.7%	0.0%	5.2%
Total	162 KB	145 J		120 sec	

Problem: Scattered bursts of delayed FIN/RST packets

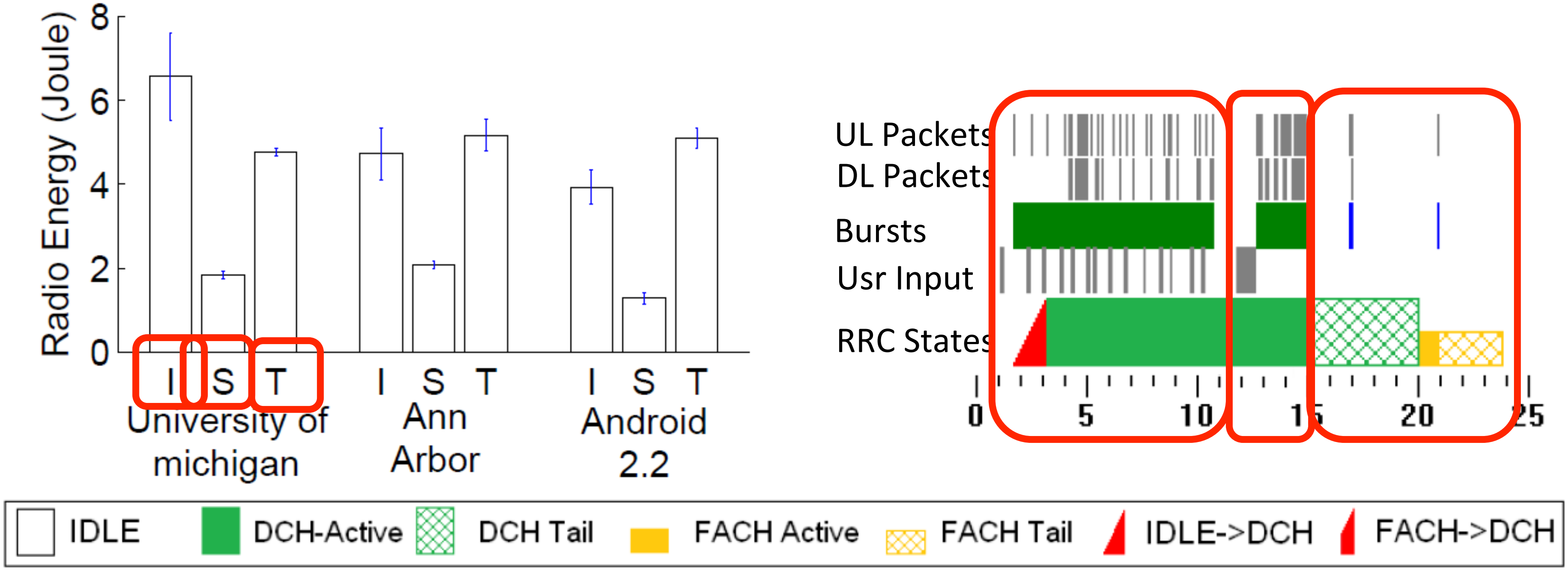
Recommendation: Close a connection immediately if possible, or within tail time



Scattered bursts of delayed FIN/RST Packets

Courtesy: Feng Qian et al.

Case Study: Google Search



Search three key words.

ARO computes energy consumption for three phases

I: Input phase S: Search phase T: Tail Phase

Problem: High resource overhead of query suggestions and instant search

Recommendation: Balance between functionality and resource when battery is low

Courtesy: Feng Qian et al.

Case Study: Audio Streaming

Constant bitrate vs. bursty streaming

Name	Server	bitrate	Radio Power
NPR News	SHOUTcast	32 kbps	36 J/min
Tune-in	Icecast	119 kbps	36 J/min
Iheartradio	QTSS	32 kbps	36 J/min
Pandora w/ Ad	Apache	bursty	11.2 J/min
Pandora w/o Ad*	Apache	bursty	4.8 J/min
Slacker	Apache	bursty	10.9 J/min

* A hypothetical case where all periodic ads are removed.

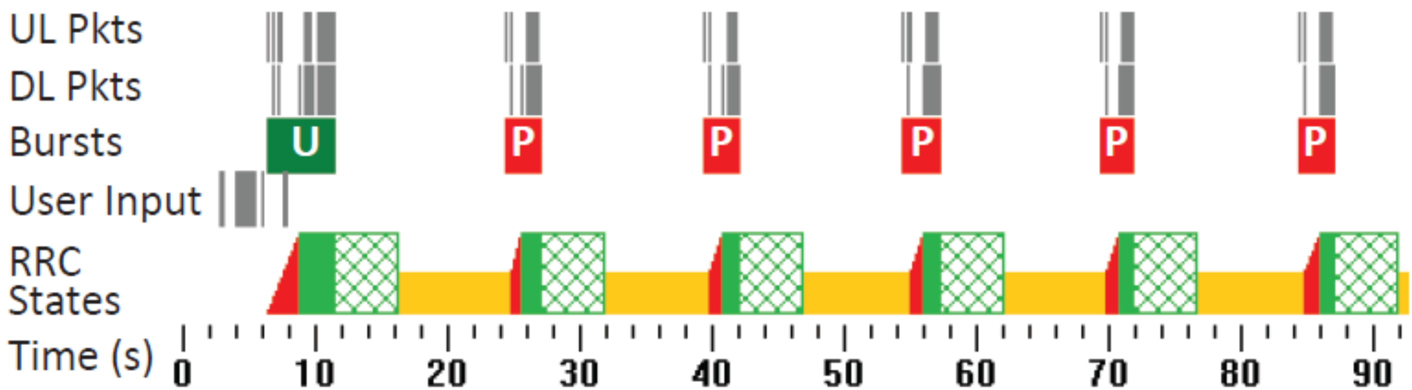
Problem: Low DCH utilization due to constant-bitrate streaming
Recommendation: Buffer data and periodically stream data in one burst

Courtesy: Feng Qian et al.

Case Study: Mobile Advertisements

Comparing three mobile ad platforms

Name	Default Refresh Rate	Avg Update Size	Radio Power	
			w/ FD	w/o FD
Google Mobile Ad	180.0 sec	6.0 KB	2.5 J/min	3.6 J/min
AdMob	62.5 sec	6.8 KB	5.7 J/min	8.8 J/min
Mobclix	15.0 sec	1.4 KB	23.2 J/min	29.6 J/min



Problem: Aggressive ad refresh rate making the handset persistently occupy FACH or DCH

Recommendation: Decrease the refresh rate, piggyback or batch ad updates

Courtesy: Feng Qian et al.

Outline

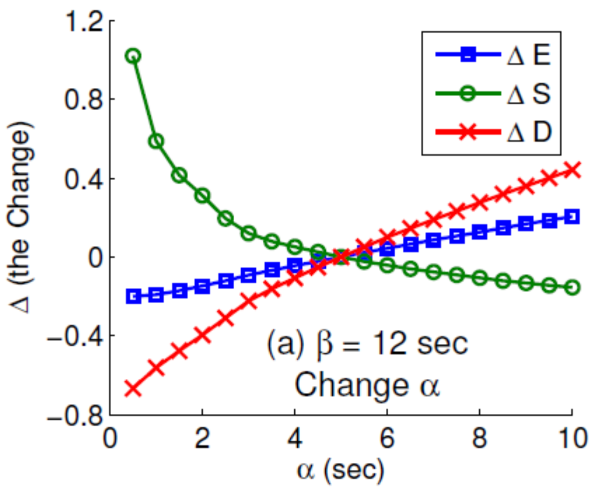
- Introduction
- RRC State Inference
- Radio Resource Usage Profiling & Optimization
- Network RRC Parameters Optimization
- Conclusion

What-if Analysis for Inactivity Timers

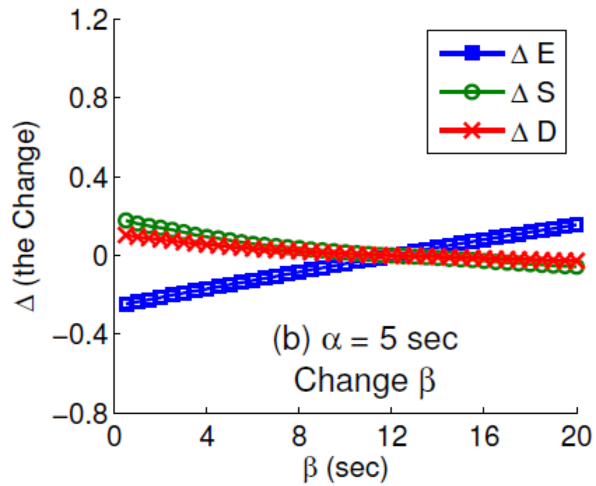
- Inactivity timers are the most crucial parameters affecting
  - UE energy consumption
  - State promotion overhead
  - Radio resource utilization (i.e., DCH occupation time)
- What is the impact of changing inactivity timers
  - Perform **what-if analysis** by replaying traces to the simulator with different inactivity timer values.

Courtesy: Feng Qian et al.

What-if Analysis for Inactivity Timers (Cont'd)



Fix the α (DCH→FACH) timer
Change the β(FACH→IDLE) timer



Fix the β(FACH→IDLE) timer
Change the α (DCH→FACH) timer

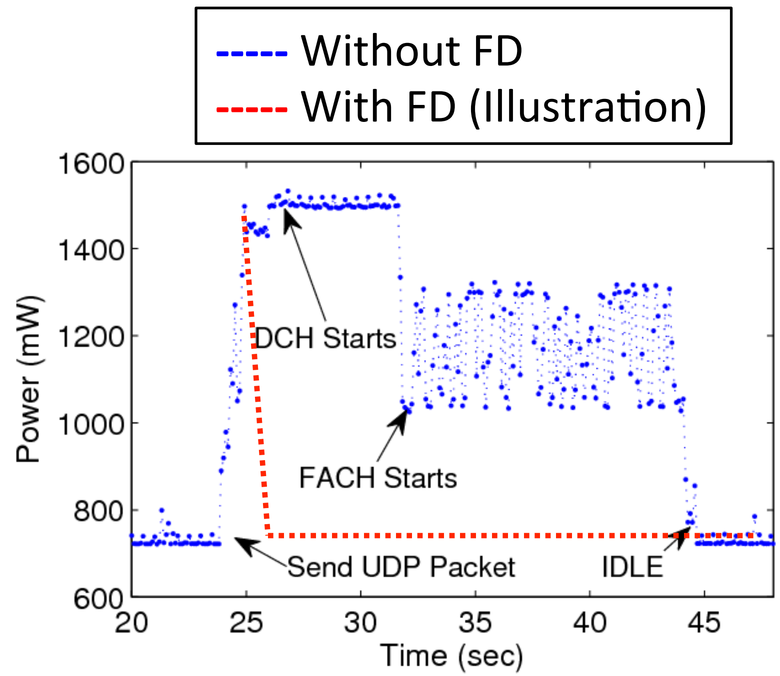
	Relative Change of…
ΔE	Radio Energy
Δ S	Promotion Delay
Δ D	DCH Occupation Time

- The α (DCH→FACH) timer imposes much higher impact on the three metrics than the β(FACH→IDLE) timer does.
- Very small α timer values (< 2 sec) cause significant increase of state promotion overhead.
- **It is difficult to well balance the tradeoff. The fundamental reason is that timers are globally and statically set to constant values.**

Courtesy: Feng Qian et al.

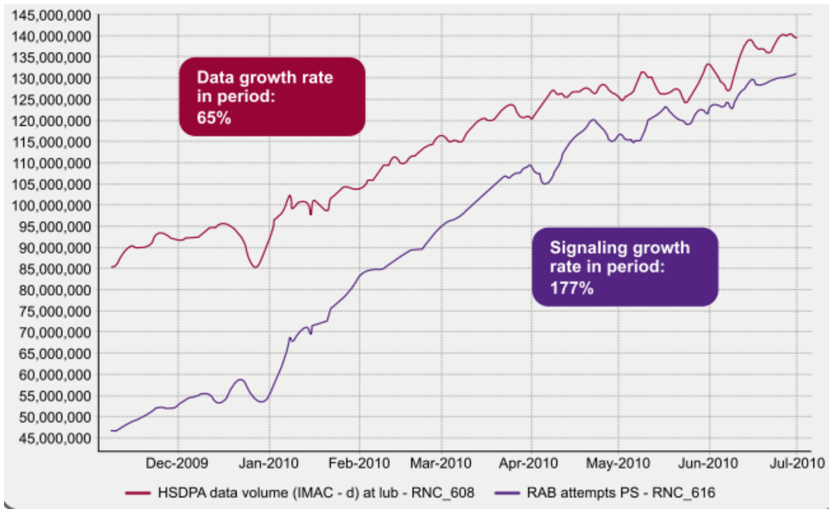
Fast Dormancy

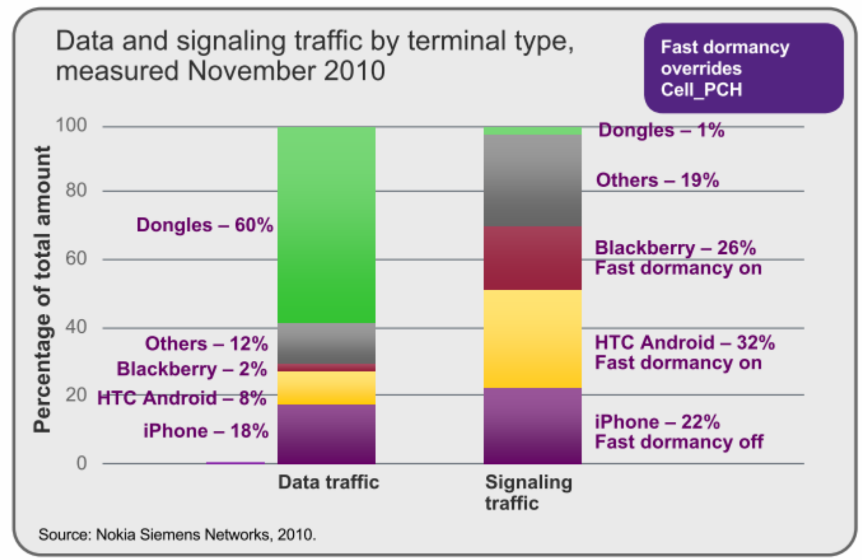
- A new feature added in 3GPP Release 7
- When finishing transferring the data, a handset sends a **special RRC message** to RAN
- The RAN immediately releases the RRC connection and lets the handset go to IDLE
- Fast Dormancy dramatically reduces the tail time, saving radio resources and battery life
- Fast Dormancy has been supported in some devices (e.g., Google Nexus One) in **application-agnostic** manner



Courtesy: Feng Qian et al.

Fast Dormancy Woes



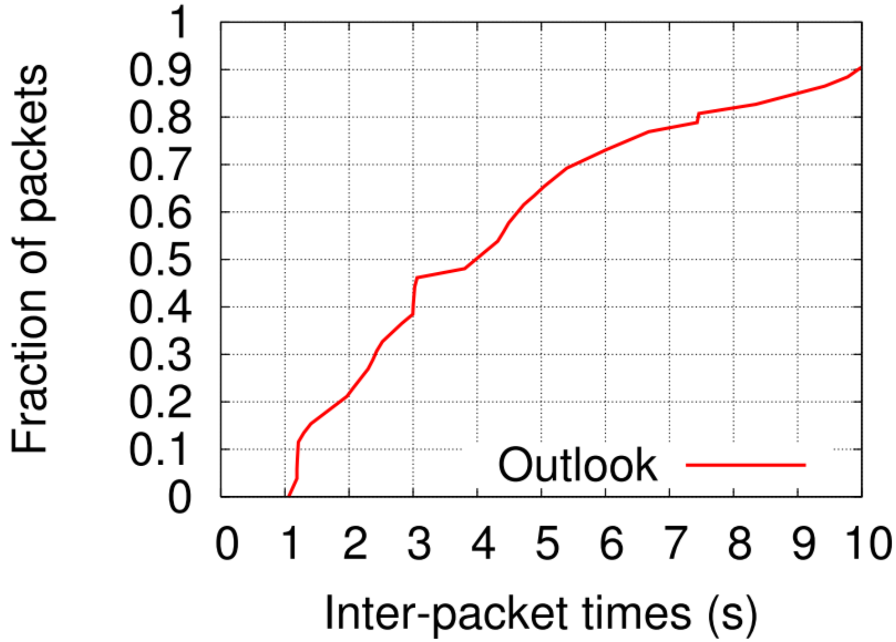


Disproportionate increase in signaling traffic caused due to increase in use of fast-dormancy

"Apple upset several operators last year when it implemented firmware 3.0 on the iPhone with a fast dormancy feature that prematurely requested a network release only to follow on with a request to connect back to the network or by a request to re-establish a connection with the network …"
[What's really causing the capacity crunch? - FierceWireless]

Courtesy: Vishnu Navda et al.

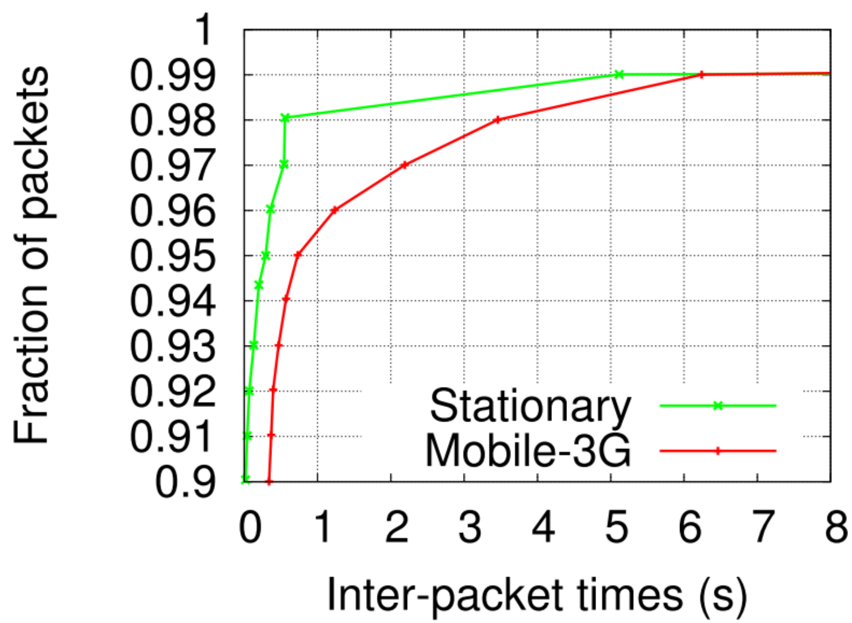
Problem #1: Chatty Background Apps



- No distinctive knee
- High mispredictions for fixed inactivity timer

Courtesy: Vishnu Navda et al.

Problem #2: Varying Network Conditions



- Signal quality variations and handoffs cause sudden latency spikes
- Aggressive timers frequently misfire

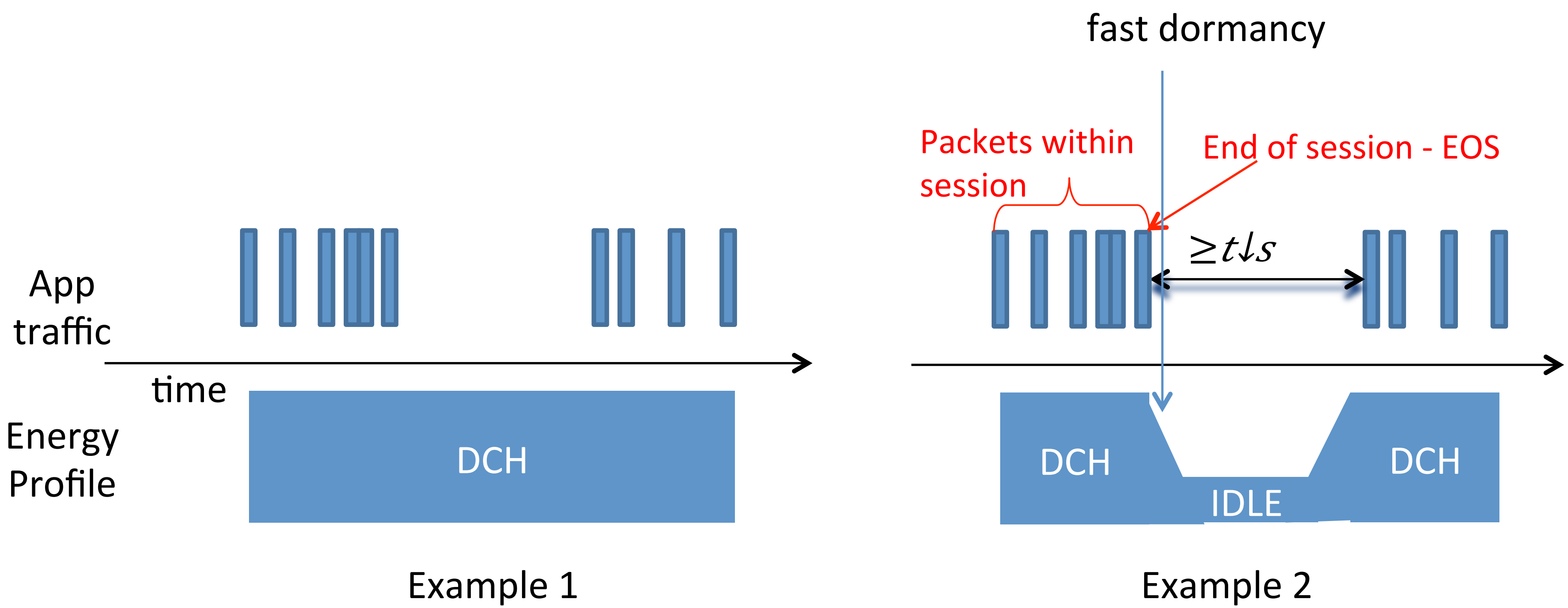
Courtesy: Vishnu Navda et al.

Objectives

- Design a fast-dormancy policy for long-standing background apps which
  - Achieves energy savings
  - Without increasing signaling overhead
  - Without requiring app modifications

Courtesy: Vishnu Navda et al.

When to Invoke Fast Dormancy?



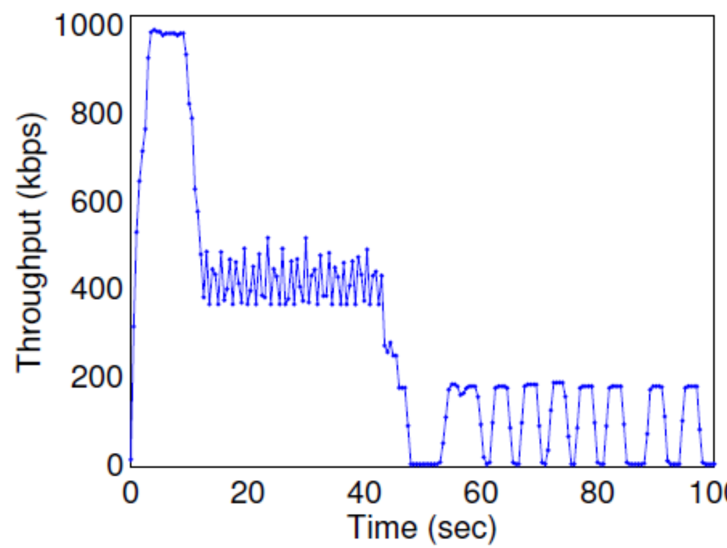
Energy savings when $t_s \geq 3\ sec$ and fast dormancy is invoked immediately after end of session

Courtesy: Vishnu Navda et al.

Use Fast Dormancy to Enhance Chunk Mode

Examle: YouTube

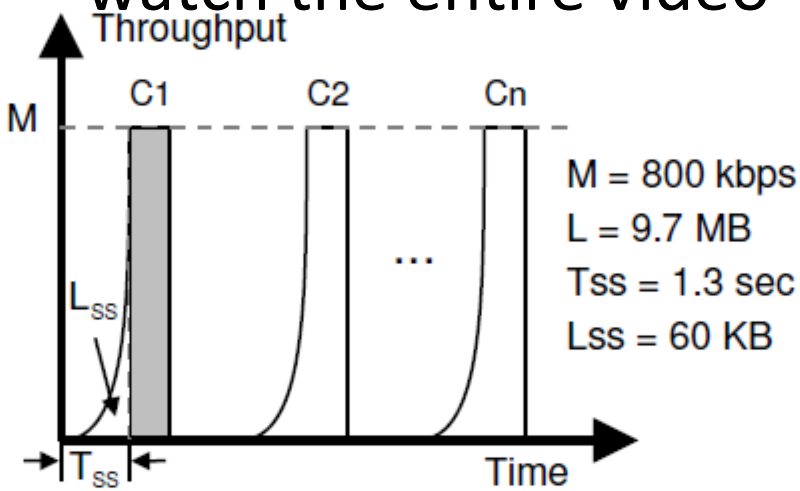
- YouTube video streaming
  - Collect a 10-min YouTube trace using Android G2 of Carrier 2.
  - Traffic pattern
    First 10 sec: maximal bw is utilized
    Next 30 sec: constant bitrate of 400kbps
    Remaining: transmit intermittently with
    the inter-burst time between 3~5 s.
  - **Under-utilization of network bandwidth** causes its long DCH occupation time.
    - Energy/radio resource efficiency is much **worse** than Pandora



Courtesy: Feng Qian et al.

Use Fast Dormancy to Enhance Chunk Mode (Cont'd)

- Proposed traffic pattern: **Chunk Mode**
  - The video content is split into *n* chunks $C_1, \ldots, C_n$
  - Each transmitted at the highest bit rate.
  - *n* should not be too small as users often do not watch the entire video



Para	Meaning
M	Maximal BW
L	Content size
T_{SS}	Slow start duration
L_{SS}	Bytes transferred in slow start

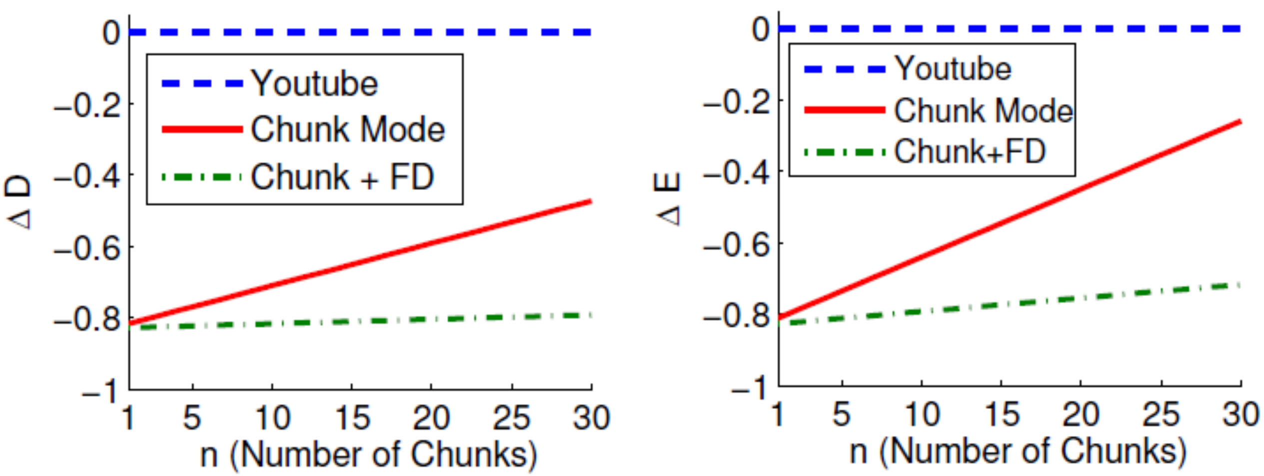
How to eliminate the Tail for each chunk?

Using Fast Dormancy

Courtesy: Feng Qian et al.

Use Fast Dormancy to Enhance Chunk Mode (Cont'd)

- Invoke fast dormancy at the end of each chunk
  - To immediately release radio resources (assuming
    ists)



	Relative Change of…
ΔE	Radio Energy
Δ D	DCH Occupation Time

Chunk Mode: Save 80% of DCH occupation time and radio energy for YouTube

Fast Dormancy: Keep ΔD and ΔE almost constant regardless of # of chunks.

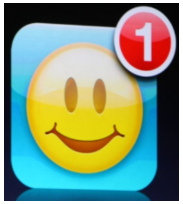
Courtesy: Feng Qian et al.

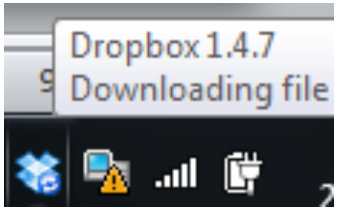
Problem: predict end of session (or onset of network inactivity)

Idea: exploit unique application characteristics (if any) at end of sessions

Typical operations performed:

- UI element update
- Memory allocation or cleanup
- Processing received data

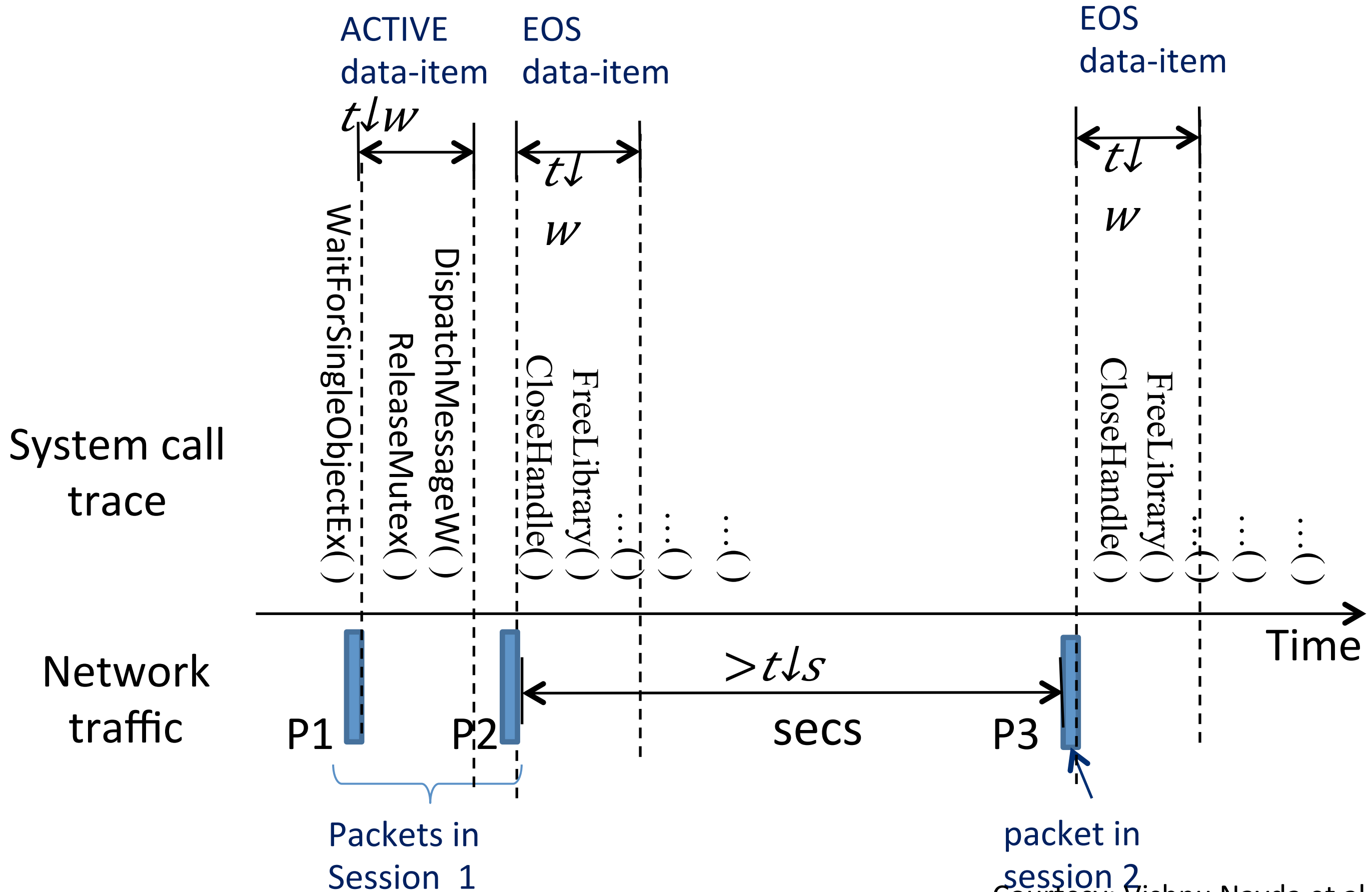




System calls invoked by an app can provide insights into the operations being performed

Predicting onset of network inactivity

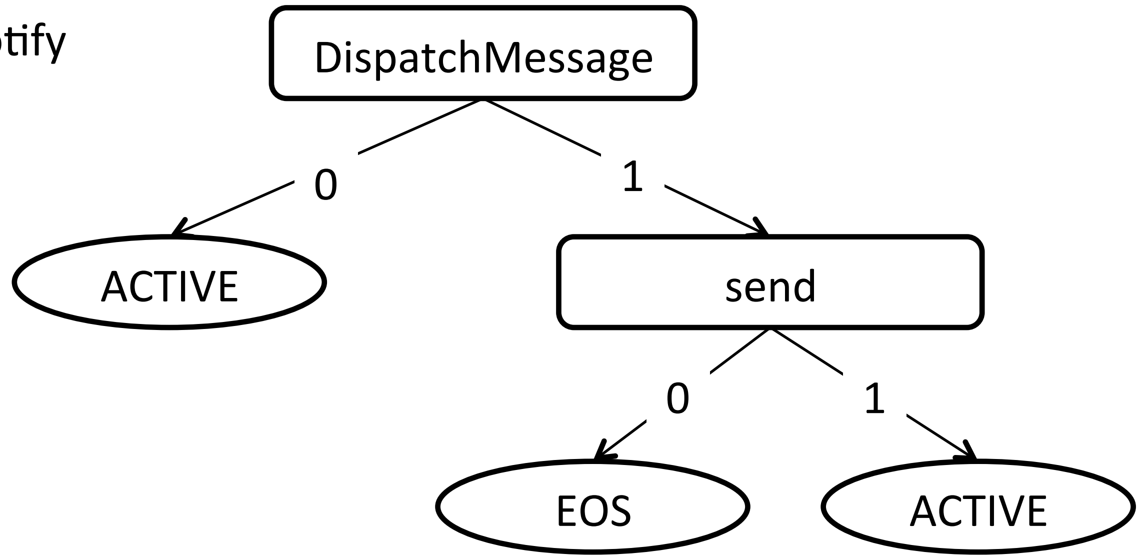
- Technique: Supervised learning using C5.0 decision trees
- Data item: system calls observed immediately after a packet (encoded as bit-vector)
- Label: ACTIVE or EOS



Courtesy: Vishnu Navda et al.

Decision tree example

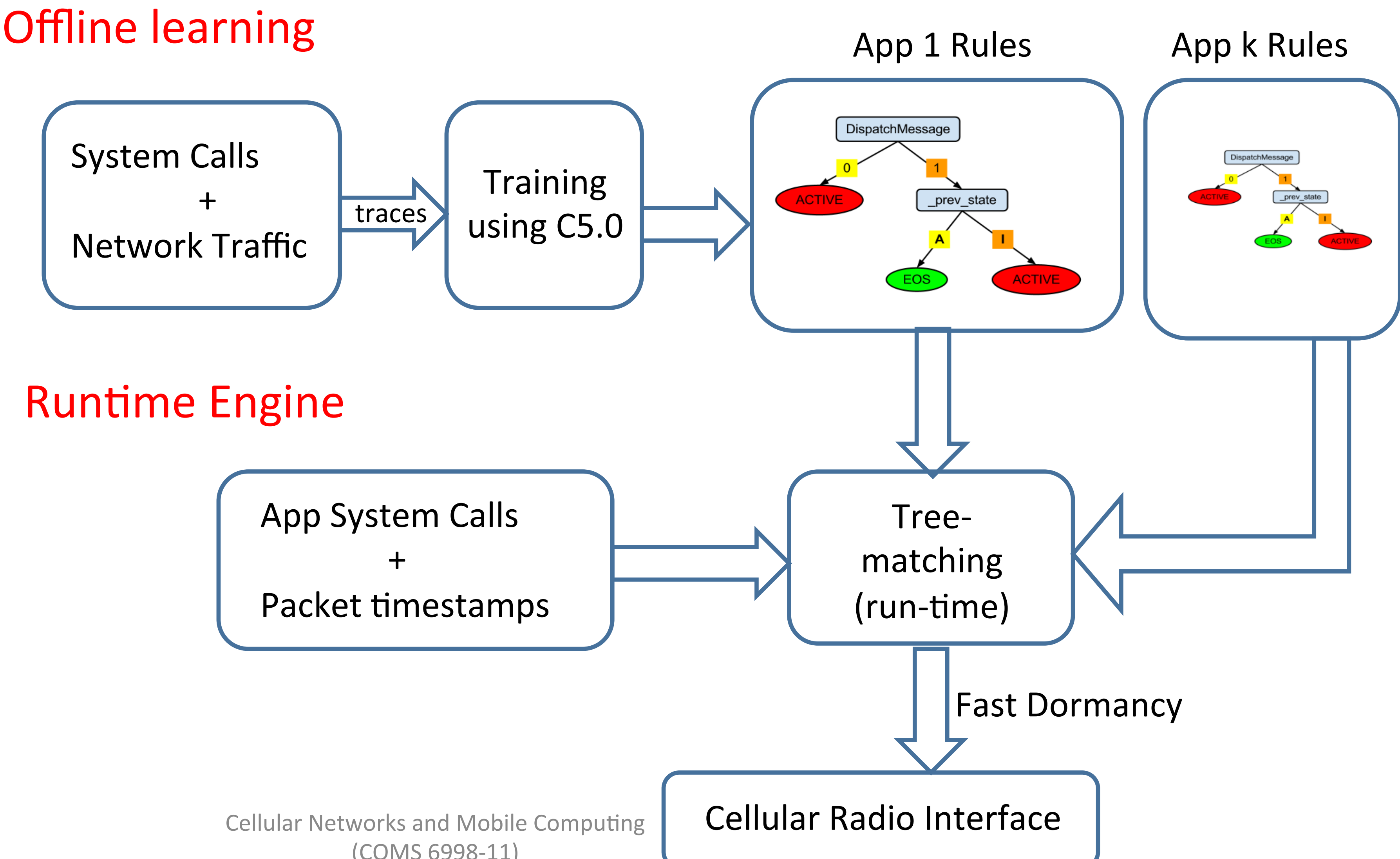
Application: gnotify



Rules:
(DispatchMessage & ! send) => EOS
! DispathcMessage => ACTIVE
(DispatchMessage & send) => ACTIVE

Courtesy: Vishnu Navda et al.

RadioJockey System



Courtesy: Vishnu Navda et al.

Evaluation

1. Trace driven simulations on traces from 14 applications (Windows and Android platform) on 3G network
   - Feature set evaluation for training
   - variable workloads and network characteristics
   - 20-40% energy savings and 1-4% increase in signaling over *3 sec idle timer*

2. Runtime evaluation on 3 concurrent background applications on Windows

Runtime Evaluation with Concurrent Background Applications

Applications	Energy Savings (%)	Signaling Overhead (%)
Outlook	24.03	4.47
GTalk	24.07	4.57
Lync	24.14	0
All	22.8	6.96

- 22-24% energy savings at a cost of 4-7 % signaling overhead
- Marginal increase in signaling due to variance in packet timestamps

Conclusion

- ARO helps developers design **cellular-friendly** smartphone applications by providing **visibility** of radio resource and energy utilization.
- Cellular friendly techniques (http://developer.att.com/home/develop/referencesandtutorials/networkapibestpractices/Top_Radio_Resource_Issues_in_Mobile_Application_Development.pdf)
  - Group multiple simultaneous connections from the same server
  - Batching and piggybacking
  - Close unnecessary TCP connections early
  - Offloading to WiFi when possible (ms setup rather than 2sec)
  - Caching and avoid duplicate content
  - Prefetching intelligently
  - Access peripherals judicially
- Try out the ARO tool at:
  - http://developer.att.com/developer/forward.jsp?passedItemId=9700312

Questions?