Energy Consumption in Mobile Phones

Motivation

- Network applications increasingly popular in mobile phones
  - 3G/4G/5G/6G enabled, WiFi enabled

- Network applications cause huge power drain and can considerably reduce battery life

How can we reduce network energy cost in phones?

Contributions

1. Compare the energy consumption characteristics of 3G, GSM and WiFi and measure the fraction of energy consumed for data transfer versus overhead.

2. Analyze the variation of the energy overhead with geographic location, time-of-day, mobility, and devices. Develop a simple energy model to quantify the energy consumption over 3G, WiFi and GSM as a function of the transfer size and the inter-transfer times.

3. Design TailEnder protocol
   - Energy reduced by 40% for common applications including email and web search

Outline

- **Measurement study (cellular and Wifi)**
- TailEnder Design
- Evaluation

3G Power consumption

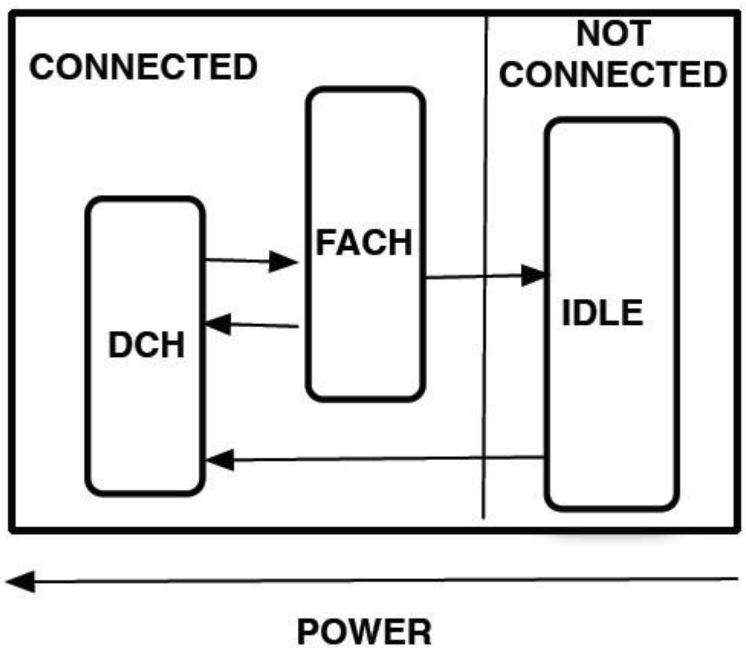
Two factors determine the energy consumption due to network activity in a cellular device.

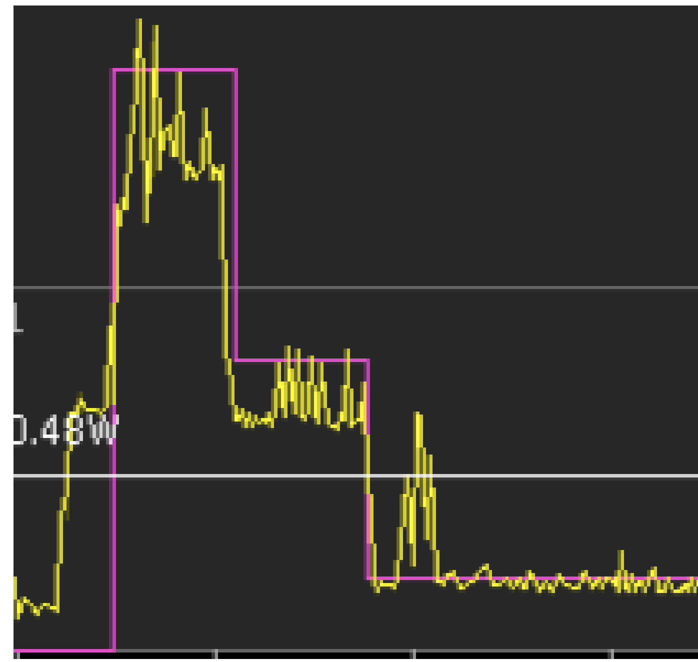
(a) First, is the transmission energy that is proportional to the **length of a transmission** and **the transmit power level.**

(b) Second, is the Radio Resource Control (RRC) protocol that is responsible for channel allocation and scaling the power consumed by the radio based on inactivity timers.

Power measurement tool

- Nokia energy profiler software (NEP)
  - State machine of RRC protocol





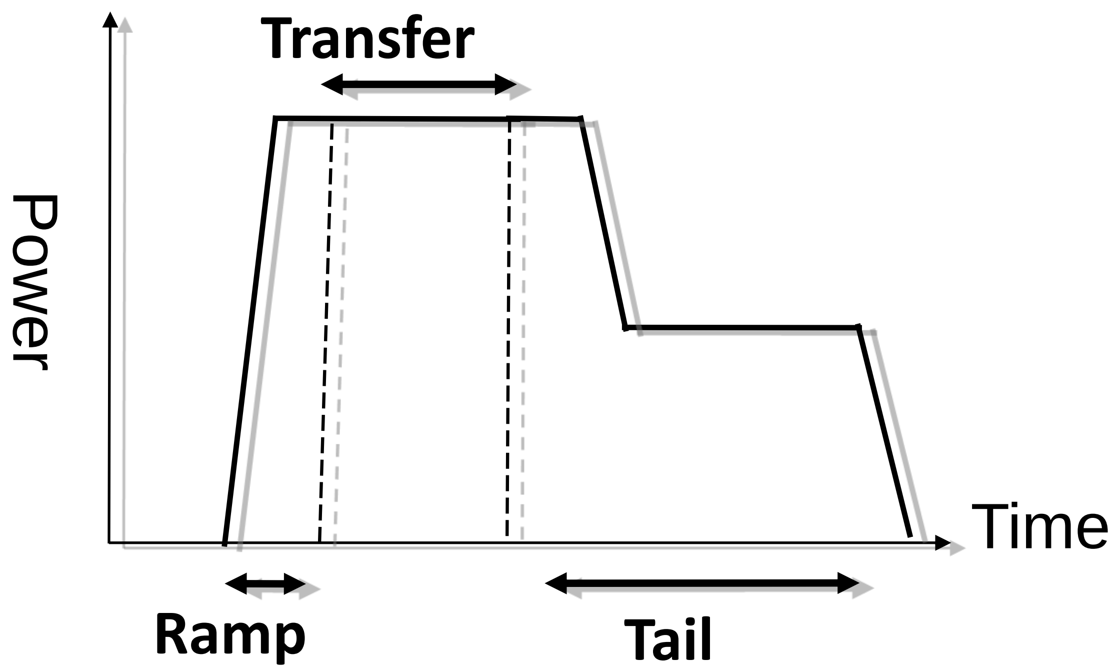
Power profile of an example network transfers

DCH – (Dedicated Channel) – reserves a dedicated channel

FACH (Forward Access Channel) – shares channel with other devices. – little traffic

3GPower consumption

Power profile of a device corresponding to network activity



3G power consumption

- **Ramp energy:** To create a dedicated channel and high throughput/low latency (DCH)
- **Transfer energy:** For data transmission
- **Tail energy**: To reduce signaling overhead and latency
  - Tail time is a trade-off between energy and latency

The tail time is set by the operator to reduce latency. Devices do not have control over it.

Measurement goals

- What fraction of energy is consumed for data transmission versus overhead?
- How does energy consumption vary with application workloads for cellular and WiFi technologies?

Measurement set up----3G/GSM

- **Devices**: 4 Nokia N95 phones
  - Enabled with AT&T 3G, GSM EDGE (2.5G) and 802.11b
- **Measurement Quantifies:** Ramp energy, transmission energy, tail energy
- **Environment:**
  - 4 cities, static/mobile, varying time of day

- Configure the phone in lowest power mode, turn off display and unused network interfaces
- Idle power is < 0.05W
- NEP running at sampling frequency of 0.25sec increases the power to 0.1W

Measurement set up----3G/GSM

- **Experiments:** Upload/Download data
  - Varying sizes ....x (1 to 1000KB)
  - Varying time intervals ....t (1 to 20 second) between successive transfers

  Measure energy consumption by running NEP in the background

  The phone initiates x KB download by issuing http req. to remote server…..then phone waits for t second, and then issues next http req.

  Repeat this process for 20 times each data size
- **Environment:**
  - 4 cities, static/mobile, varying time of day

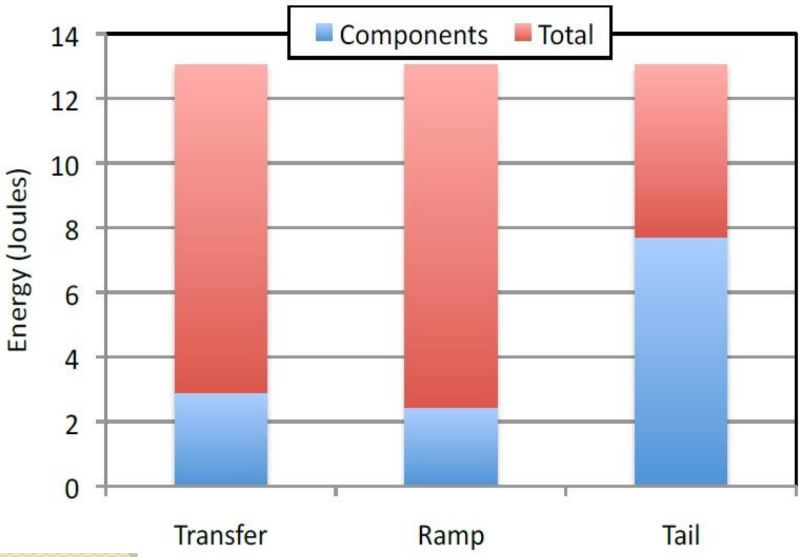
WiFi Power consumption

- Network power consumption due to
  - Scan/Association to AP
  - Transfer

Two experiments

(a) For each data transfer, first scan for Wifi AP, associate and make transfer

(b) Make one scan and association for entire data transfer



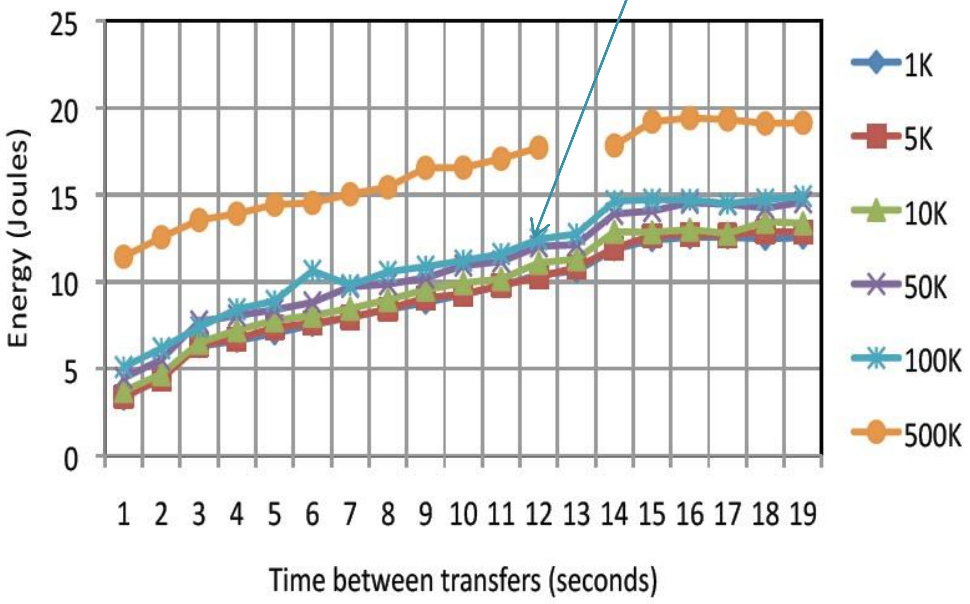
i. Tail energy is >60% of total
ii. Ramp only 14%

Consider transfer of 100KB data

Energy increases from 5J to 13J---
1 sec to 12 sec.

Time>12.5 sec, plateau at 15J

3G Measurements: (a) Ramp, transfer and tail energy for 50K 3G download. (b) Average energy consumed for transfer against the time between successive transfers



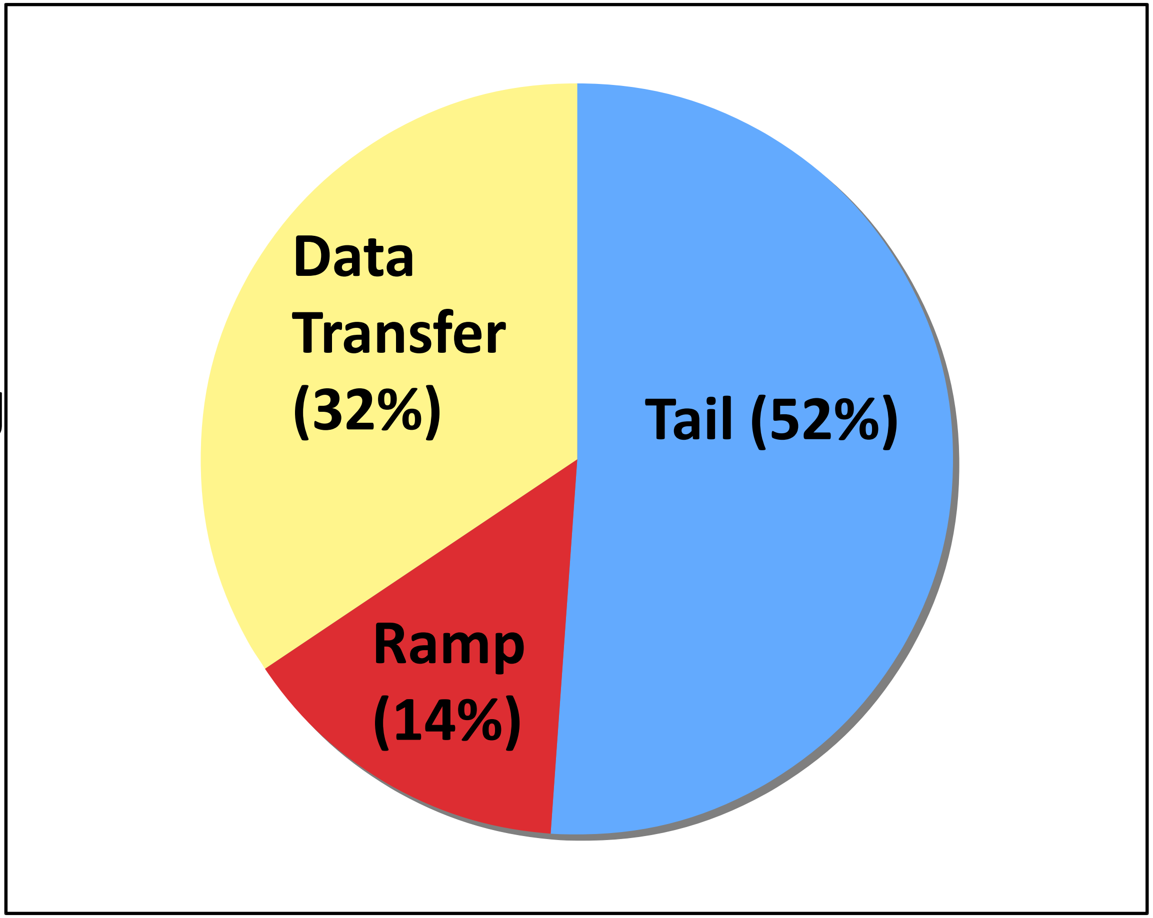
3G results

3G Energy Distribution for a 100K download

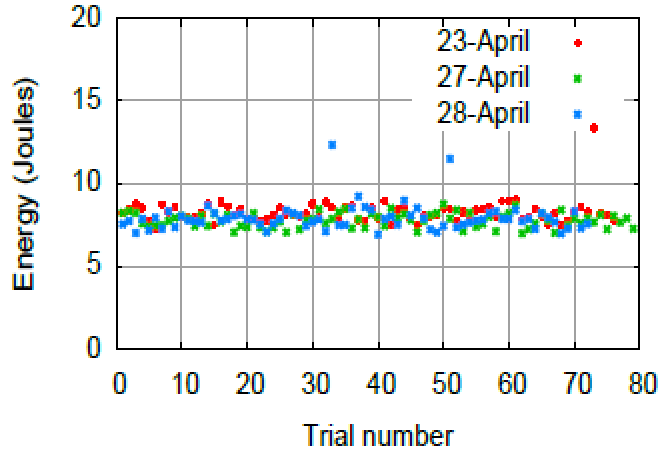
Total energy= 14.8J

Tail time = 13s

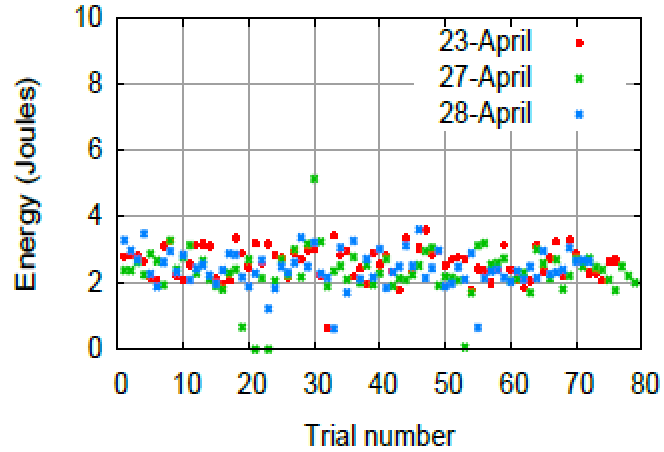
Tail energy = 7.3J



Measurement set up----3G
Geographical and temporal variation

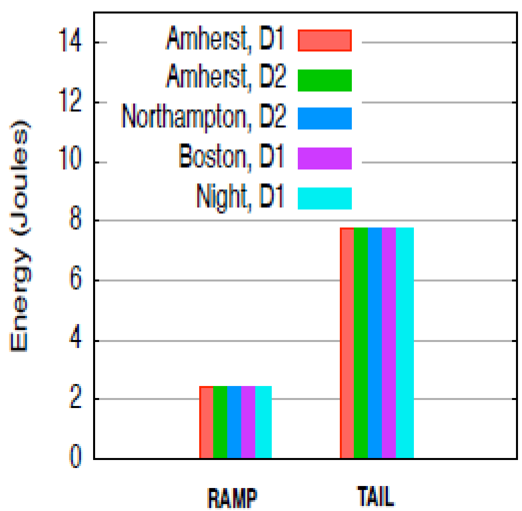


(a) Temporal variation: Tail energy



(b) Temporal variation: Ramp energy

Day wise tail and ramp time

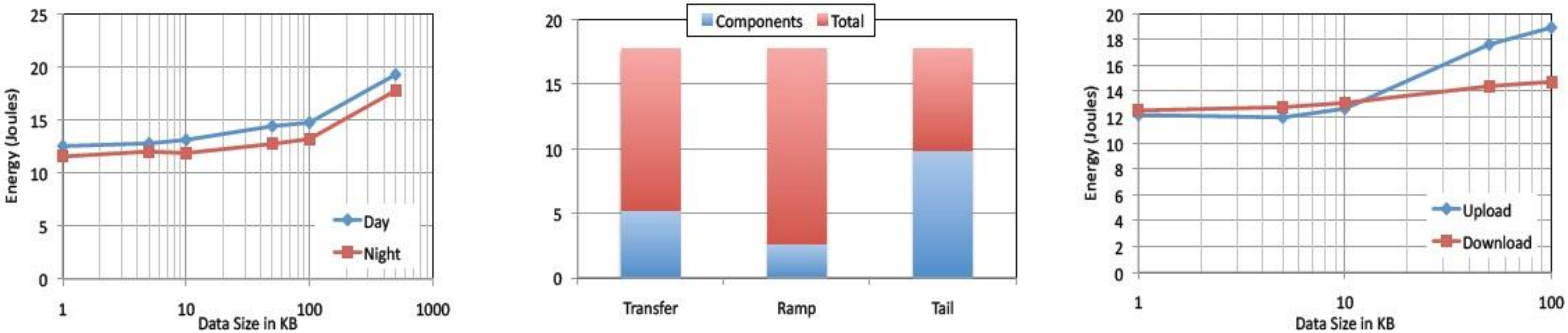


(c) Geographical variation

(a) impact of different cell towers on tail time
(b) consistency of tail time

Tail energy is consistent across different locations and for two devices

Tail time is statically configured by the network operator



(a) Download: Day versus Night

(b) Upload measurements

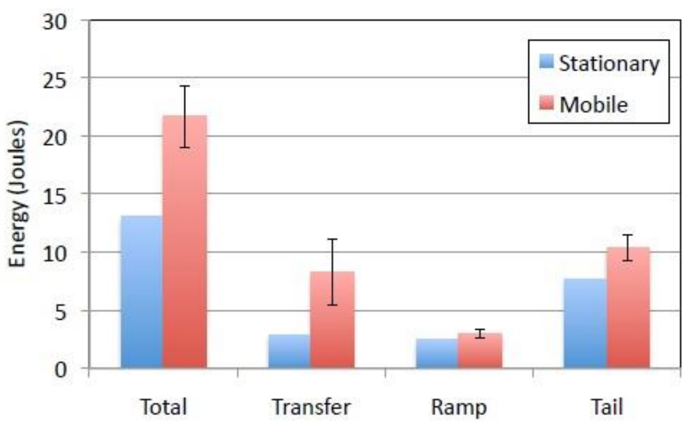
(c) Upload versus Download

Lower congestion during night

Tail energy costs >55% of total

Transfer energy for upload > download

Figure 4: : 3G: (a) Comparison of energy consumption of day versus night time transfers. x-axis is in logarithmic scale. (b) Transfer, Tail and Ramp energy for upload experiments (c) Comparison of energy consumption of download versus upload experiments.



Mobility in outdoor settings affect transfer rate due to factors such as signal strength, hand-off between cell towers ---- varying transfer time

Tail energy costs significantly (50%)

Figure 5: 3G Mobility Measurements: Energy components of 50K 3G transfers with confidence intervals. (Averaged over 35 mobility trials)

100K download: GSM

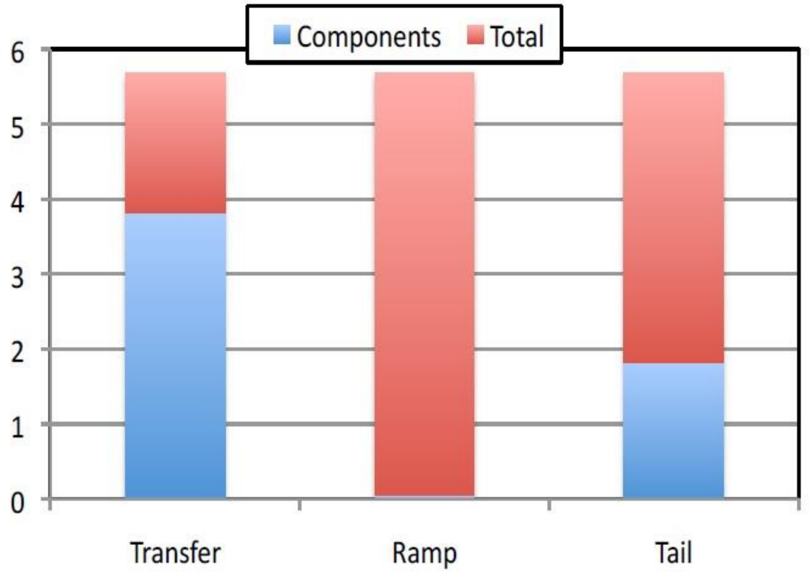
- GSM
  - Data transfer = 70%
  - Tail energy= 30%

Perform experiments with two Nokia device equipped with GSM

Tail energy: only accounts for 30% of transfer energy
Ramp: negligible
Tail time : 6sec

Data size dominates, rather inter transfer interval

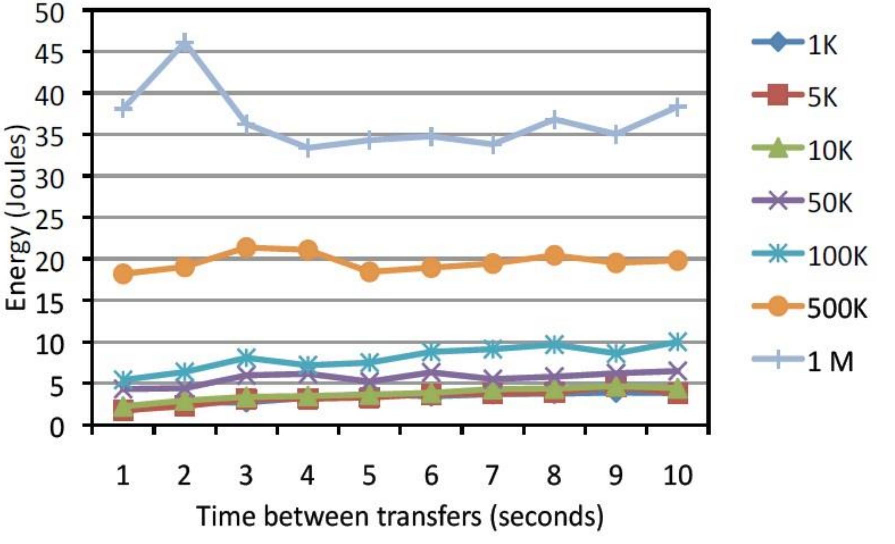
Measurement set up- GSM



(a) GSM: Energy components

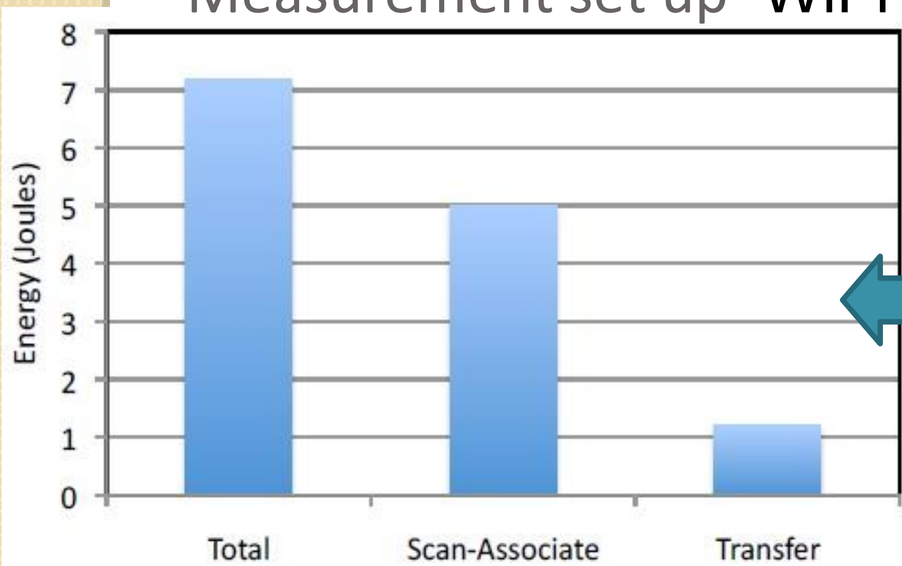
Avg energy does not depend on interval

500KB, varies from 19J to 21J



(b) GSM: Varying data sizes

Measurement set up- WiFi



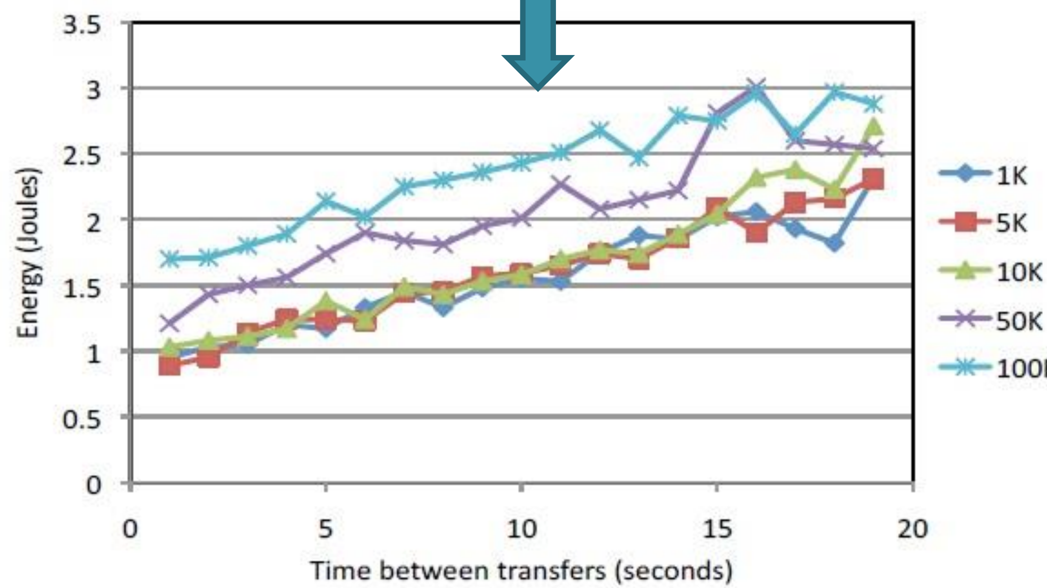
Data transfer = 32%

Scan/Associate = 68%

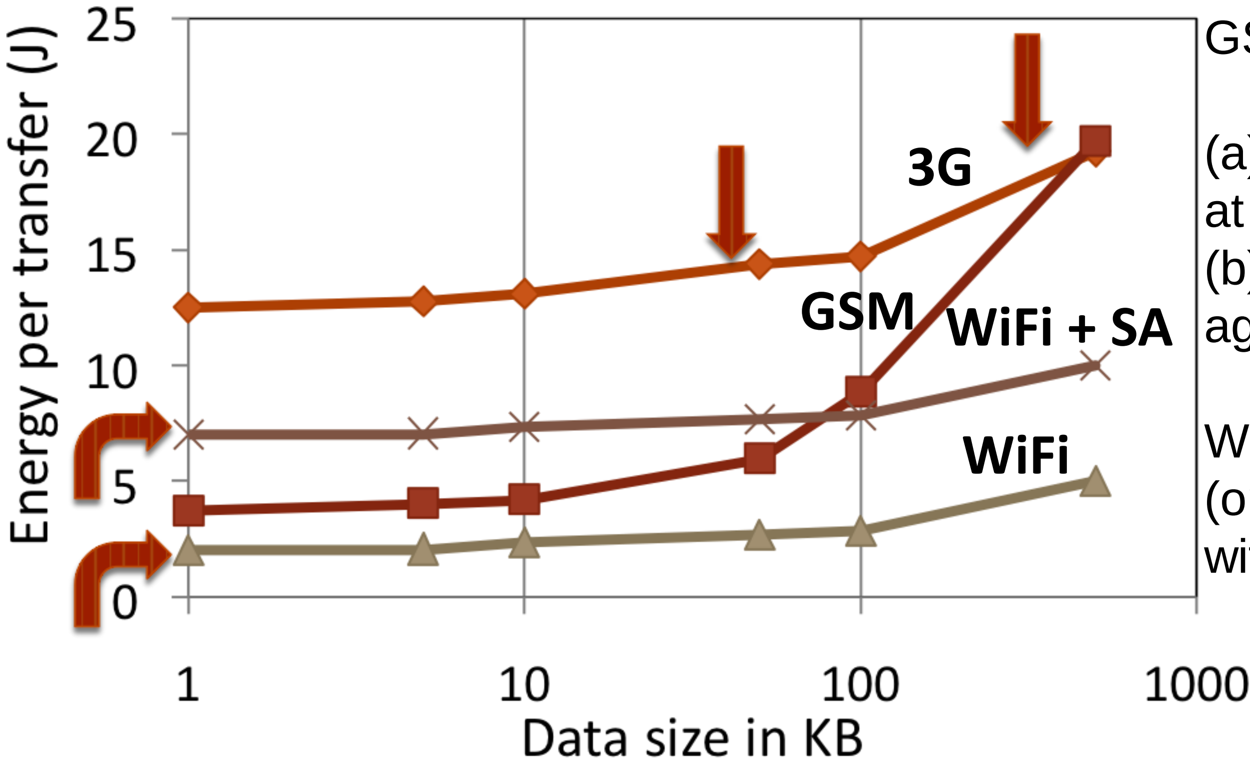
Scanning, association energy is times of transfer

Increases with interval time (maintenance energy)
No plateau

Wifi Measurements: (a) Proportional energy consumption for scan/associate and transfer (50KB). (b) Energy consumption for different transfer sizes against the inter-transfer time.



Comparison: Varying data sizes



3G : more energy
GSM –less 40-70%

(a) GSM radio operate
at low power level
(b) Tail time less (6sec
against 3G (12.5sec)

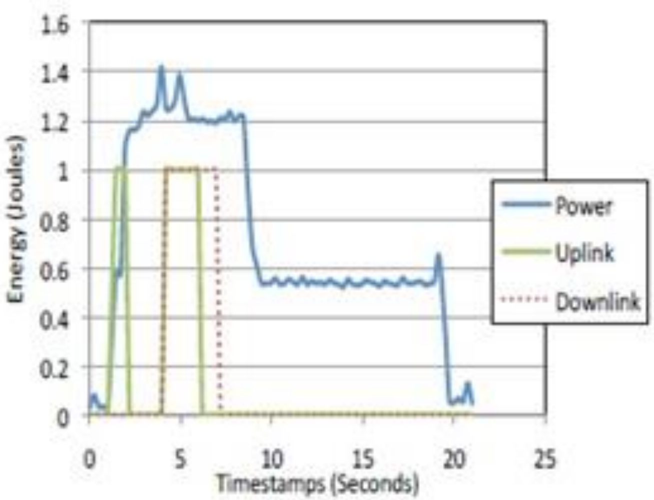
Wifi is energy efficient
(once it is connected
with AP)

- WiFi energy cost lowest without scan and associate
- 3G most energy inefficient

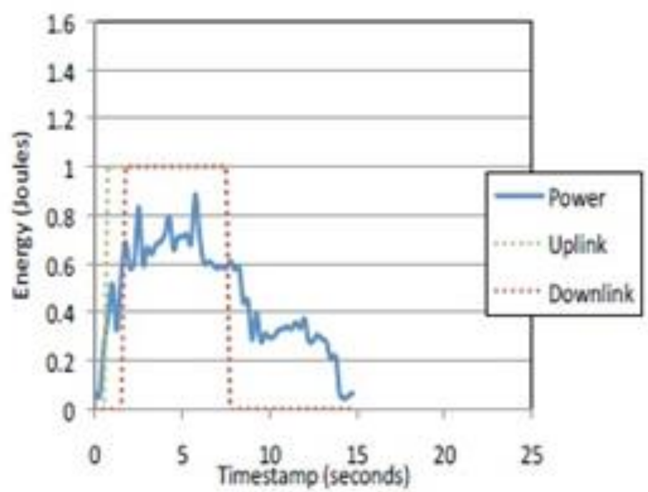
Instantaneous Power measurement over 3G and GSM

Measurement corresponds to 50KB data transfer
Uplink and downlink show the network activities (request and download)

GSM radio operates at the (i) low power, (ii) tail time less



(a) 3G: Power Profile - 50K

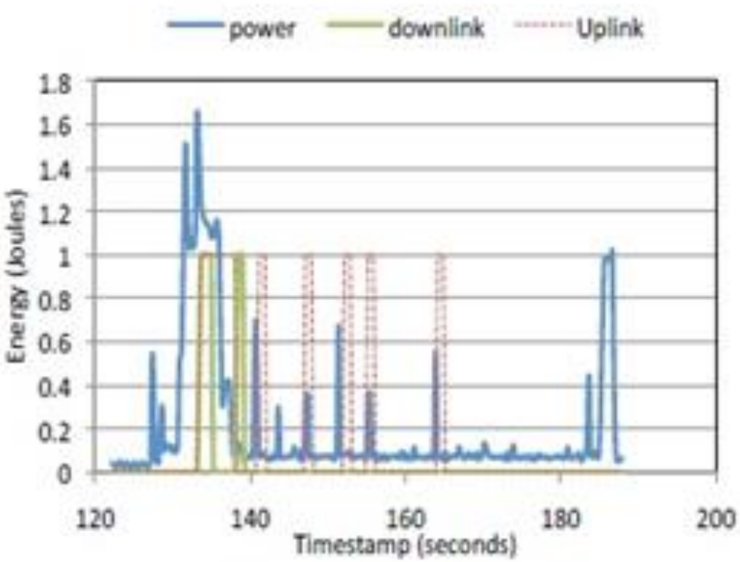


(b) GSM: Power Profile - 50K

Instantaneous Power measurement over Wifi

Measurement corresponds to 50KB data transfer

Initial spike corresponds to energy consumed for scanning and AP association



(c) Wifi: Power Profile - 50K

Energy model

Empirically estimate the energy consumption as a function of data size and interval between successive transfer

		3G	GSM	WiFi
Transfer Energy	$R(x)$	$0.025(x) + 3.5$	$0.036(x) + 1.7$	$0.007(x) + 5.9$
Tail energy	E	0.62 J/sec	0.25 J/sec	NA
Maintenance	M	0.02 J/sec	0.03 J/sec	0.05 J/sec
tail-time	T	12.5 seconds	6 seconds	NA
Energy per 50KB transfer with a 20-second interval		12.5 J	5.0 J	7.6 J

- Energy model derived from the measurement study

- R(x) denotes the sum of the Ramp energy and the transfer energy to send x bytes
- E denotes the Tail energy.
- For WiFi, R(x) to denotes the sum of the transfer energy and the energy for scanning and association

Summary

1. In 3G, nearly 60% of the energy is *tail energy*, is wasted in high-power states after the completion of a typical transfer. In comparison, the *ramp energy* spent in switching to this high-power state before the transfer is small. The tail and ramp energies can be amortized over frequent successive transfers, but only if the transfers occur within *Tail time* of each other.
2. In GSM, although a similar trend exists, the *tail time* is much smaller compared to 3G (6 vs. 12 secs). Furthermore, the lower data rate of GSM implies that more energy is spent in the actual transfer of data compared to in the tail.
3. In WiFi, the association overhead is comparable to the tail energy of 3G, but the data transfer itself is significantly more efficient than 3G for all transfer sizes. When the scan cost is included, WiFi becomes inefficient for small sized transfers compared to GSM, but is still more energy efficient than 3G.

Outline

- Measurement study
- **TailEnder design**
- Evaluation

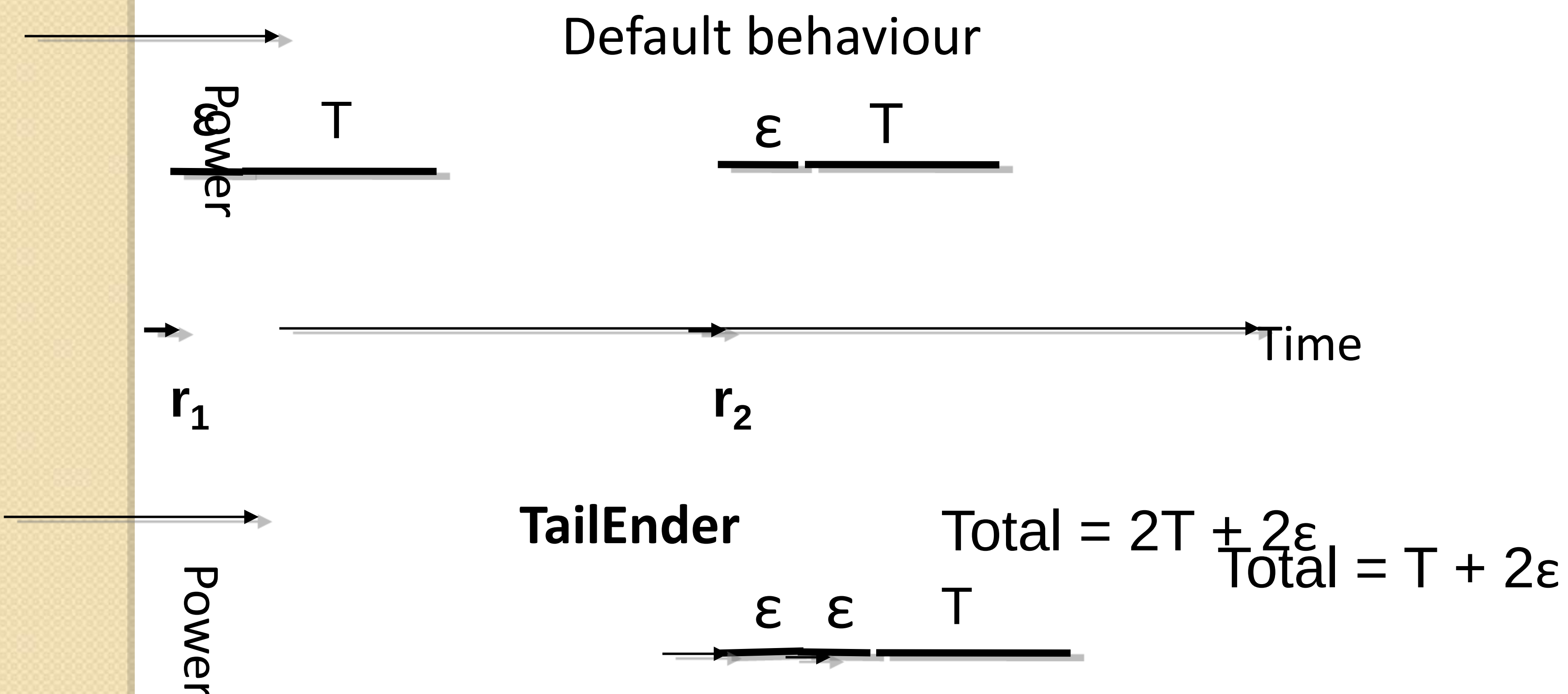
TailEnder: reduce energy consumption on mobile phone

- **Observation**: Several applications can
  - Tolerate delays: Email, Newsfeeds

  (user may wait for a short time if it results substantial energy save)
  - Prefetch: Web search

- (DT applications) Schedules transmissions such that time spent by the device at high energy state can be minimized
- (Prefetching) Determines number of documents to prefetch

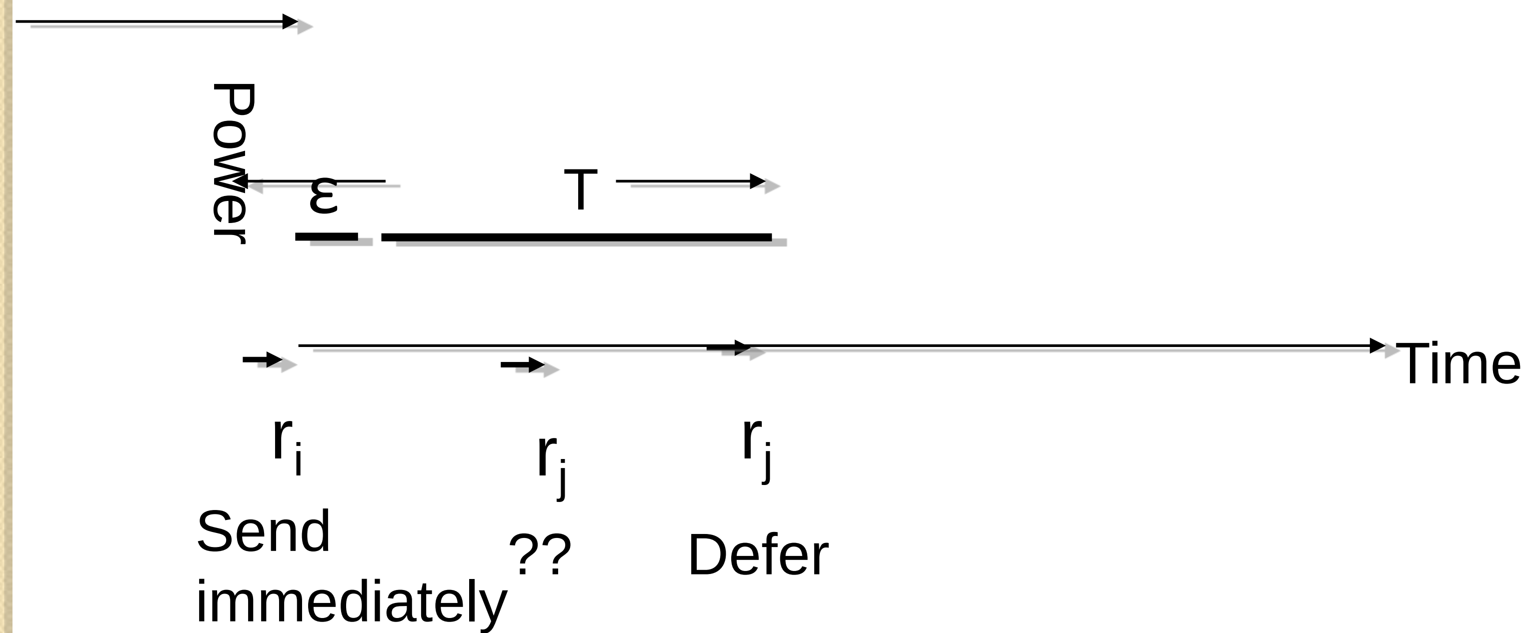
Exploiting delay tolerance



How can we schedule requests such that the time in the high power state is minimized?

TailEnder scheduling

- Online problem: No knowledge of future requests



Consider n equal sized requests

where each request r_i is associated with an arrival time a_i and a deadline d_i by which it needs to be transmitted. When the request r_i is scheduled to be transmitted at time s_i, the radio transitions to the high power state, transfers request r_i instantaneously, and remains in the high power states for T time units, equal to the *Tail time*. Also, based on our measurements, we ignore the energy overhead to switch to the high-power state. Note that when multiple requests are transmitted at the same time, the device is in the high power state only for T time units. Let Z denote the the total time spent in high-power states for a given schedule of requests. The problem is to compute a schedule $s_1, s_2, \cdots, s_n$ that minimizes Z, such that $a_i \leq s_i \leq d_i$.

In practice, we need to solve an online version of the problem, where arrivals and their deadlines are not known in advance. TailEnder uses a simple online algorithm to schedule transmission of an incoming request r_i at time t. The main idea of TailEnder is to transmit a request r_i if either

- the request arrives within a time $x \cdot T$ from the previous deadline d', or
- the request's deadline is reached.

We show that the algorithm provably achieves a competitive ratio of 1.28 compared to the optimal if the value of x is set to 0.57. Further, we show that no deterministic online algorithm can achieve a competitive ratio lower than 1.28.

TailEnder algorithm

TailEnder scheduler (t, r_i, d_i, a_i):

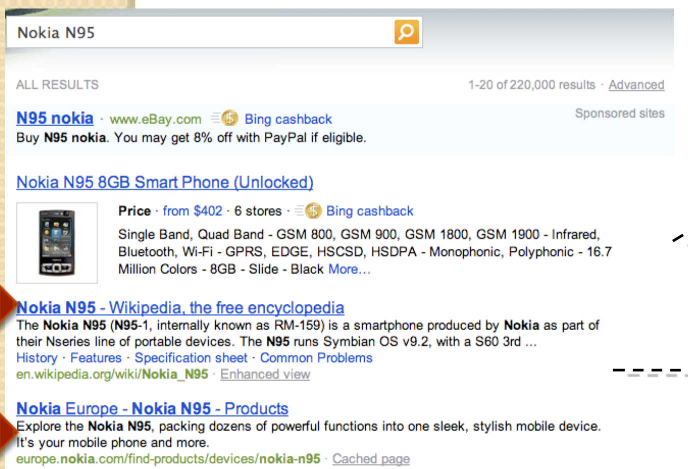
1. Let Δ be the last deadline when a packet was transmitted (initialized to $-\infty$ and reset in Step 3(c)).
2. If $(t < d_i)$
   (a) if $(\Delta + \rho \cdot T > a_i)$, transmit.
   (b) else add the request to queue Q.
3. If $(t == d_i)$
   (a) Transmit r_i
   (b) Transmit all requests in Q and set $Q = null$
   (c) Set $\Delta = d_i$

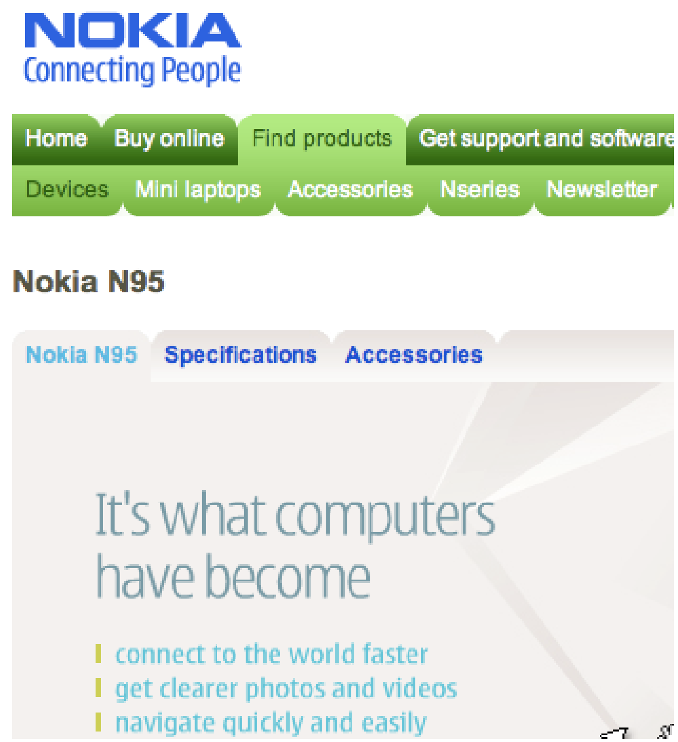
Outline

- Measurement study
- **TailEnder Design**
  - Application that are delay tolerant
  - **Application that can prefetch**
- Evaluation

TailEnder for web search

rrent web search model





Idea: Prefetch web pages.

Challenge: Prefetching is not free!

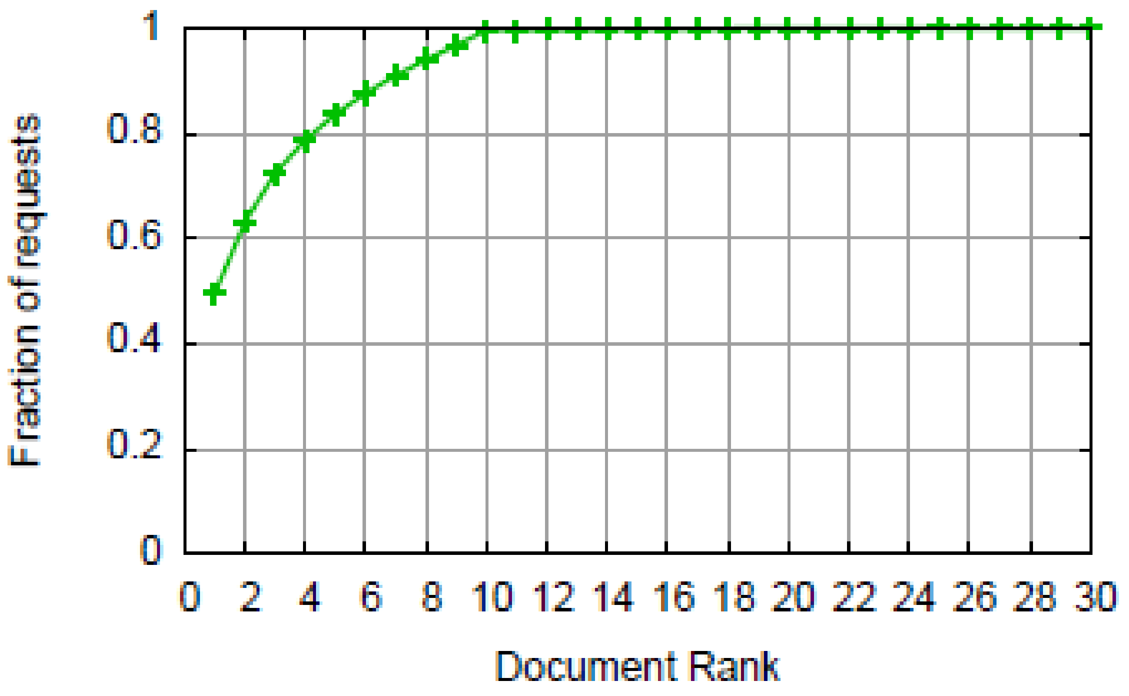


Figure 11 shows the distribution of web documents that are requested by the user when searching the web. The graph is generated using Microsoft Search logs [6]. The logs contain over 8 million user queries and were collected over a month. Figure 11 shows that 40% of the time, a user requests for the first document presented by the search engine. A user requests for a document ranked 11 or more, less than 0.00001% of the time.

Expected fraction of energy savings if top k documents are pre-fetched:

$$\frac{E \cdot p(k) - R(k)}{TE}$$

User think time is> tail timer

- k be the number of prefetched documents
- p(k) be the probability that a user requests a document within rank k.
- E is the Tail energy
- R(k) be the energy required to receive k documents.
- TE is the total energy required to receive a document.

TE = (ramp energy + receive the document + tail energy)

How many web pages to prefetch?

- Analyzed web logs of 8 million queries
  - Time, url clicked, size of he doc
- Computed the probability of click at each web page rank

TailEnder prefetches the top 10 web pages per query

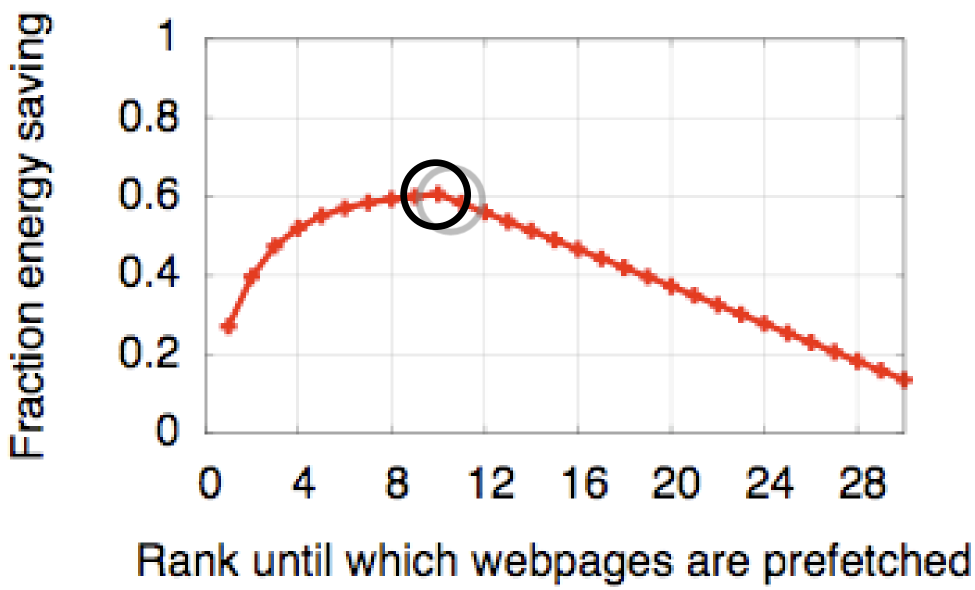


Figure 12 shows the expected energy savings for varying k as estimated by Equation 4.2.1. The value of $p(k)$ is obtained from statistics presented in Figure 11, and E, $R(k)$ and TE are obtained from the 3G energy measurements (in Table 1). We set the size of a document to be the average web document size seen in the search logs.

Figure 12 shows that prefetching 10 web documents maximizes the energy saved. When more documents are prefetched, the cost of prefetching is greater than the energy savings. When too few documents are prefetched, the expected energy savings is low since the user may not request a prefetched document. Therefore, for Web search applications, TailEnder prefetches 10 web documents for each user query. In Section 5, we show that TailEnder can save substantial amount of energy when applied to real Web search sessions.

Outline

- Measurement study
- TailEnder Design
- **Evaluation**

Applications

	User 1	User 2	User 3
Incoming	446	405	321
Incoming size	214MB	162MB	161MB
Outgoing	219	183	354
Outgoing size	107MB	66MB	178MB

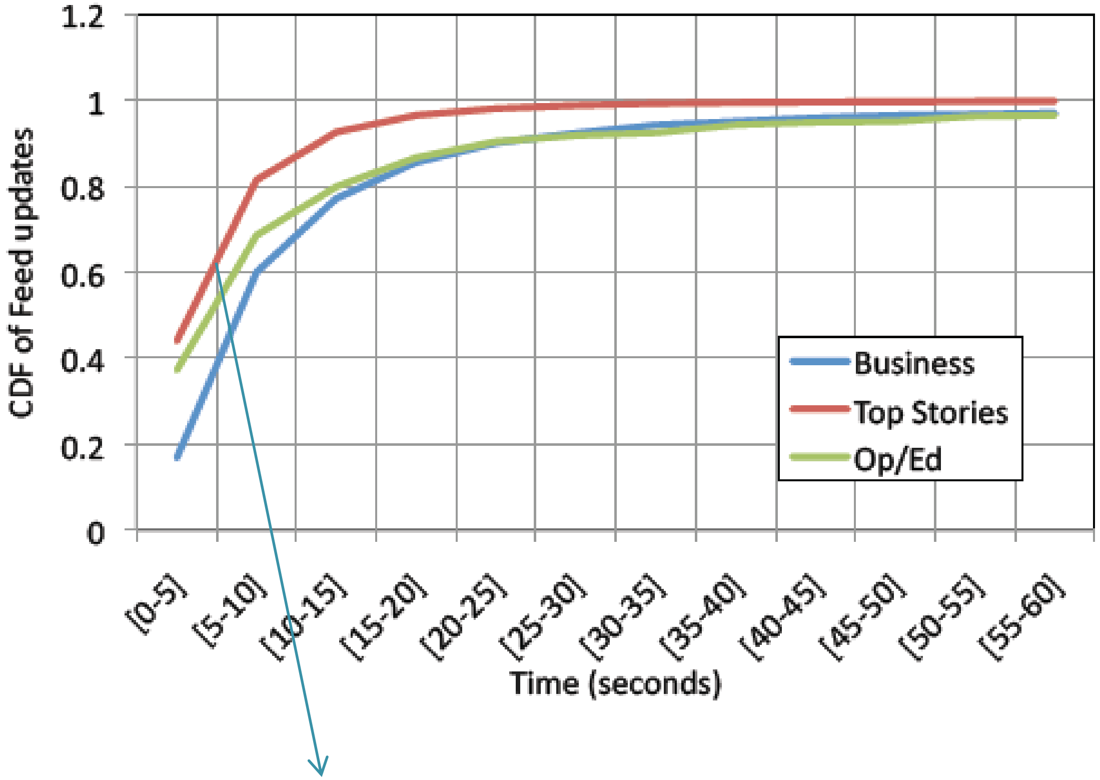
- Email:
  - Data from 3 users over a 1 week period
  - Extract email timestamp and size
- News Feed
  - 10 different Yahoo news feed
  - Polled Once in every 5 sec, for 3 days
  - Log the arrival time and size of new feed

Feed	Total stories
Opinion/Editorials	507
Health	2177
Technology	4659
Business	5616
Sports	7265
Politics	12069
US News	12389
Entertainment	16757
World	21006
Top Stories	23232

- Web search: Microsoft search log (8M queries for one month)
  - Time and url clicked for each query, size of the doc.
  - Extracted click logs from a sample of 1000 queries

News feed—inter arrival times of update

Top stories: higher frequency



60% update within 10-15 sec inter arrival time

Evaluation

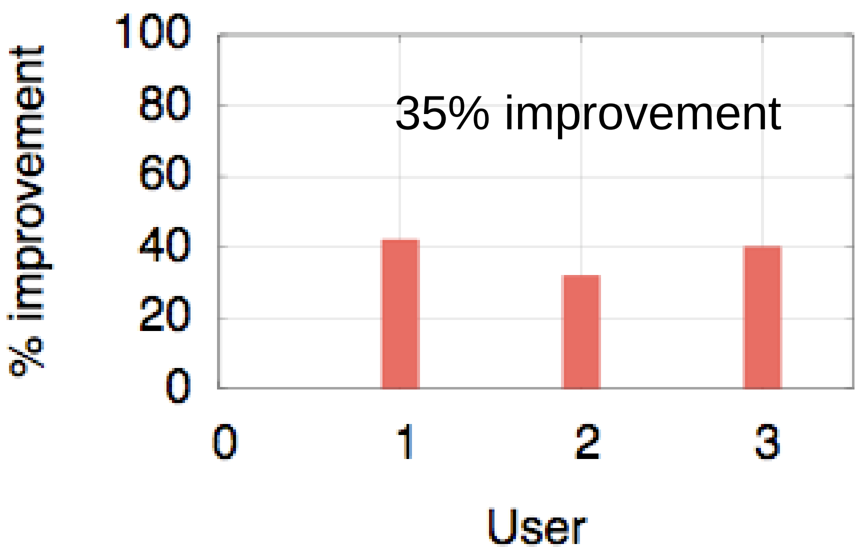
- Methodology
  - Model-driven simulation
  - Emulation on the phones
- Baseline
  - Default algorithm that schedules every requests when it arrives

Model driven evaluation

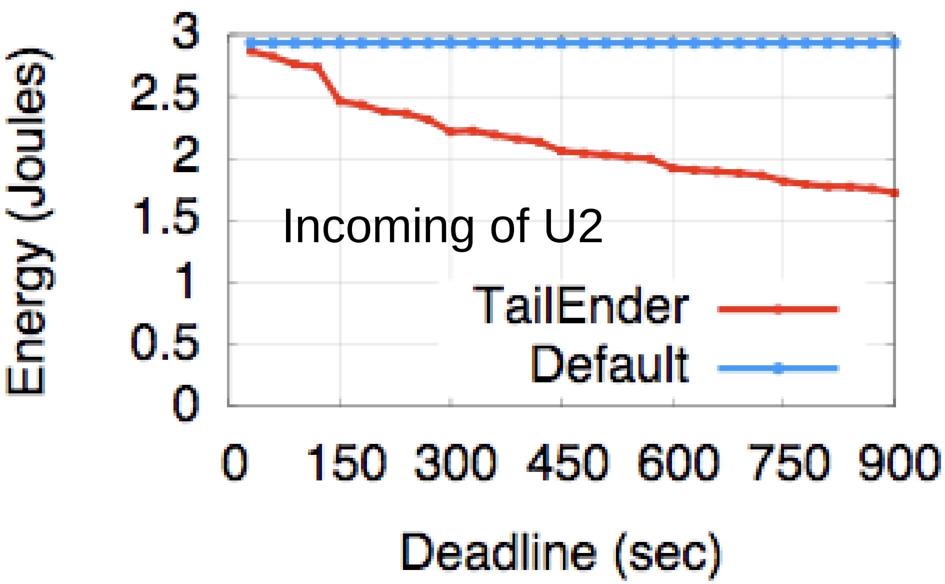
The application trace consists of a sequence of arrivals of the form (s_i, a_i), where s_i is the size of the request and the a_i is the time of arrival. For example, for the news feed application, a_i is the time a topic is updated and s_i is the size of the update. Requests could be downloads as in news feeds or uploads as in outgoing emails. The Default protocol schedules transmissions as requests arrive. For delay tolerant applications, TailEnder schedules transmissions using the algorithm shown in Figure 10. For applications that benefit from prefetching, TailEnder schedules transmissions for all prefetched documents. For both protocols, we estimate the energy consumption as a sum of the *Ramp energy* (if the device is not in high power state), transmission energy and the energy consumed because of staying in the high power state after transmission. If a request is scheduled for transmission before the *Tail time* of the previous transmission, the previous transmission does not incur an overhead for the entire *Tail time*.

Model-driven evaluation: Email

With delay tolerance = 10 minutes



For increasing delay tolerance (3G)



TailEnder nearly halves the energy consumption for a 15 minute delay tolerance. (Over GSM, improvement is only 25%)

With delay tolerance = 10 minutes

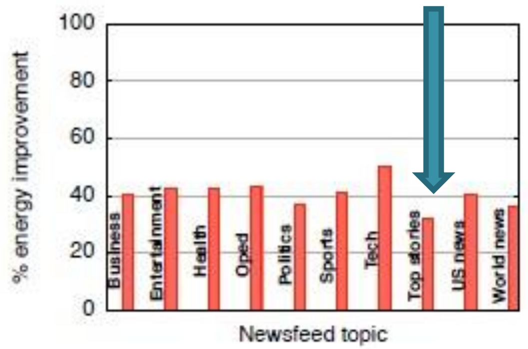


Figure 14: News feed: Energy improvement in using TailEnder for different news feed topics

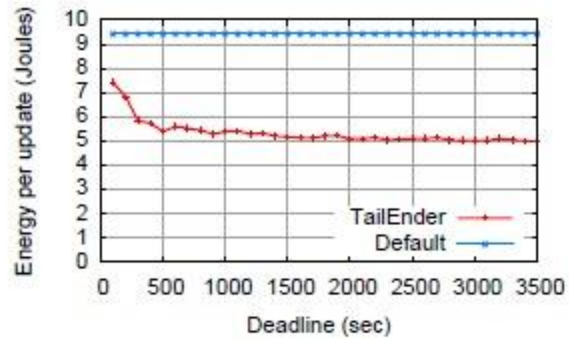


Figure 15: Newsfeed: The energy consumption of TailEnder and Default for varying deadline over 3G

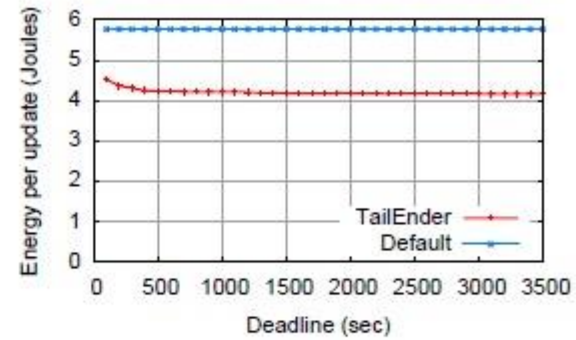


Figure 16: News feed: The energy consumption of TailEnder and Default for varying deadline over GSM

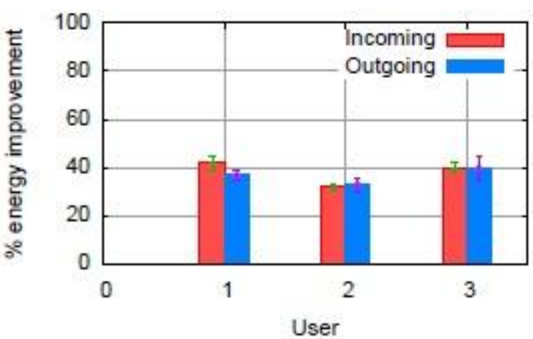


Figure 17: Email: Energy improvement using TailEnder for email application

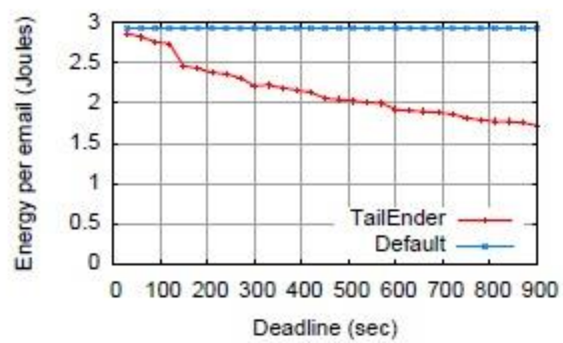


Figure 18: Email: The energy consumption of TailEnder and Default for varying deadline over 3G

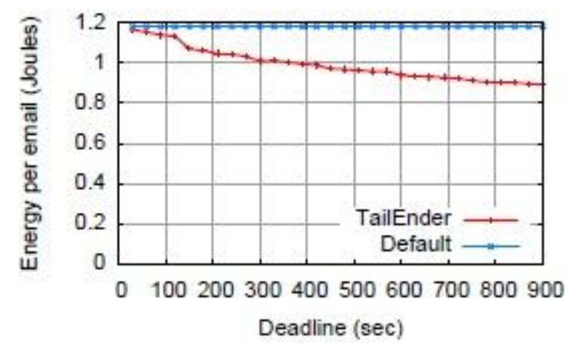


Figure 19: Email: The energy consumption of TailEnder and Default for varying deadline over GSM

Model-driven evaluation: news feed

der for each of the news feed topics. We set the deadline for sending the news feeds update to 10 minutes; i.e., a newsfeed content needs to be sent to the user with a maximum delay of 10 minutes since the content was updated. The average improvement across all news feeds is 42%. The largest improvement is observed for the Tech news feed at 52% and the smallest improvement for the Top story news feed at 36%. One possible reason for the top story news feed to yield lower performance improvement is that 60% of the top story updates arrive within 10 seconds, which is the less than the *Tail time* of 12.5 seconds (see Figure 13). Therefore, Default does not incur a *Tail energy* penalty for a large portion of the updates.

Model-driven evaluation: news feed

Deadline variation

Figure 15 and 16 show the expected energy consumption for *business* news feeds using TailEnder and Default for varying deadline settings over 3G and GSM respectively. Figure 15 shows that as deadline increases to 25 minutes(1500 seconds), TailEnder's energy decreases to nearly half of the energy consumption of Default, decreasing from 10 Joules to 5 Joules per update. When sending data over GSM, the energy decreases from 6 Joules to 4 Joules when using TailEnder, compared to Default, yielding a 30% improvement. The improvements of TailEnder over default are smaller in GSM compared to 3G. However, the improvements are substantial when considering the relative proportions of GSM's *Tail energy*and transfer energies.

Model-driven evaluation: Web search

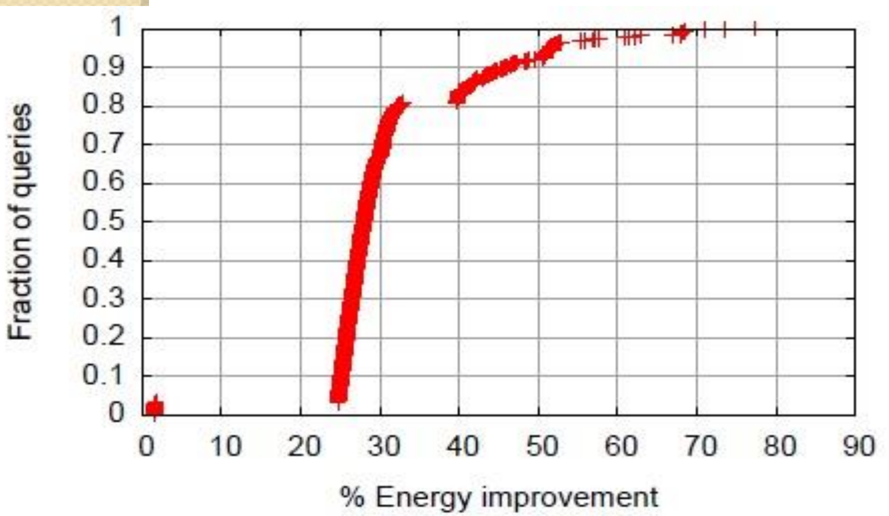
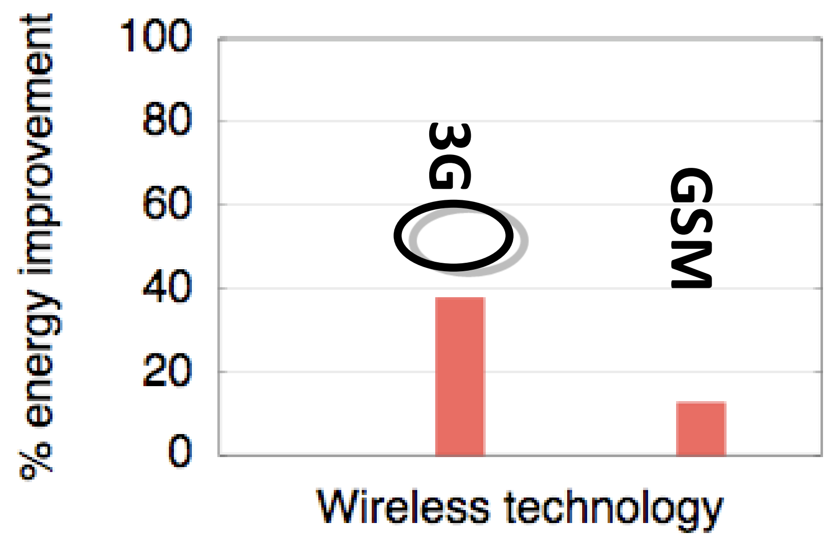


Figure 21: Web search: CDF of the Energy improvement using TailEnder for each query



Model-driven evaluation: Web search

Figure 20 shows the energy improvement using TailEnder for Web search application, when sending data over 3G and GSM. For Web search, TailEnder prefetches the top 10 documents for each requested query. Default only fetches documents that are requested by the user. TailEnder reduces energy by nearly 40% when data is sent over 3G and by about 16% when data is sent over GSM.

To understand the distribution of energy savings per query, we plot the CDF of the energy improvement in Figure 21. The plot shows that about 2% of the queries see little energy improvement. TailEnder reduces energy for 80% of the queries by 25–33%. For the remaining 18% of the queries, TailEnder reduces energy by over 40%. We find that the 18% of the queries that benefits most by TailEnder's prefetching are queries for which the user requested 3 or more documents.

Using Wifi for energy saving

When WiFi is always available, the energy consumption is 10 times lower compared to Default and more than 4 times lower compared to TailEnder for all three applications. Even when WiFi is available only 50% of the time, sending data over WiFi reduces energy consumption by 3 times compared to Default for all three applications. Previous works [21] do not observe such substantial energy savings when using WiFi

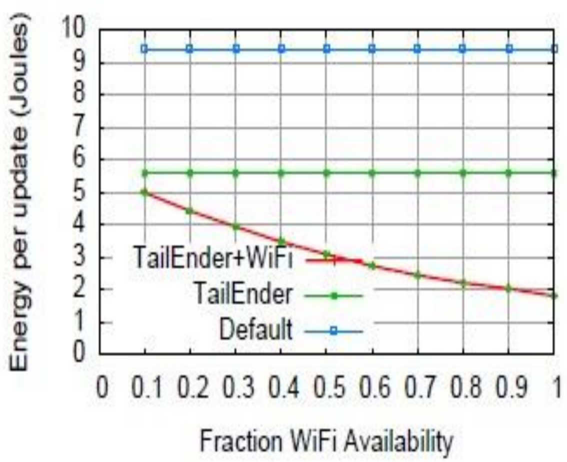


Figure 22: News feed. Vertical bars show 95% confidence interval

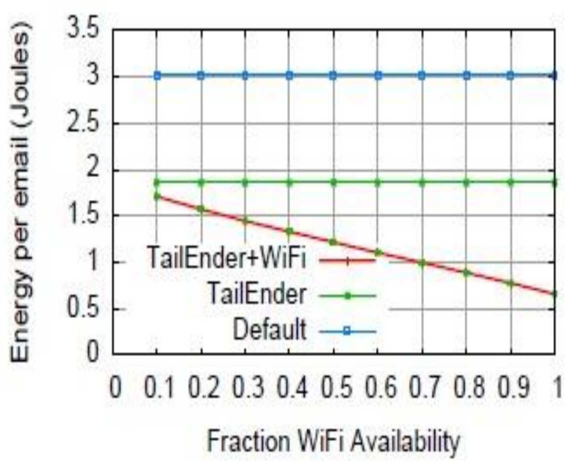


Figure 23: E-mail. Vertical bars show 95% confidence interval

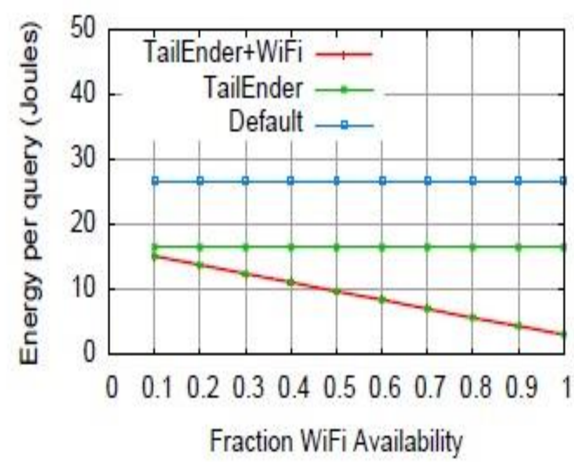


Figure 24: Web Search. Vertical bars show 95% confidence interval

Experiments on mobile phone

We conduct data transfer experiments on the phone using application-level traces. We convert an application trace into a sequence of transfers $S = \{< s_1, a_1 >, < s_2, a_2 >, \cdots, < s_n, a_n >\}$, such that data of size s_i is downloaded by the mobile phone at time a_i. Then, from a fully charged

We run two sequences of transfer, one generated by TailEnder and the other by Default. Given an application trace, TailEnder schedules the transfers according to whether the application is delay tolerant or can benefit from prefetching. Default schedules transfers as they arrive. We conduct the experiments for two applications: downloading Tech news feeds and Web search. For the news feed application, the metric is the number of stories downloaded and for Web search the metric is the number of queries for which all user requested documents were delivered.

TailEnder downloads more than 60% news feed updates compared to Default, and the total size of data downloaded by protocol increases from 127 MB to 240 MB providing a 47% improvement. Our model-based evaluation showed that

	Default	TailEnder
Stories	1411	3900
Total transfer size	127 MB	240 MB

Table 4: News feeds experiment. TailEnder downloads more than twice as many news feeds compared to Default

	Default	TailEnder
Queries	672	1011
Documents	864	10110
Transfers	1462	1011
Average transfers per query	9.3K	147.5K

Table 5: Web search experiment. TailEnder downloads 50% more queries compared to Default

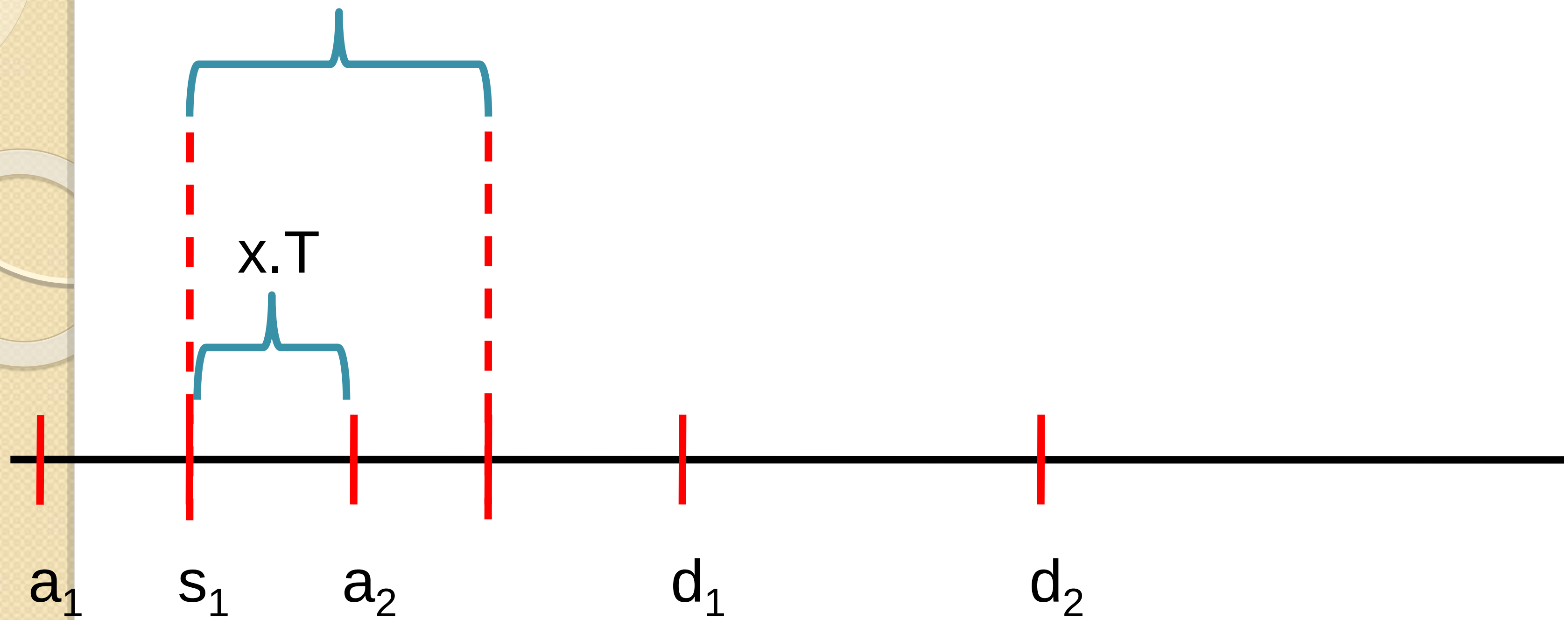
TailEnder : Theorem

Any online algorithm ALG can be atmost 1.28 competitive with the offline adversary OPT

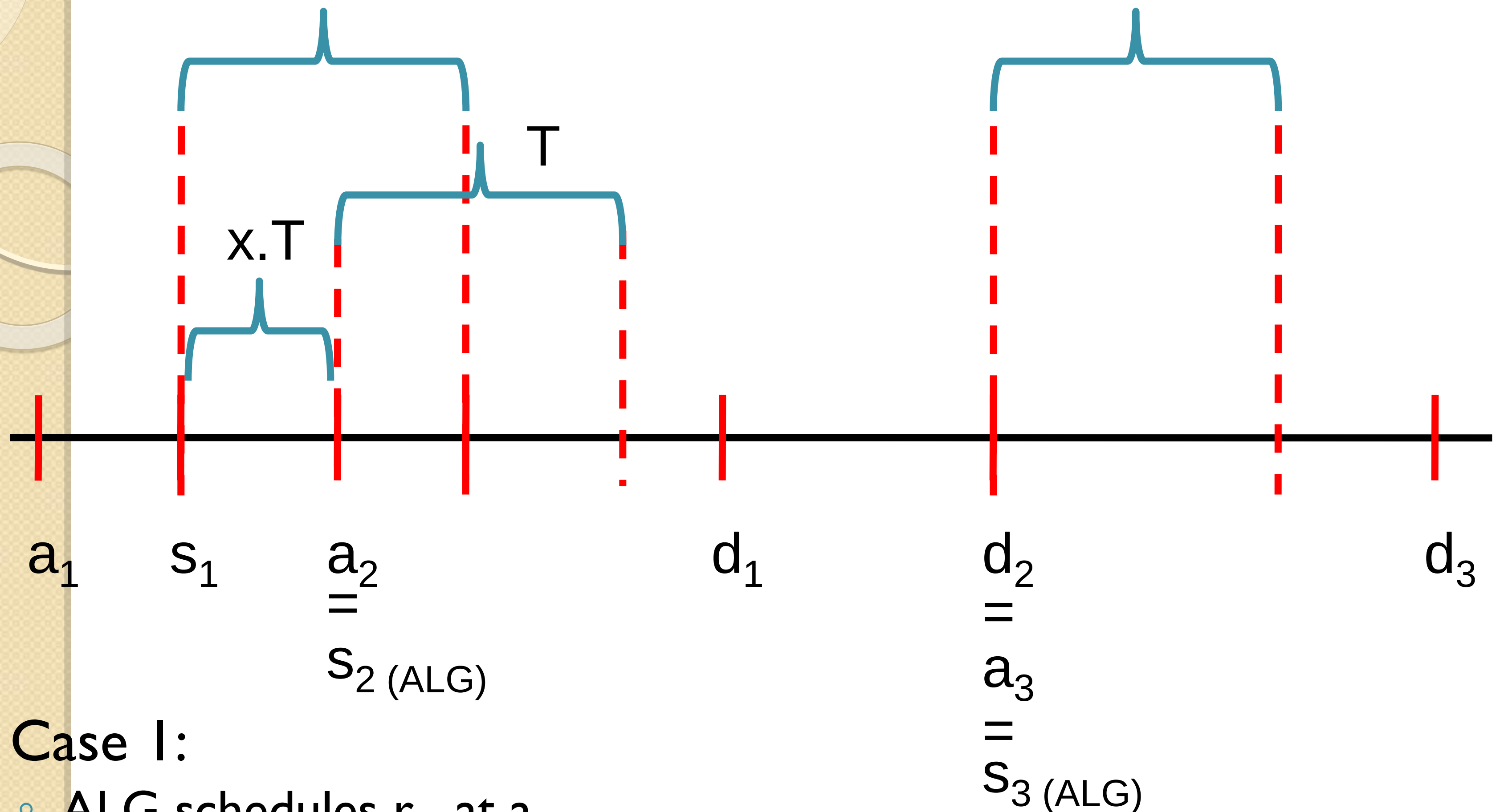
Z → time spent by the mobile in high-energy state

We have to show, Z(ALG) / Z(OPT) =1.28

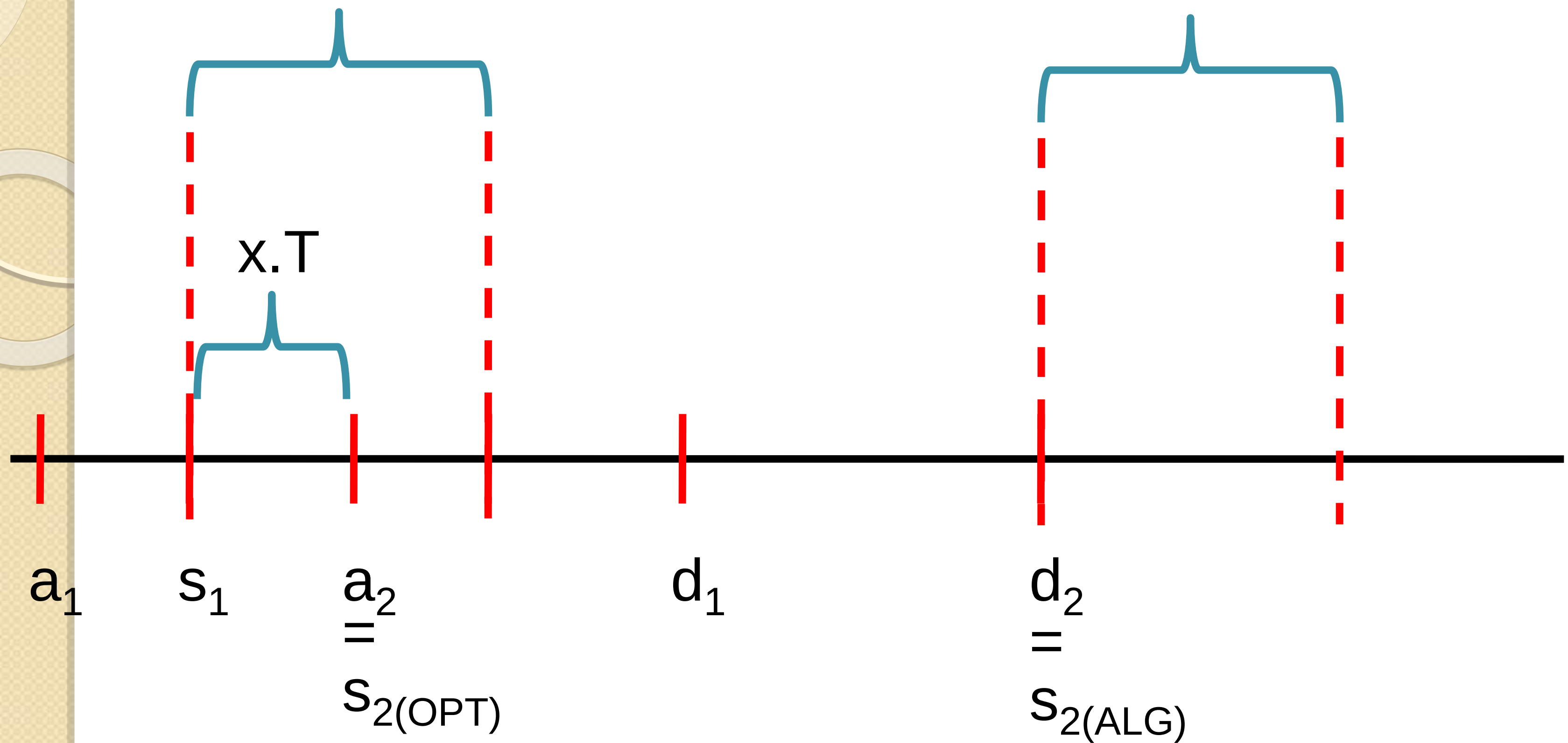
PROOF. We prove the theorem by constructing the offline adversary, OPT, that incrementally generates new requests after observing the actions of ALG. OPT generates a new request at time $x \cdot T$ after the ALG schedules a previous request. We show that $\frac{Z(ALG)}{Z(OPT)}$ is maximized when x is set to 0.57; in other words, when OPT generates a new request at $0.57T$ after ALG schedules a request, OPT can force the ALG schedule to remain in the high power state for the longest time compared to its own schedule.



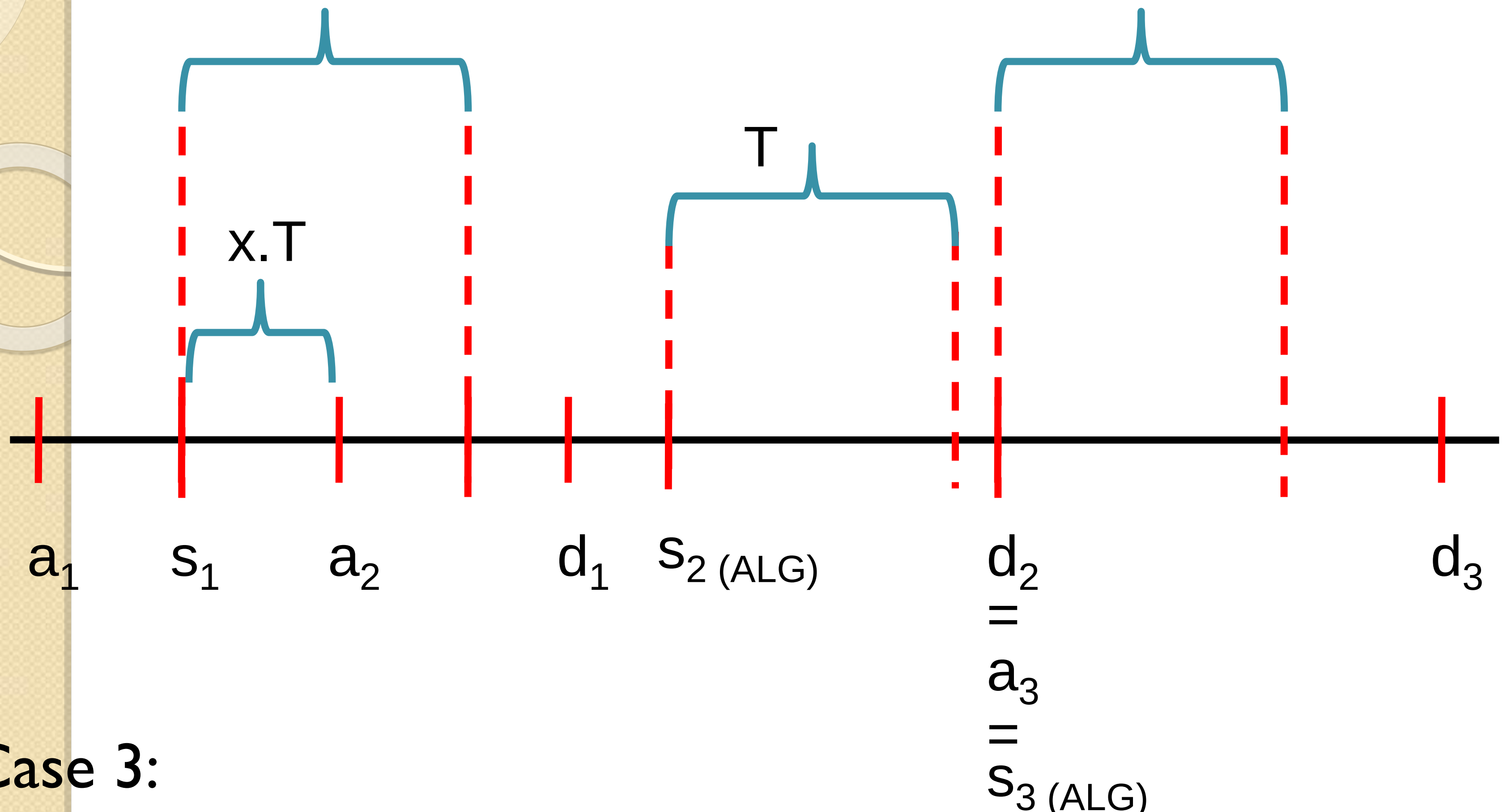
- First request $r_1 = (a_1 , d_1)$ [T → Tail time]
- ALG and OPT schedule it at s_1 [a → arrival time]
- Step 1: OPT generates $r_2 = (a_2 , d_2)$ [d → deadline]
  - s.t $a_2 = s_1 + xT$ [s → sched. time]
  - $d_2 >> T$
- ALG schedules r_2 in one of the following 3 ways



- Case 1:
  - ALG schedules r_2 at a_2
  - In response, OPT generates $r_3 = (a_3 , d_3)$ s.t $a_3 = d_2$
  - OPT schedules r_2 and r_3 at d_2
  - $Z(OPT) = T\ (r_1) + T(r_2 \text{ and } r_3) = 2T$
  - $Z(ALG)=T\ (r_1) + xT\ (r_2) + T(r_3)$
    $= (x+2)\ T$



- Case 2:
  - ALG schedules r_2 at d_2
  - In response, OPT schedules r_2 at a_2 and does not generate any more request
  - $Z(OPT) = xT\ (r_1) + T(r_2) = (1+x)T$
  - $Z(ALG) = T\ (r_1) + T\ (r_2) = 2T$



- Case 3:
  - ALG schedules r_2 at s_2 s.t [$a_2 < s_2 < d_2$]
  - In response, OPT generates $r_3 = (a_3 , d_3)$ s.t $a_3 = d_2$
  - OPT schedules r_2 and r_3 at d_2
  - Z(OPT) = T (r_1)+ T(r_2 and r_3) = 2T
  - Z(ALG) >= xT (r_1) + T (r_2) + T(r_3)
    >= (x+2) T

- Competitive ratio
  - Z(ALG) /Z(OPT) = (2+x) / 2 [Choice 1 & 3]
    = 2/(1+x) [Choice 2]
  - Lower bound of competitive ratio min ((2+x)/2 , 2/(1+x))
  - Compute the value of x by solving

$$(2+x) / 2 = 2 / (1+x)$$

Solving this, x=0.57

Z(ALG) / Z(OPT) = 1.28

request at time $0.57T$ after the previous schedule, ALG is forced to be in the high energy state 1.28 times longer than OPT for any choice it makes. If $x > 0.57$, then ALG can schedule according to *Choice 2* and reduce the competitive ratio to less than 1.28. Similarly, if $x < 0.57$, then ALG can schedule according to *Choice 1* and reduce the competitive ratio.

In summary, OPT can generate new requests such that ALG is forced to be in the high energy state 1.28 times longer than OPT. Therefore any online algorithm ALG can at most be 1.28 competitive with an offline adversary.