Interpreting the significance of Android energy optimisation by collecting large-scale usage information

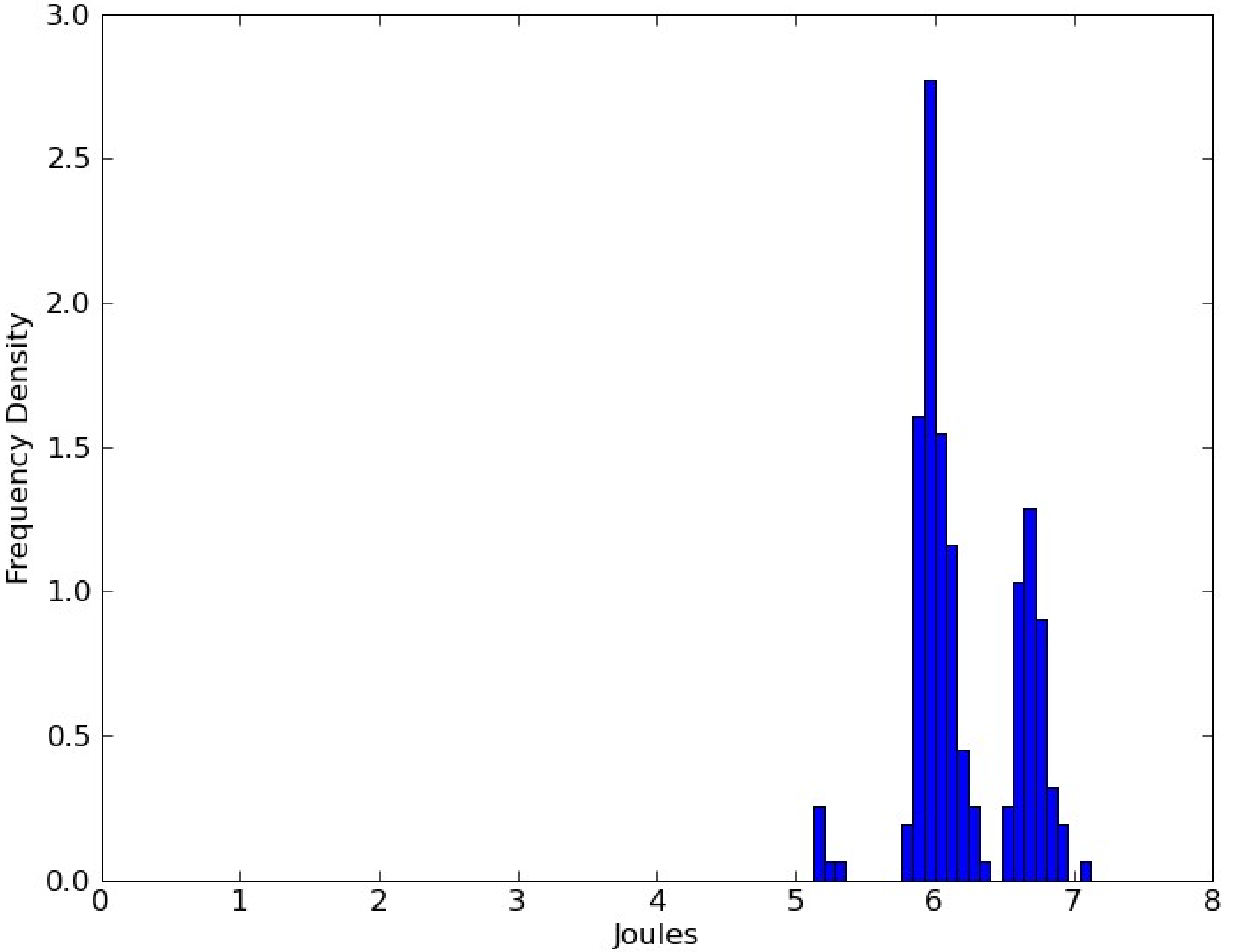
Andrew Rice
August-2011

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Part 1: We want to know how much energy a particular ***action*** will consume

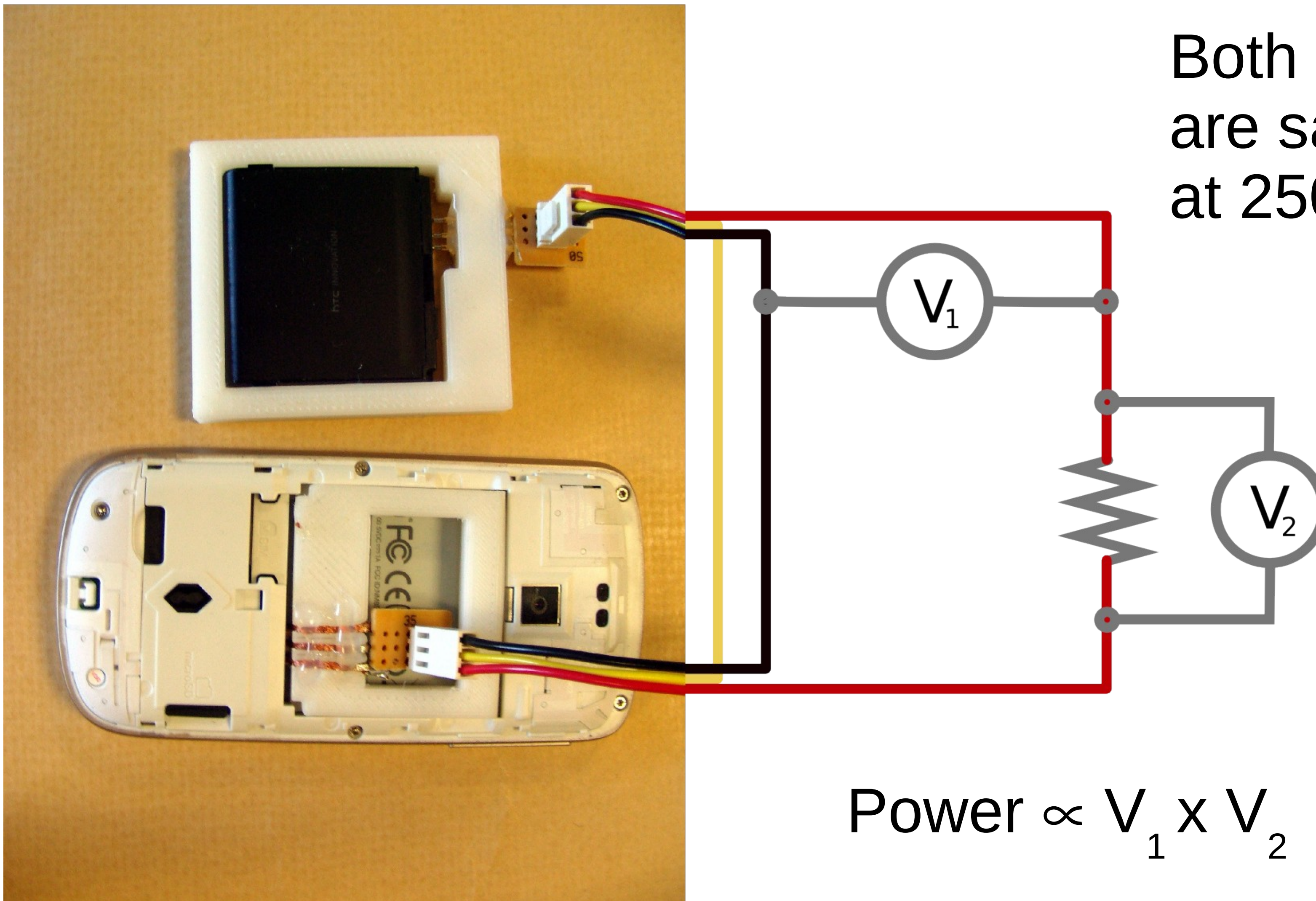
Part 2: We want to know if this is significant in real usage

Example: joining the wireless network consumes 6 Joules



HTC G1 (or Magic), Android 1.1, 194 trials

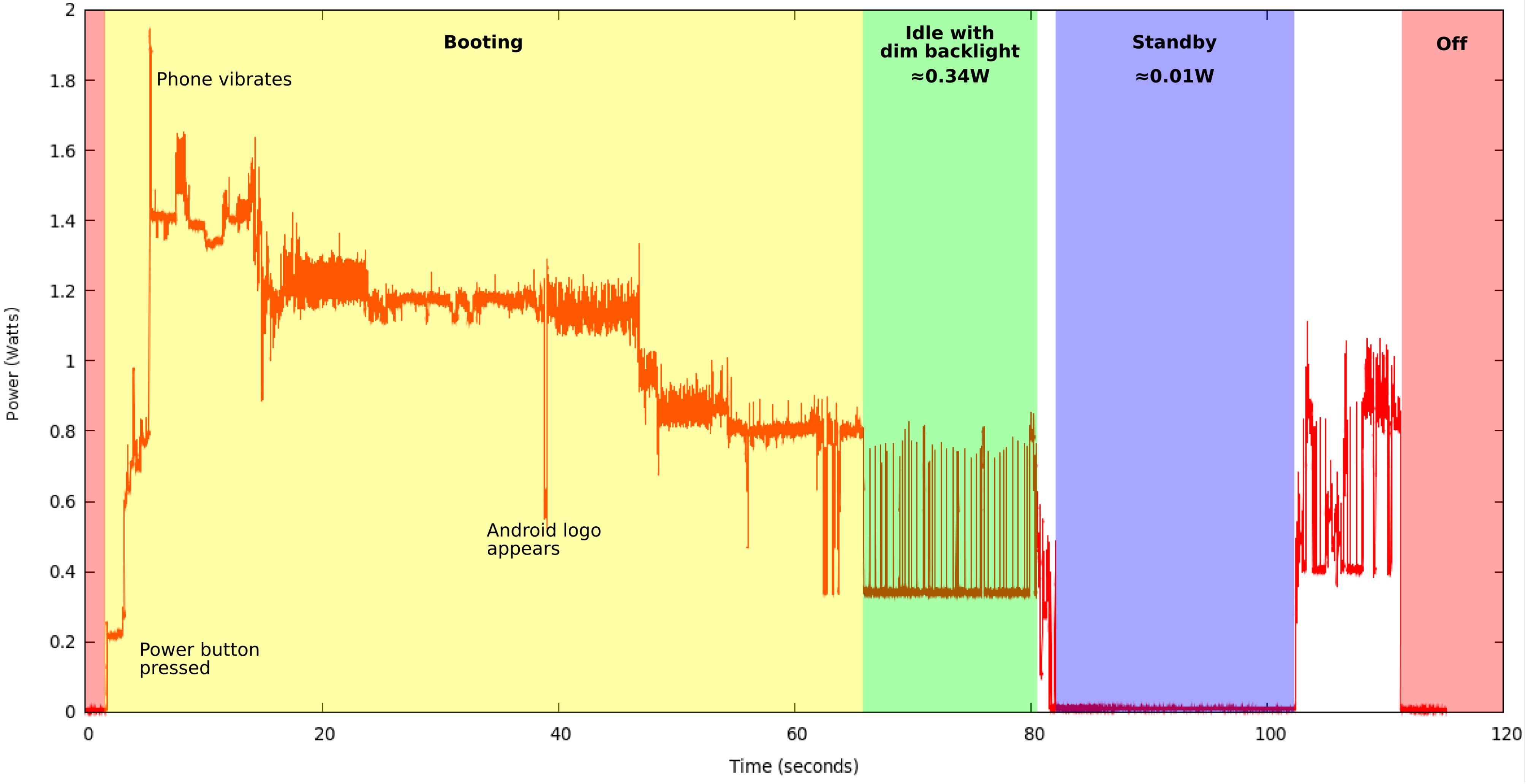
We measure energy consumption by intercepting the power supply



Both voltages are sampled at 250 kHz

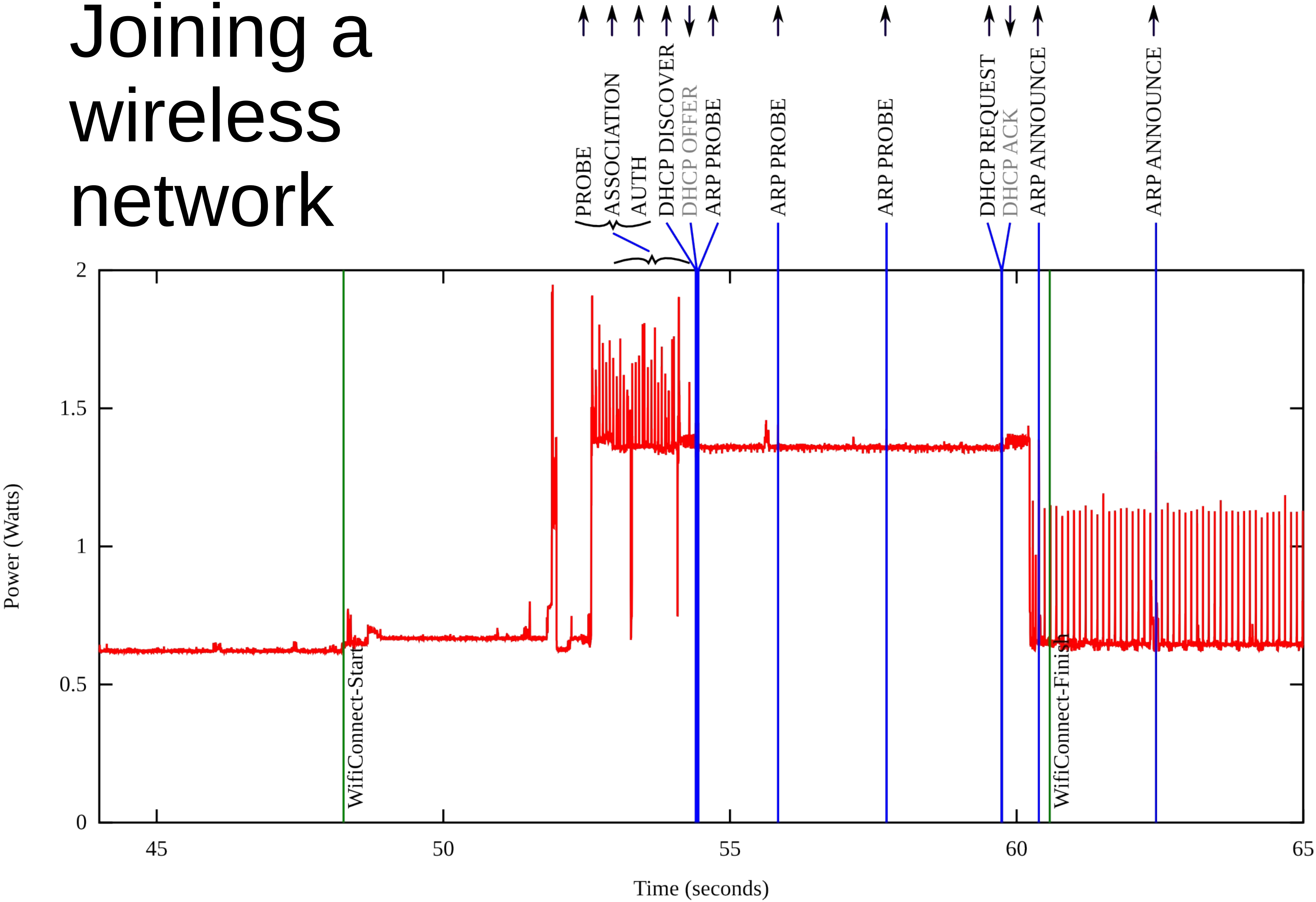
Power $\propto V_1 \times V_2$

Trace of the G1 boot process



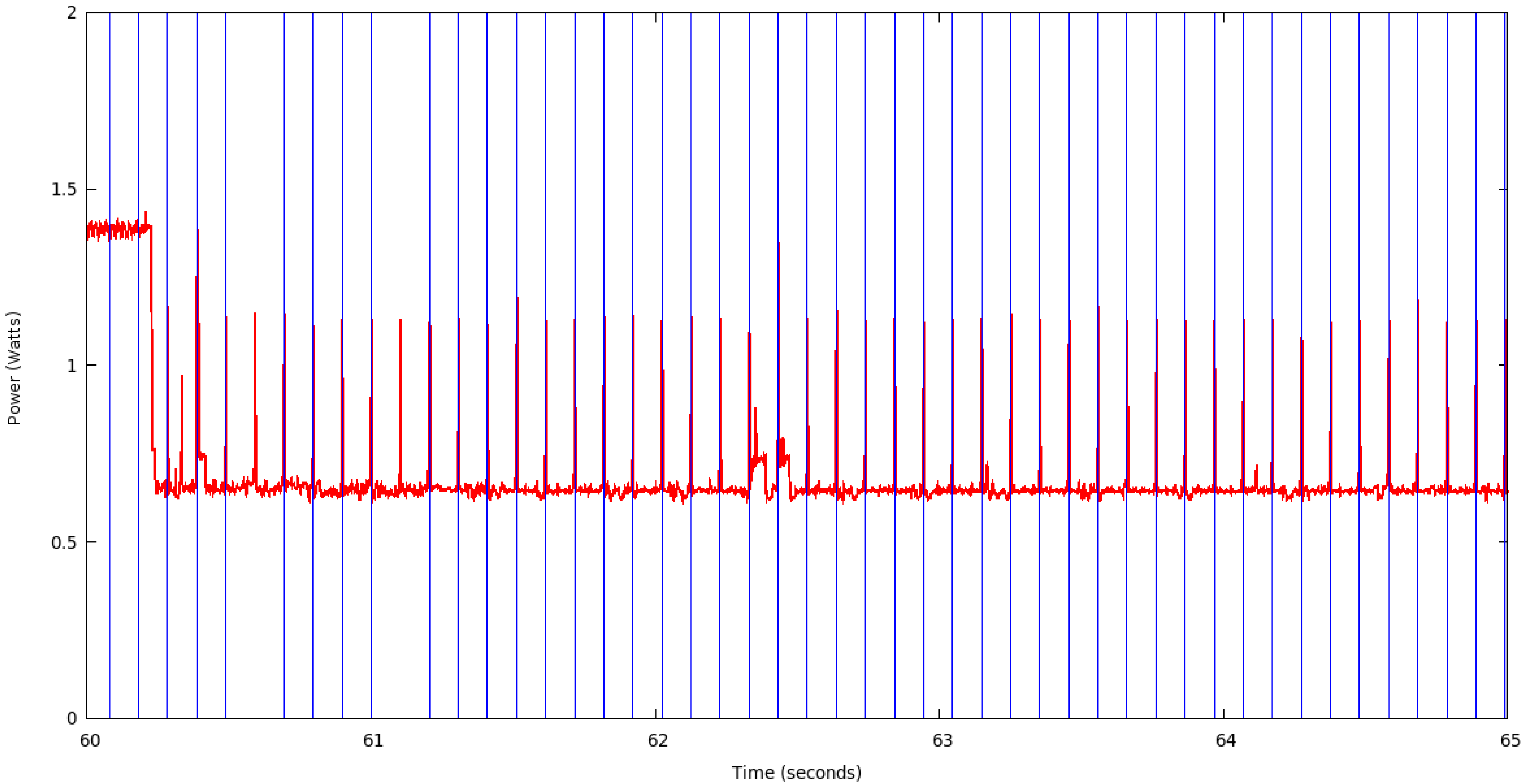
HTC G1 (or Magic), Android 1.1

Joining a wireless network



HTC G1 (or Magic), Android 1.1

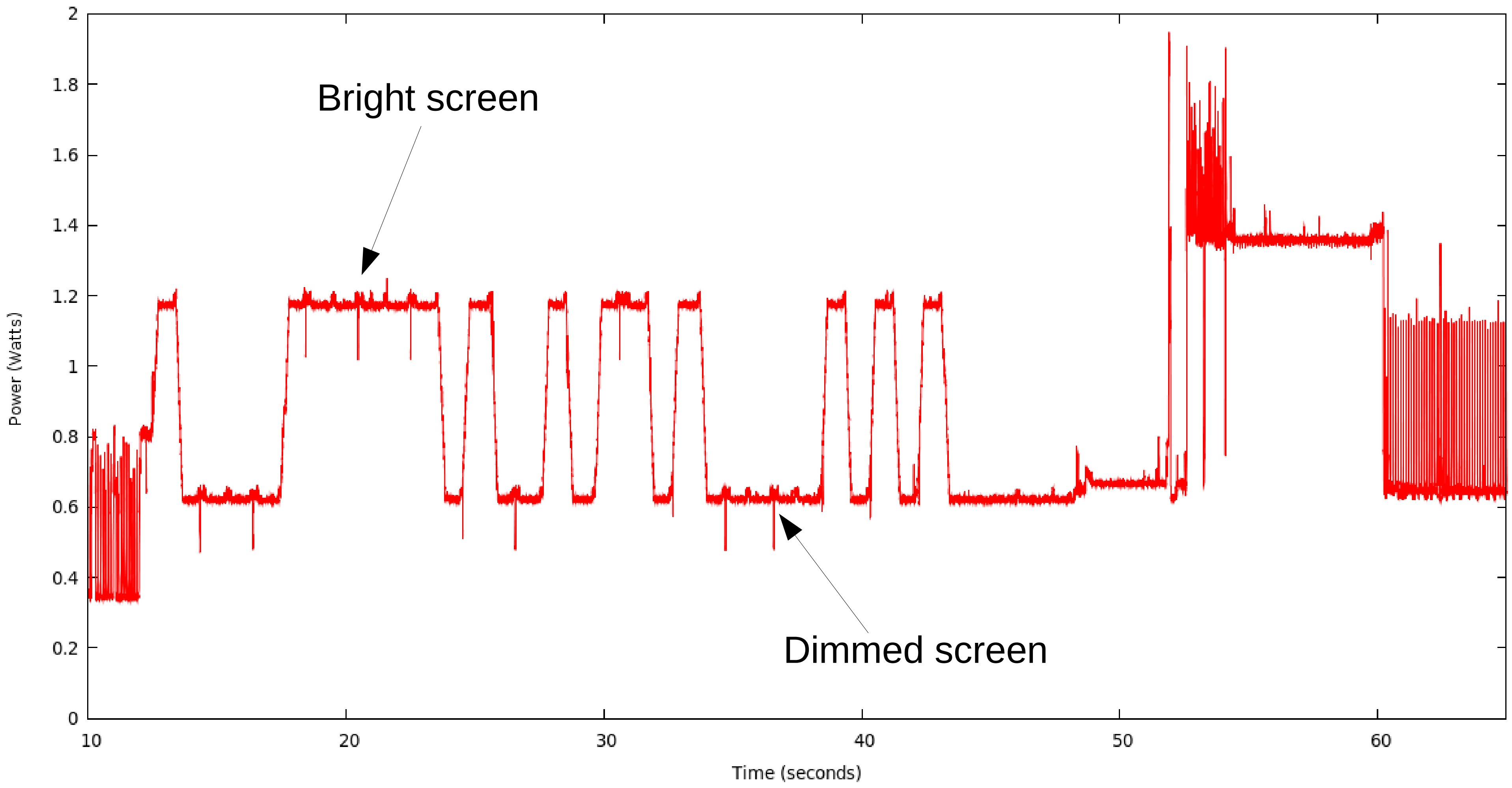
Access point beacons correlate with spikes in the power trace



HTC G1 (or Magic), Android 1.1

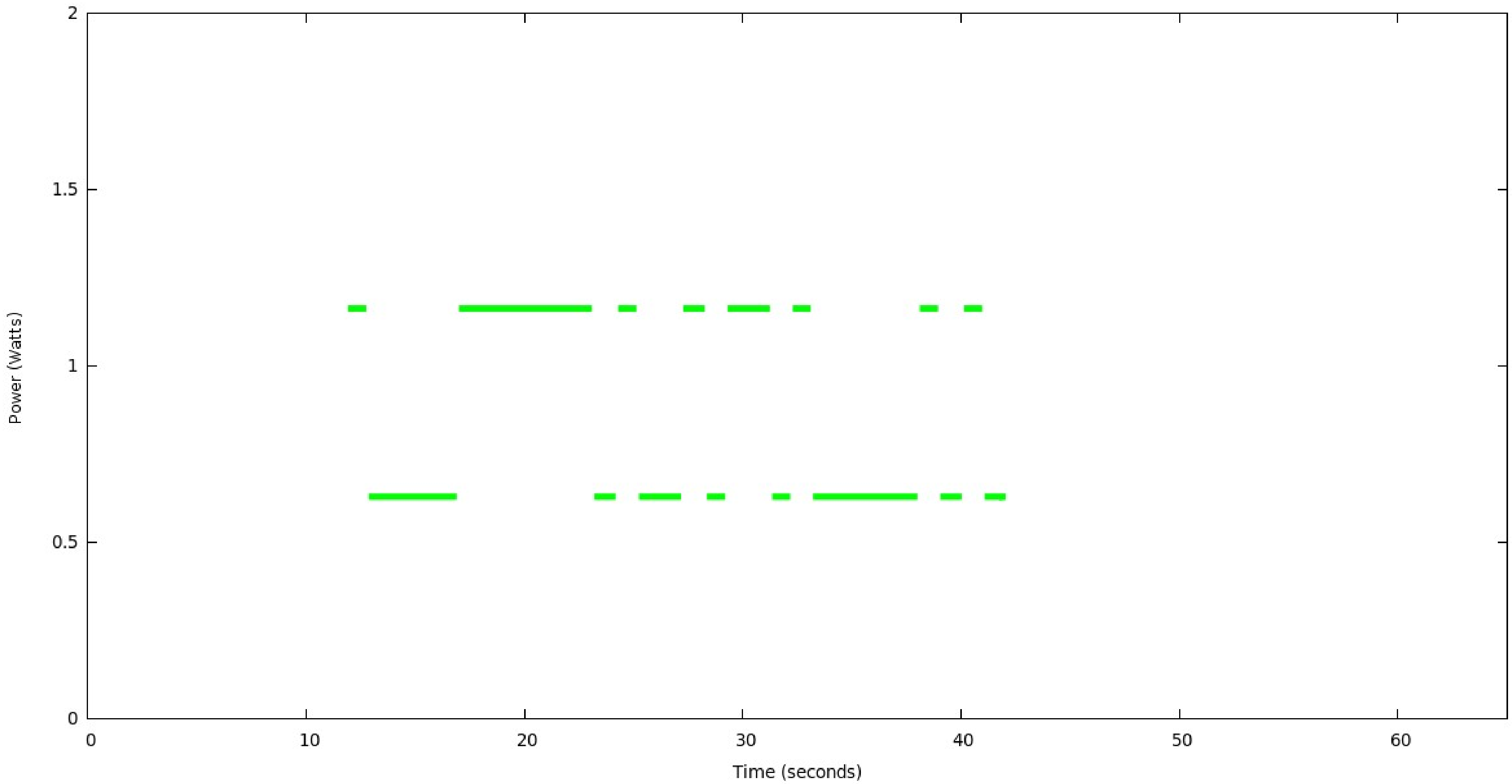
Timestamped events from the phone must be aligned with the appropriate sample points

The synchronization information is embedded in power trace



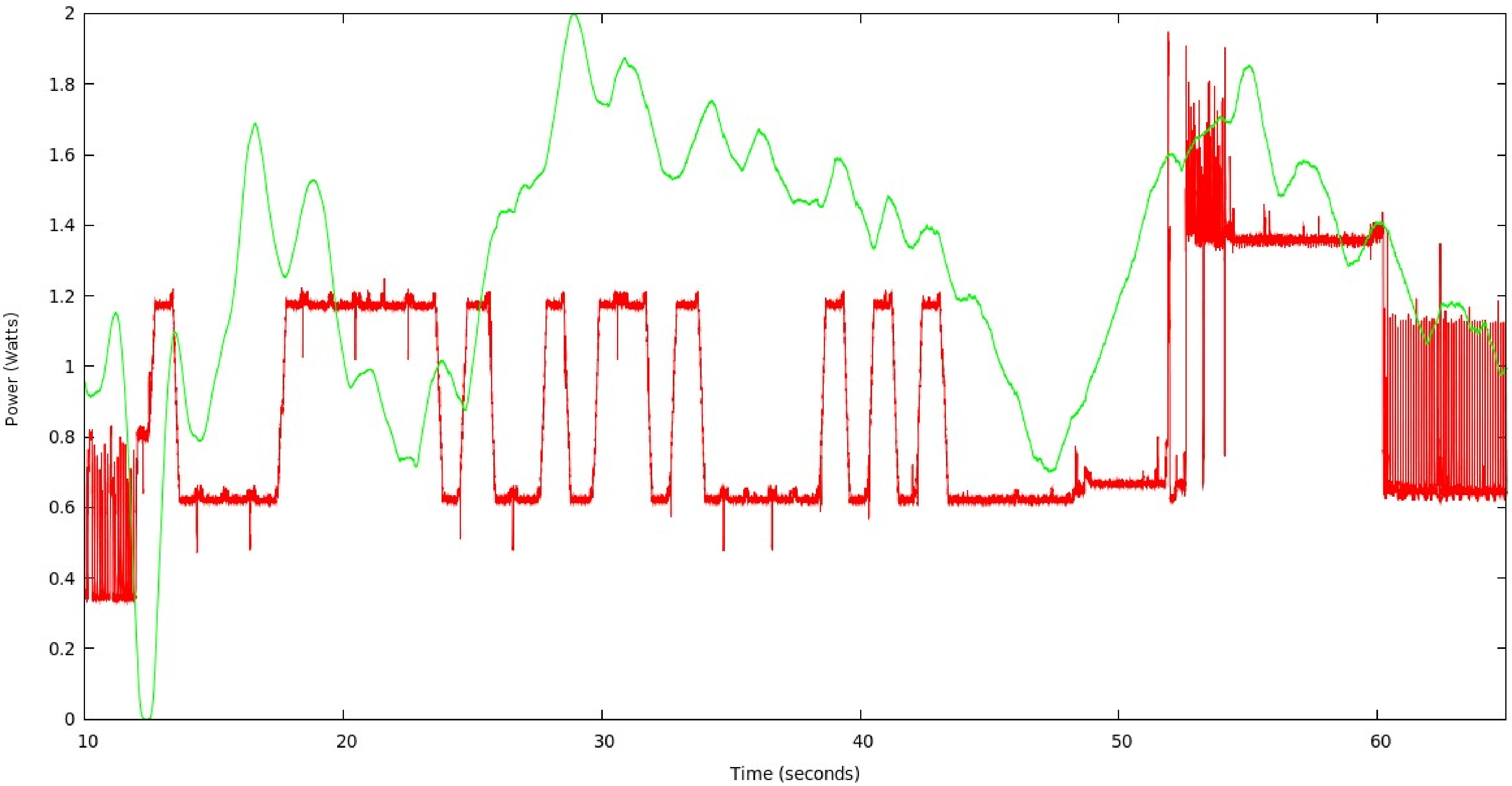
HTC G1 (or Magic), Android 1.1

Hypothesise matching pulses



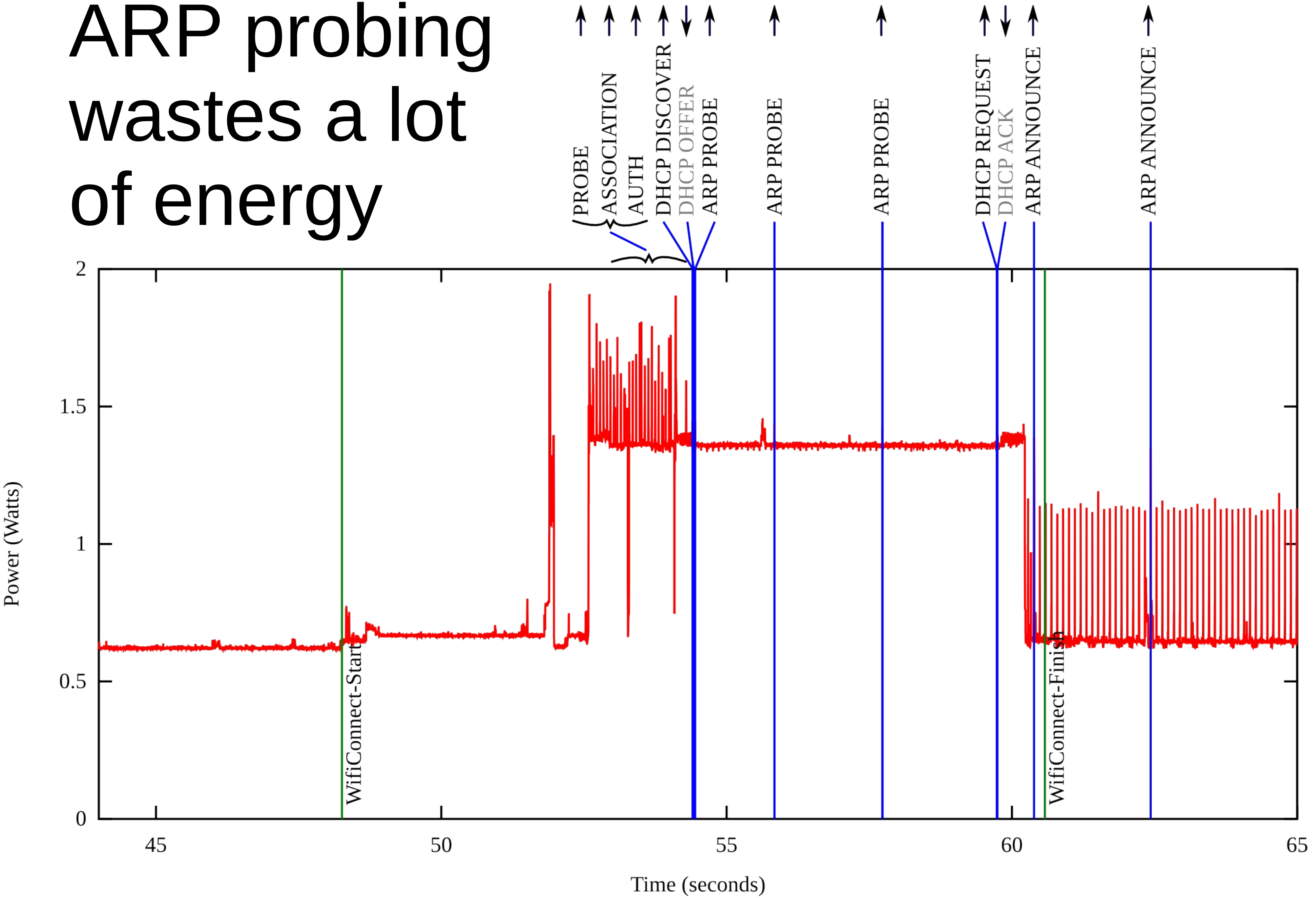
HTC G1 (or Magic), Android 1.1

Find alignment from autocorrelation with a hypothesised signal



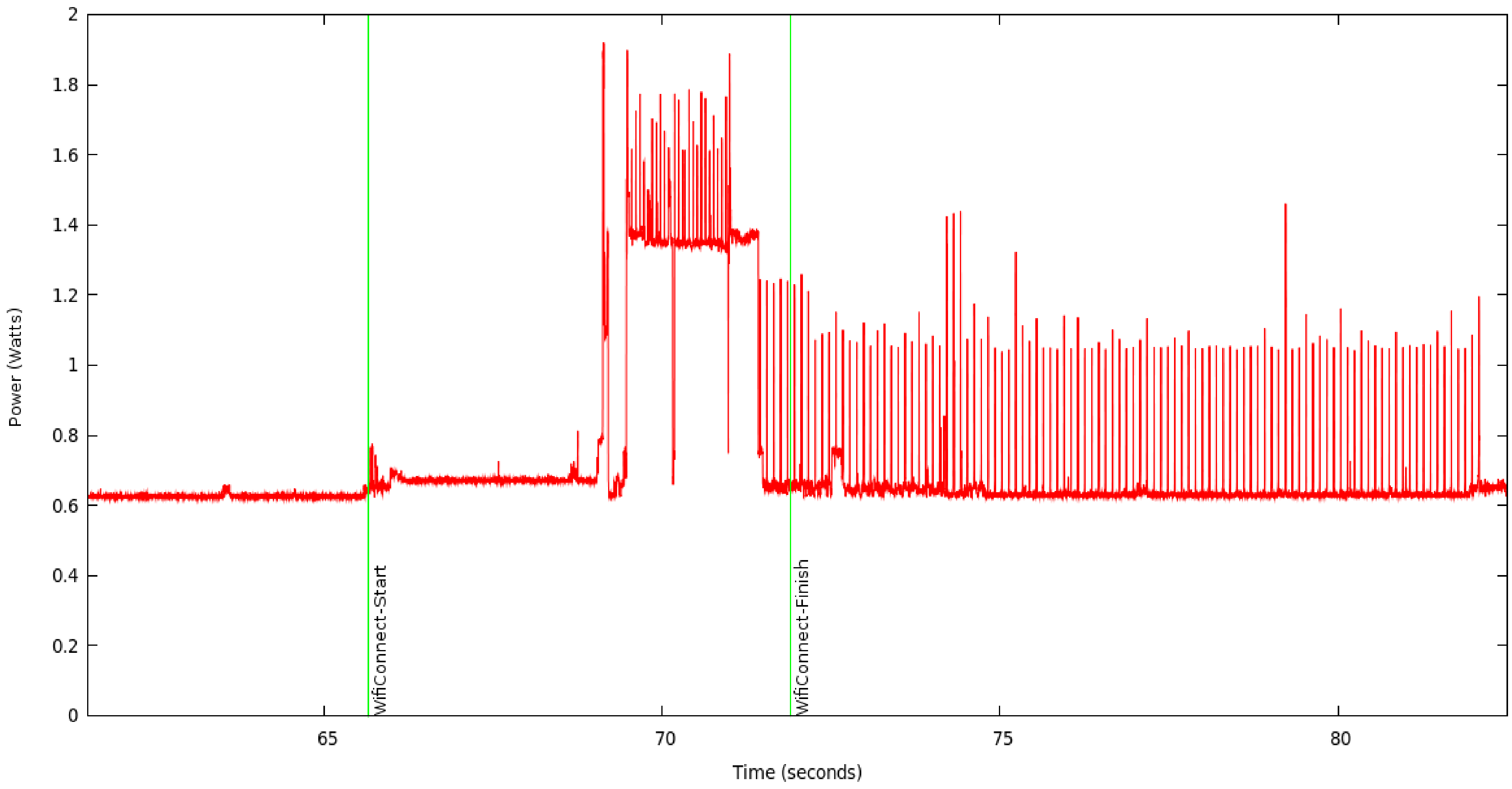
HTC G1 (or Magic), Android 1.1

ARP probing wastes a lot of energy



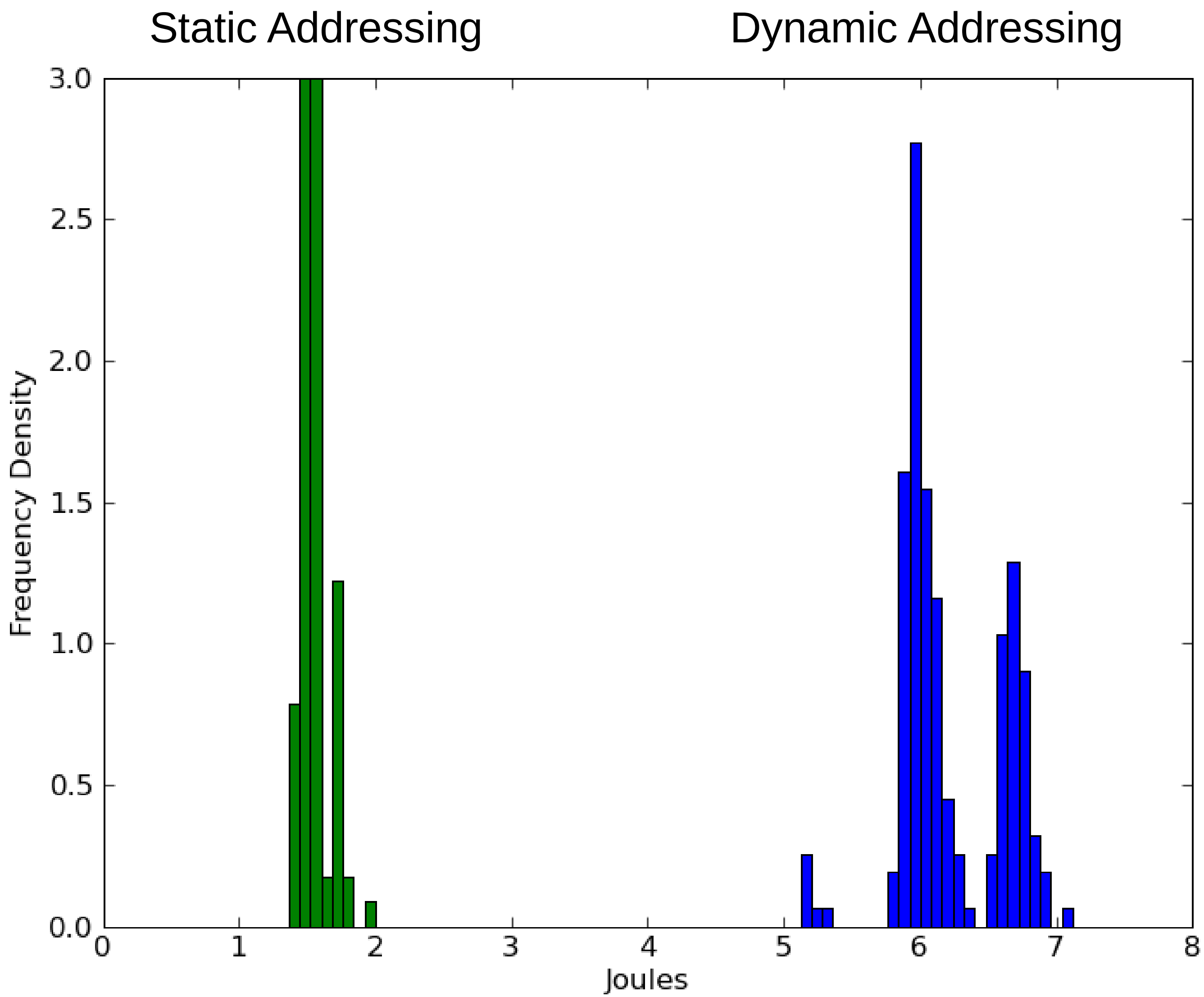
HTC G1 (or Magic), Android 1.1

Remove the DHCP overhead by using static addressing



HTC G1 (or Magic), Android 1.1

Static addressing reduces the connection cost to 1.5 Joules



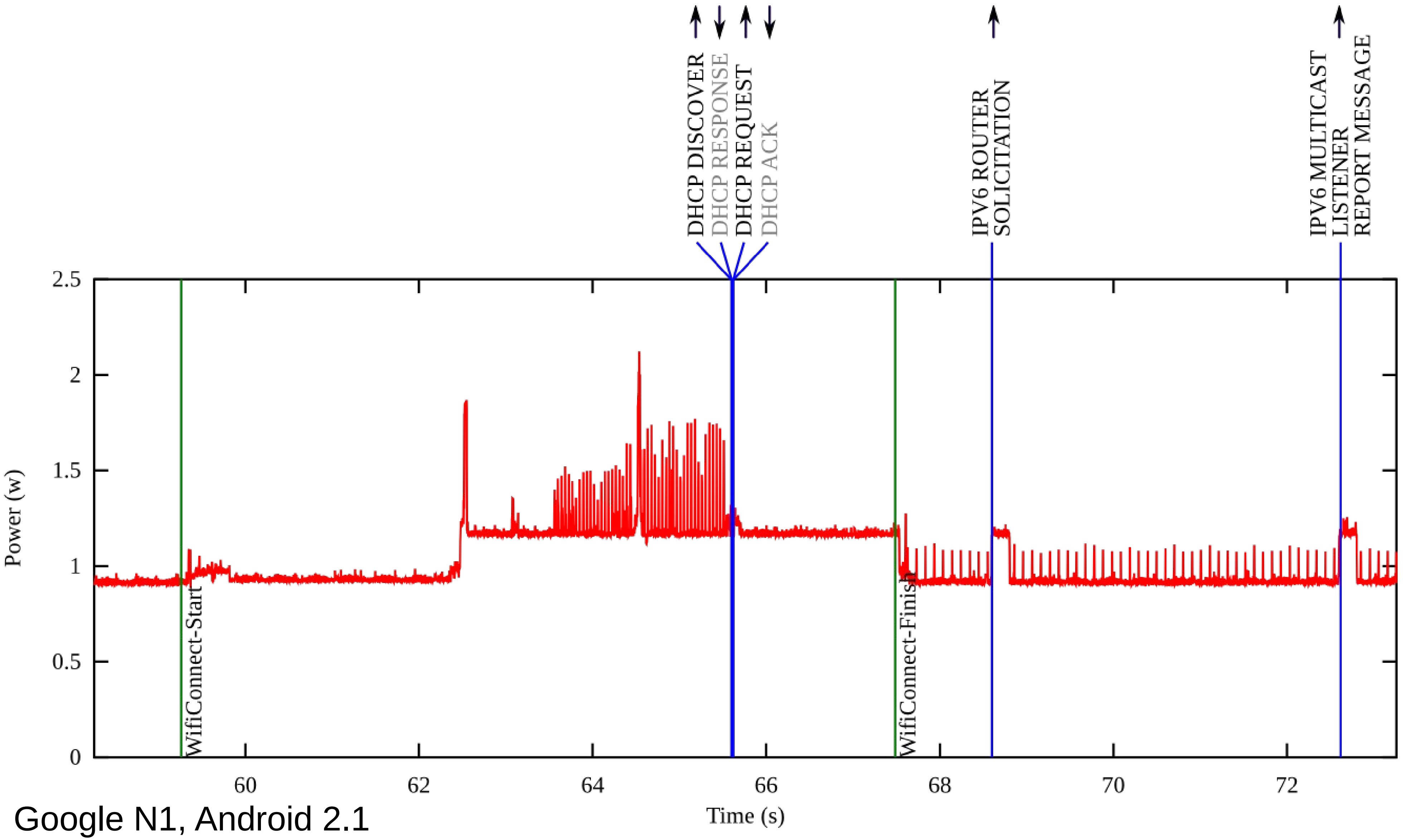
HTC G1 (or Magic), Android 1.1, Static = 143 trials, Dynamic = 194 trials

We could remove the ARP probes from our client implementation

RFC2131 “...the client SHOULD probe the newly received address, e.g., with ARP.”

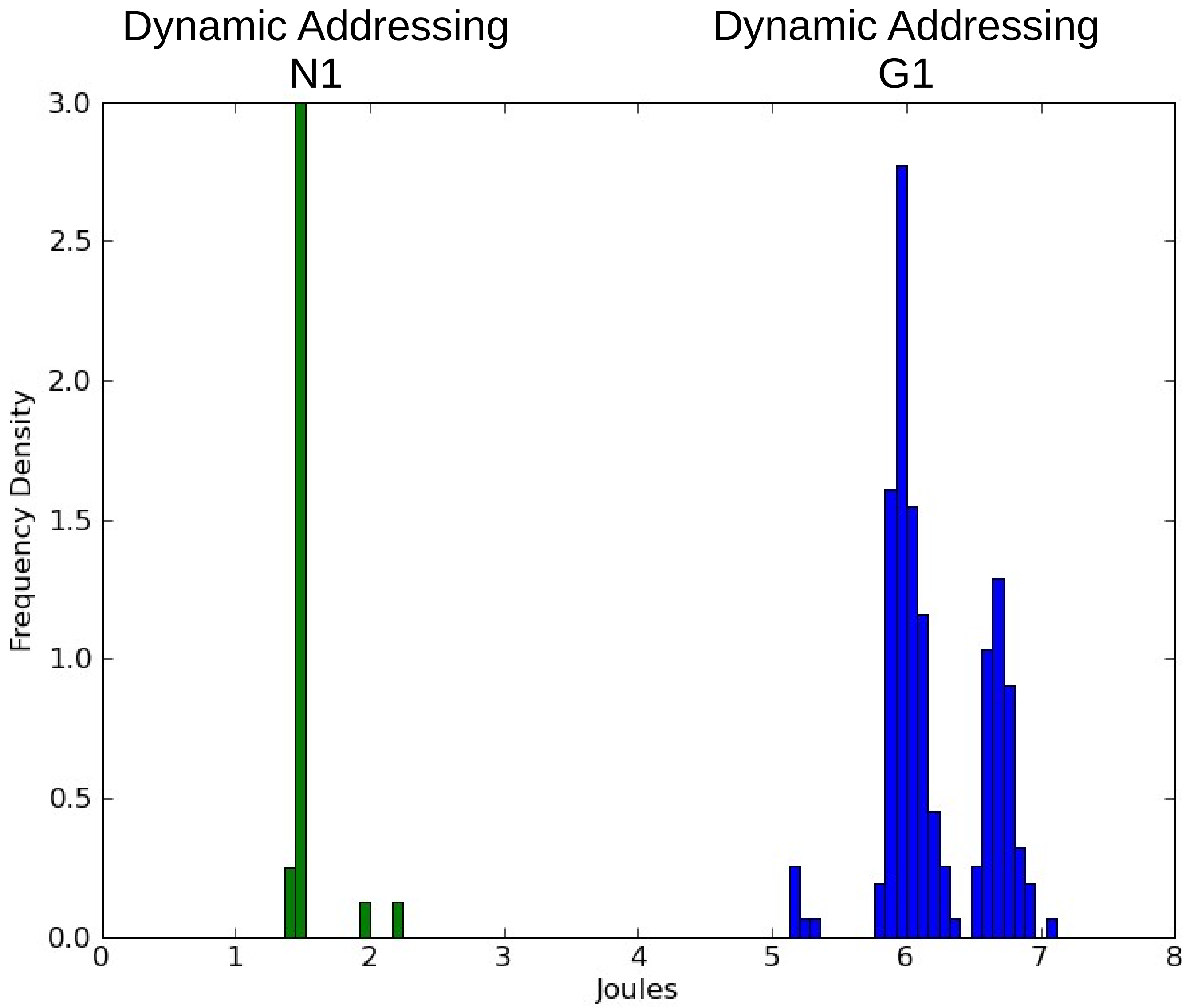
RFC2119 – SHOULD “...there may exist valid reasons in particular circumstances to ignore a particular item”

Android 2.1 doesn't ARP probe in our tests



Google N1, Android 2.1

Dynamic addressing now costs 1.5J



Google N1, Android 2.1, 100 trials / HTC G1 (or Magic), Android 1.1, 194 trials

How much energy is 5 Joules?

- 5 seconds of talk time
- 8 minutes of standby time
- 3.5 minutes of idle wireless (the extra cost of having the wireless on is approx. 0.024W)

Knowing the connection cost helps with system design

- How long should the wireless stay active whilst idle?
  - 6J connection → 250 seconds idle cost
  - 1.5J connection → 62 seconds idle cost
- Is it worth forcing programmers to tell the system explicitly?

Its not clear whether its worth the effort to apply these optimisations

- Wifi connection – should we change the API to get more detail of an application's intent?
- Sending data – should we change the operating system to support packet level co-scheduling?
- Changes to API are costly
  - To implement
  - To migrate existing applications

We are attempting to build a substantive dataset of smart-phone use

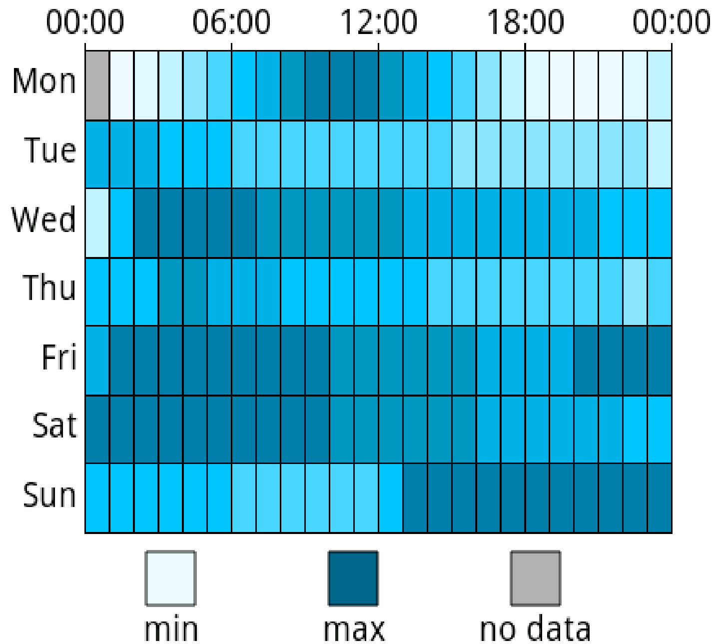
PhD work by Daniel Wagner

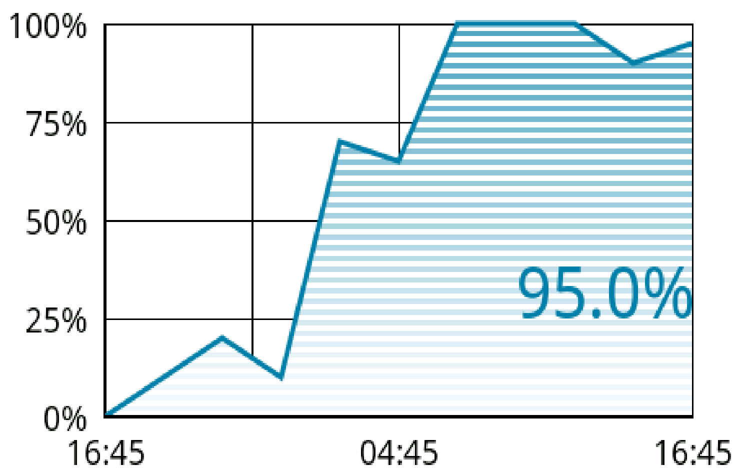
We collect everything...

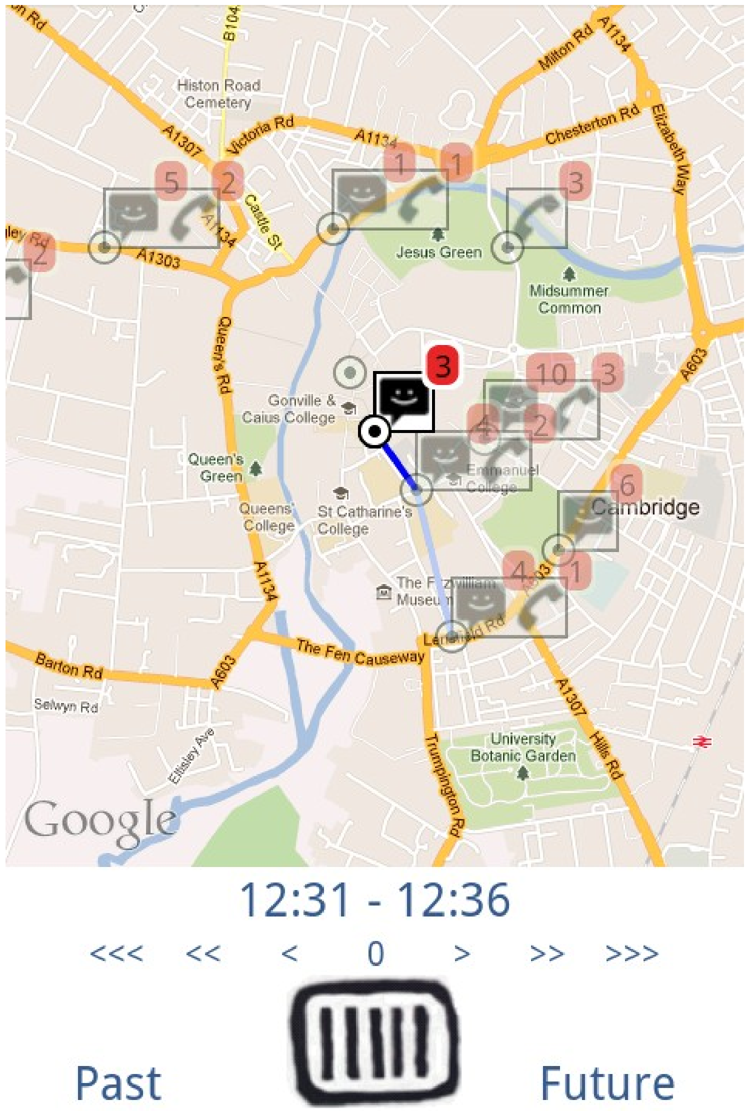
Handset: on/off, OS version, device type
Screen: on/off, brightness
Storage: size/free/type
Telephony: ringer/mode/roaming/sigstrength/data
Tel events: calls/text/mms/data
Battery: charging/voltage/level
Wifi: connects/scans/data
Bluetooth: connects/scans/data
Apps: source/running/resource use

Some of these require polling

More features coming over the summer





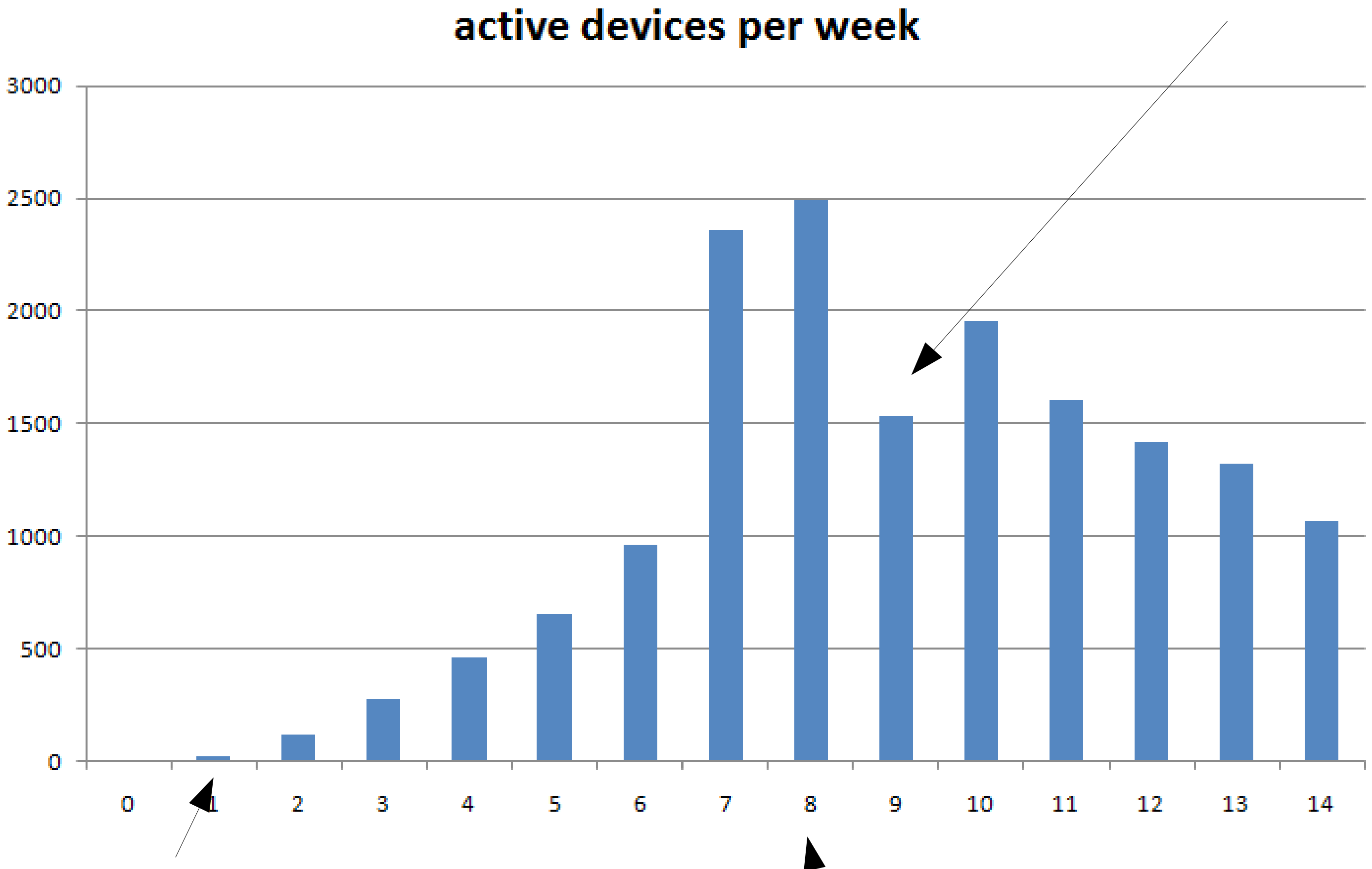


We remove direct identifiers from trace

- Your contacts each get a unique pseudonym
- This doesn't give you anonymity
- You can assign a readable name for your use
- We will only release data which is at least 3 months old → you can opt out retroactively
- Pause functionality available

Current progress (6-Aug-2011)

Server fail



Release date May 9th

Jun 19th Engadget

Implementation lessons... timestamps are not reliable

- Users manually change the time
  - Travelling, daylight saving
- Sometimes the OS reports invalid dates
  - e.g. after an update for some reason
- How do network corrections get applied?
- Solution: record phone uptime and insert real-time clock events to anchor it

Users are highly sensitive to the size of your application

- Consider effective methods of minimizing size
- Android sorts by size – don't be the biggest!

Please install Device Analyzer
and/or
Please tell us if you have concerns

http://deviceanalyzer.cl.cam.ac.uk

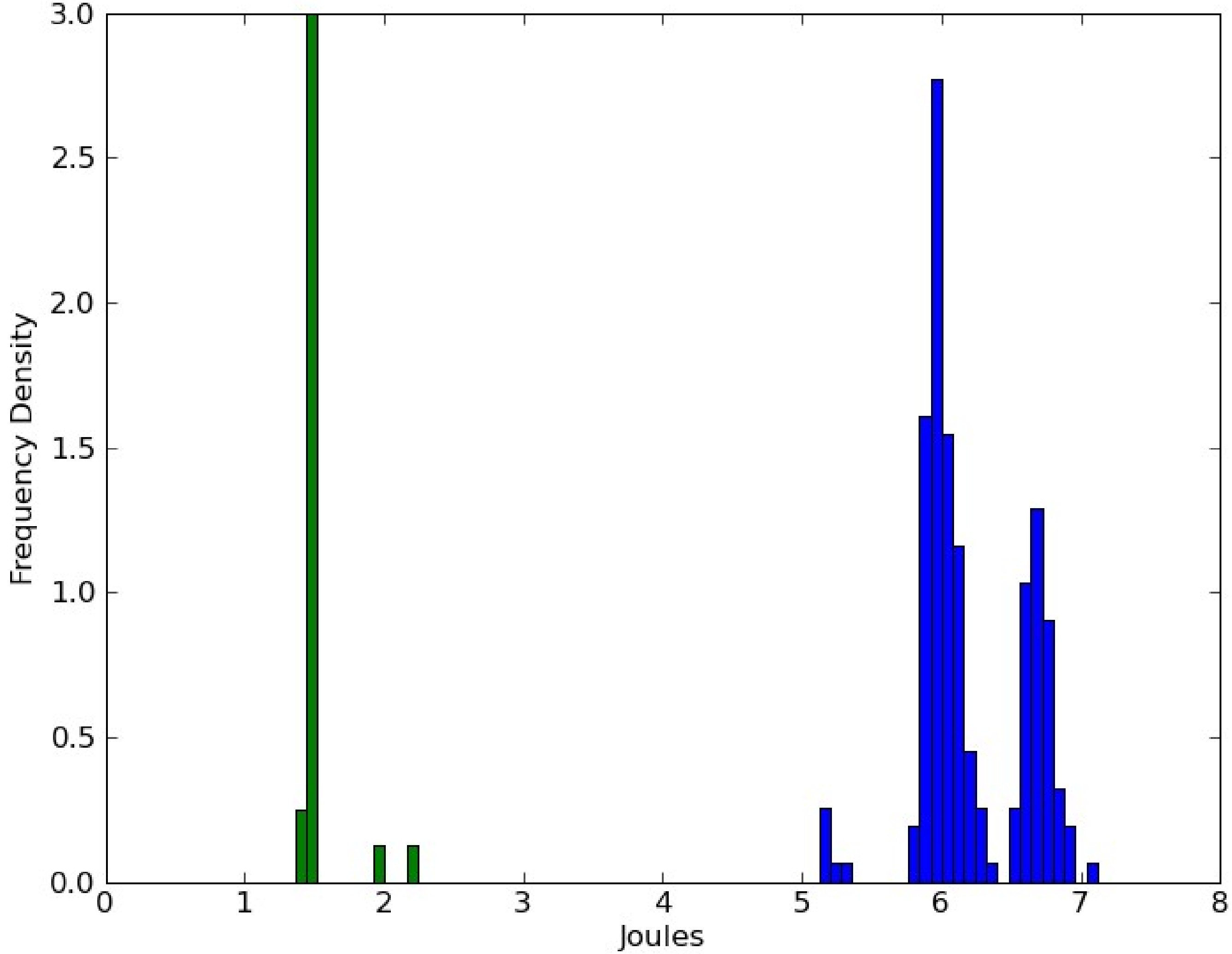
Or search for Device Analyzer by dtg-android on the Android Market

Thanks to
Daniel Wagner, Andy Hopper,
Alastair Beresford, Simon Hay,
Google & Qualcomm

The distribution for the G1 phone splits into 3 parts

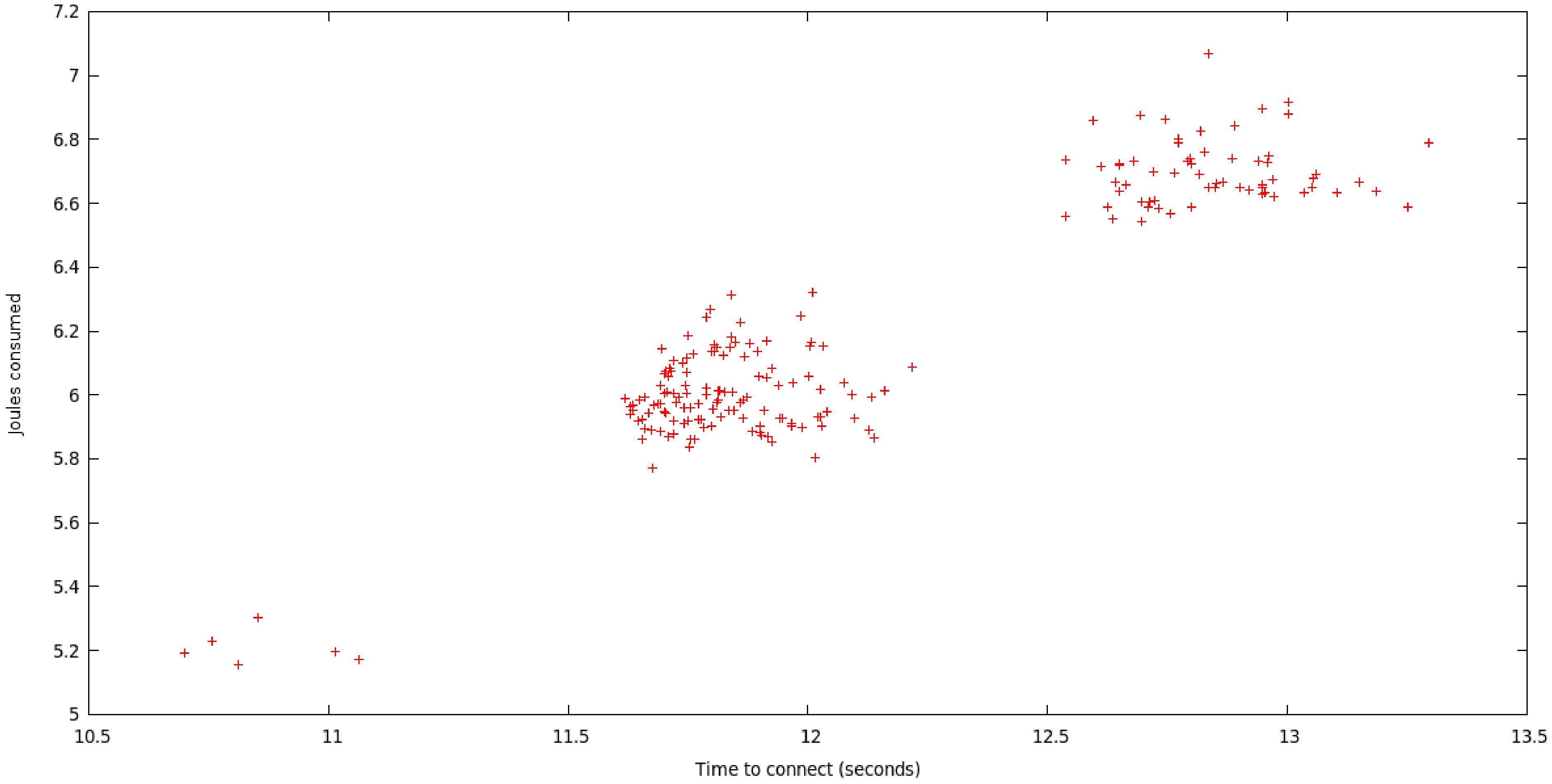
Dynamic Addressing
N1

Dynamic Addressing
G1



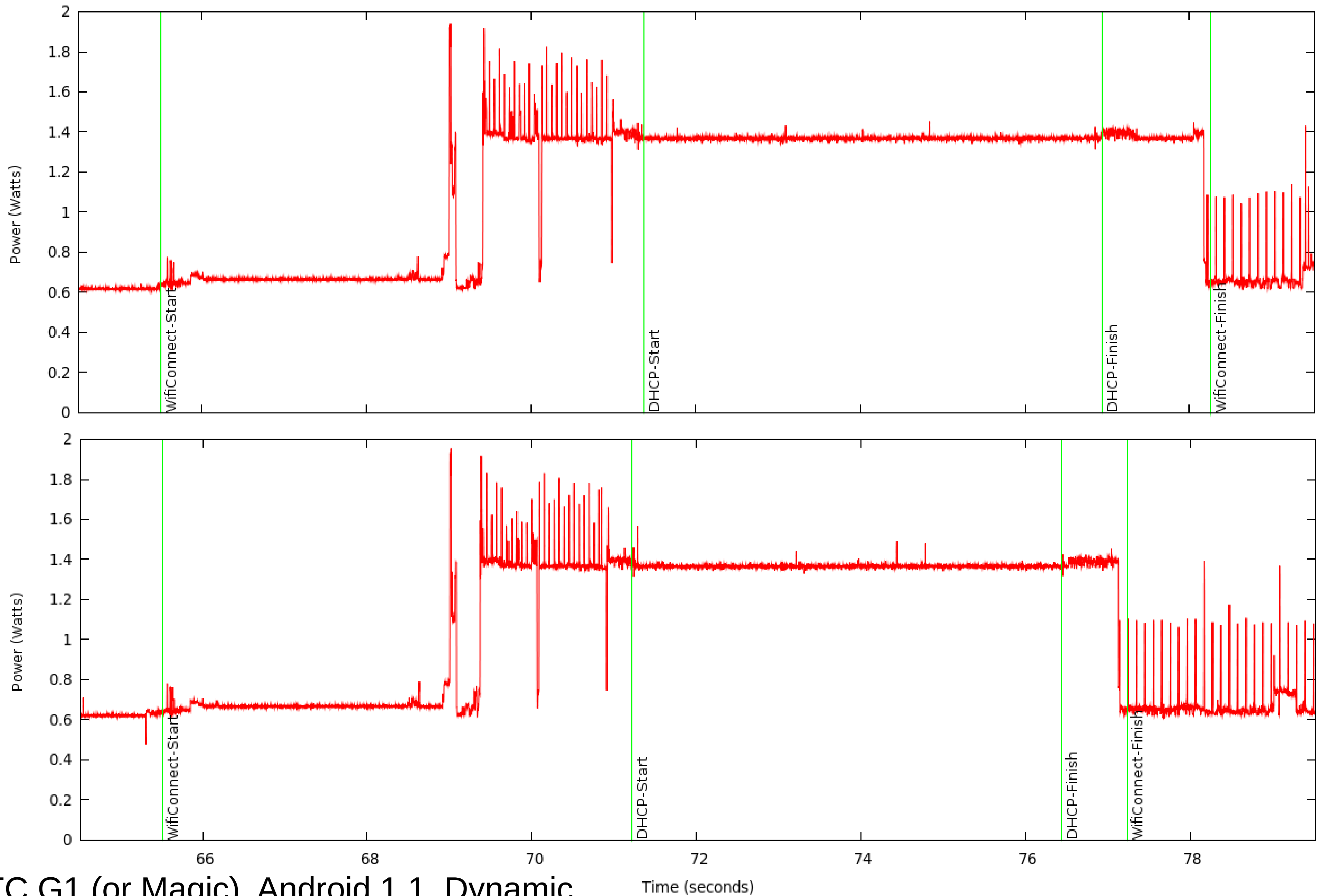
Google N1, Android 2.1, 100 trials / HTC G1 (or Magic), Android 1.1, 194 trials

The G1 histogram peaks are due to discontinuities in connection time



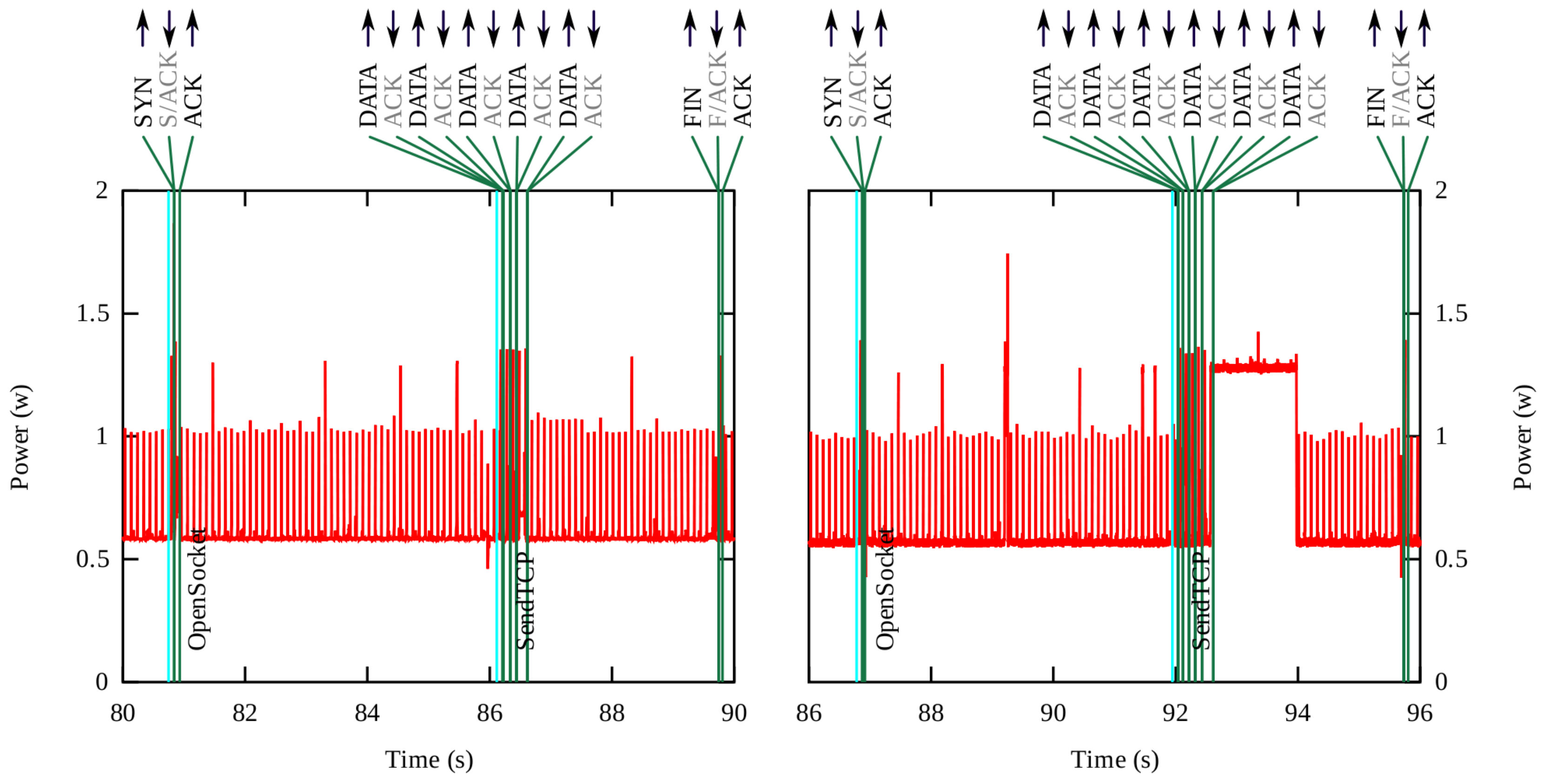
HTC G1 (or Magic), Android 1.1, Dynamic

Caused by power control in radio?



HTC G1 (or Magic), Android 1.1, Dynamic

This power control is evident when sending data too

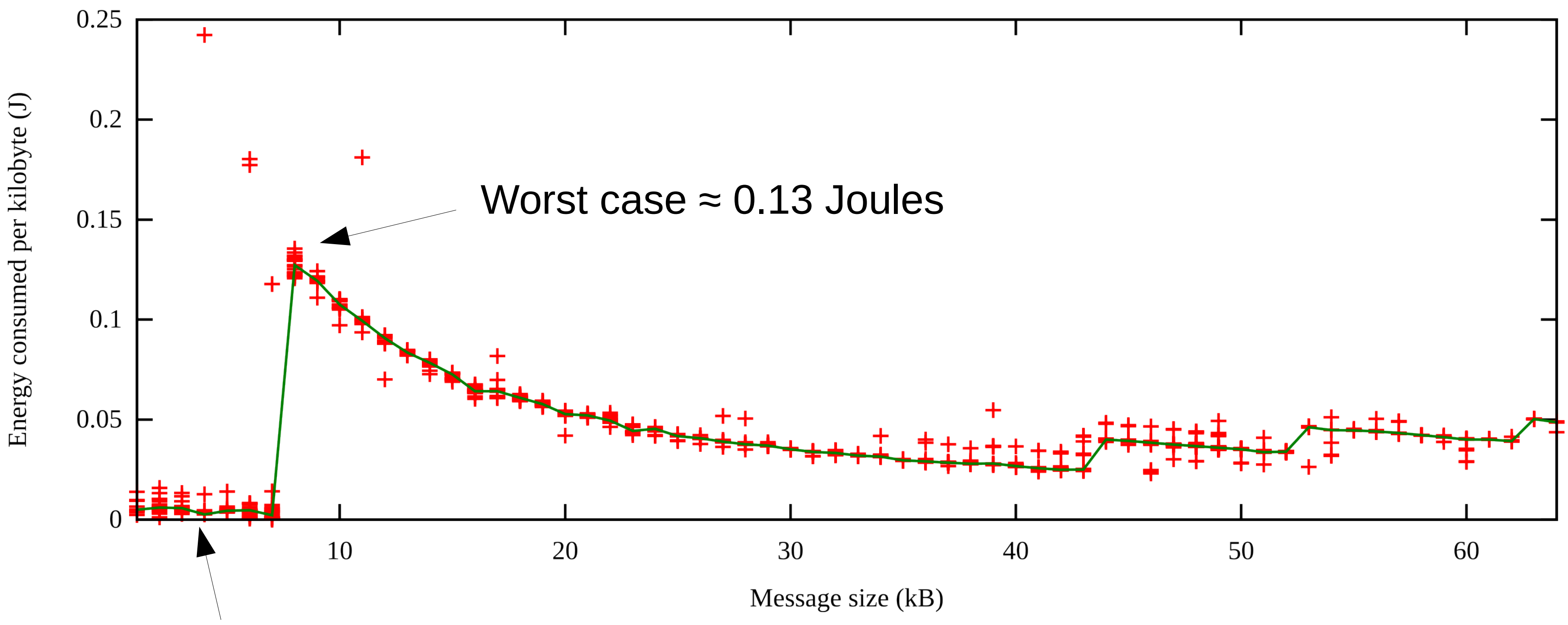


Send 7K of data over TCP

Send 8K of data over TCP

HTC G1 (or Magic), Android 1.1

This effect has a big impact on energy cost



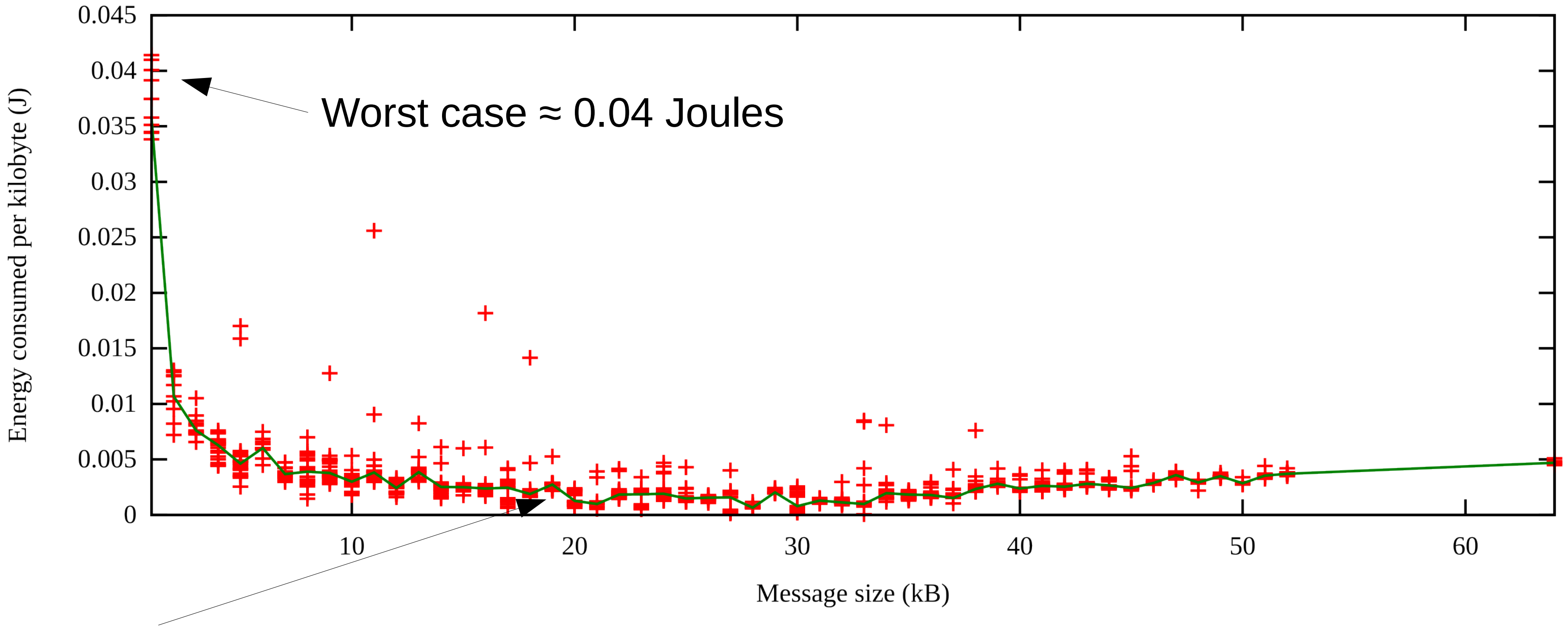
Best case ≈ 0.005 Joules

HTC G1 (or Magic), Android 1.1, 1120 Trials (HTC Hero, Android 1.5 is the same)

N1 energy performance

Best case: same

Worst case: much better



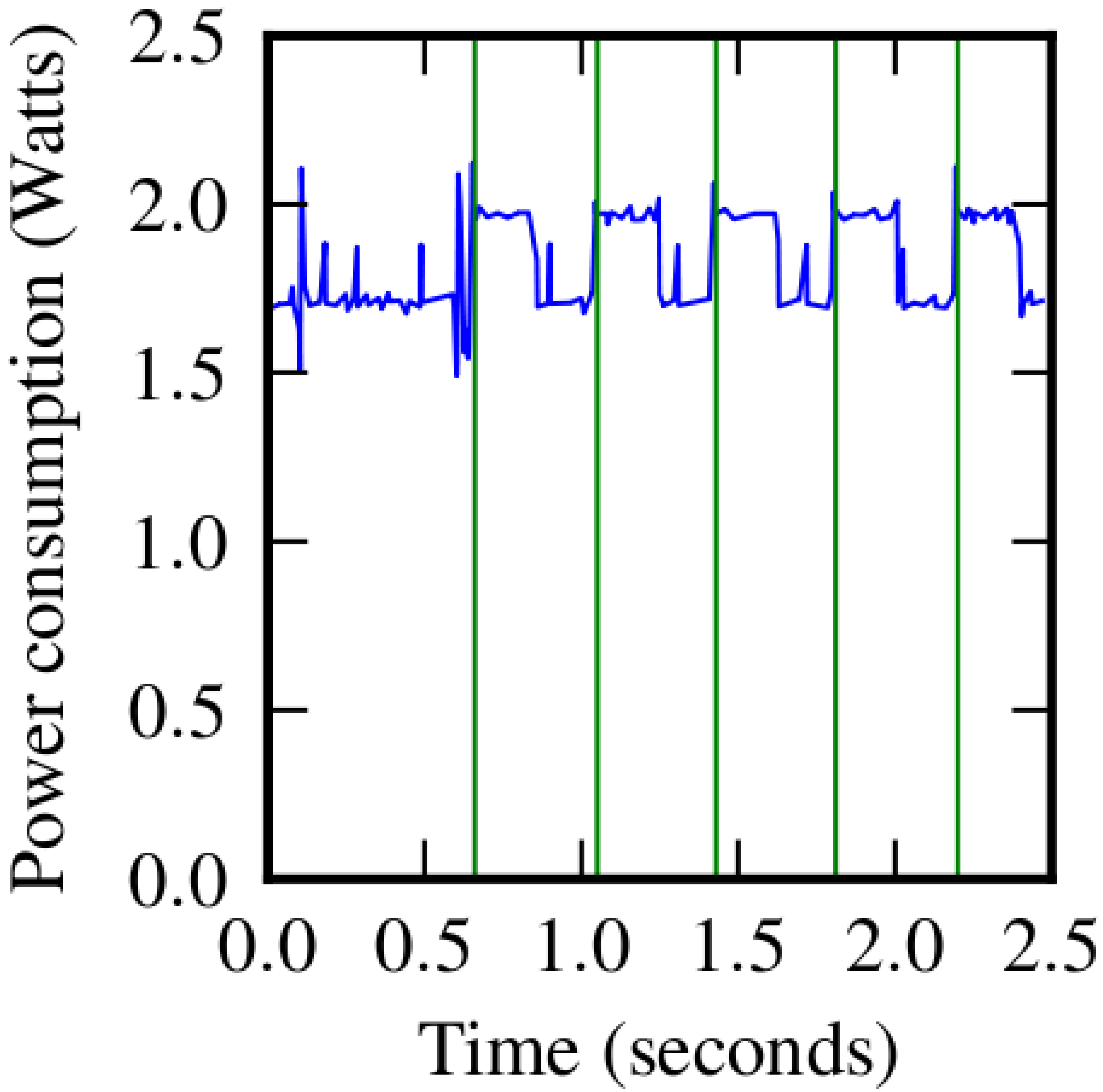
Best case ≈ 0.005 Joules

Google N1, Android 2.1, 900 Trials

Programmer should make a different choice depending on the platform

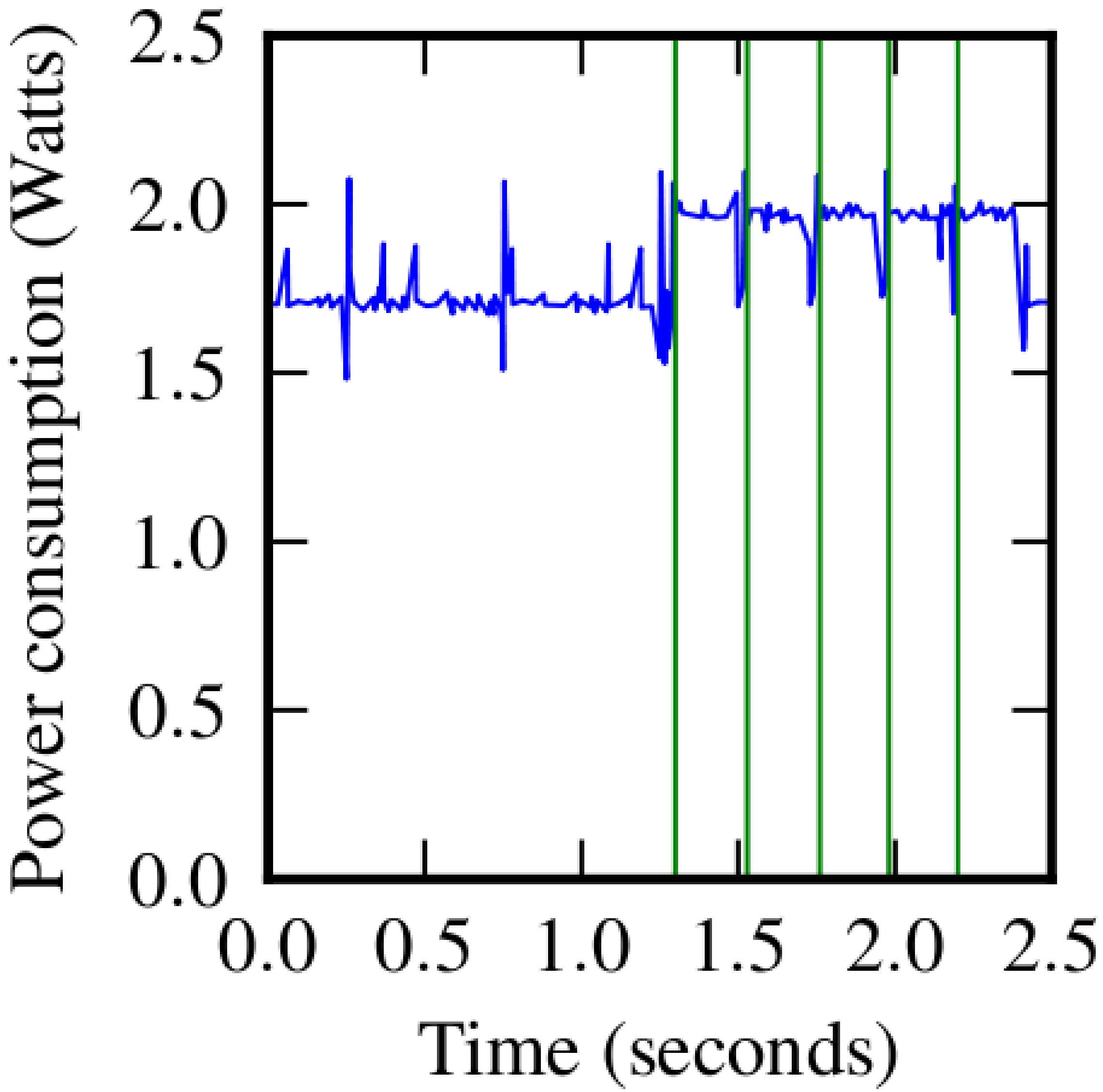
- Using a G1 => send 7k chunks
- Using a Nexus One => the larger the better
- We see unexpected behaviour in both graphs

Measure sending costs by sending UDP packets



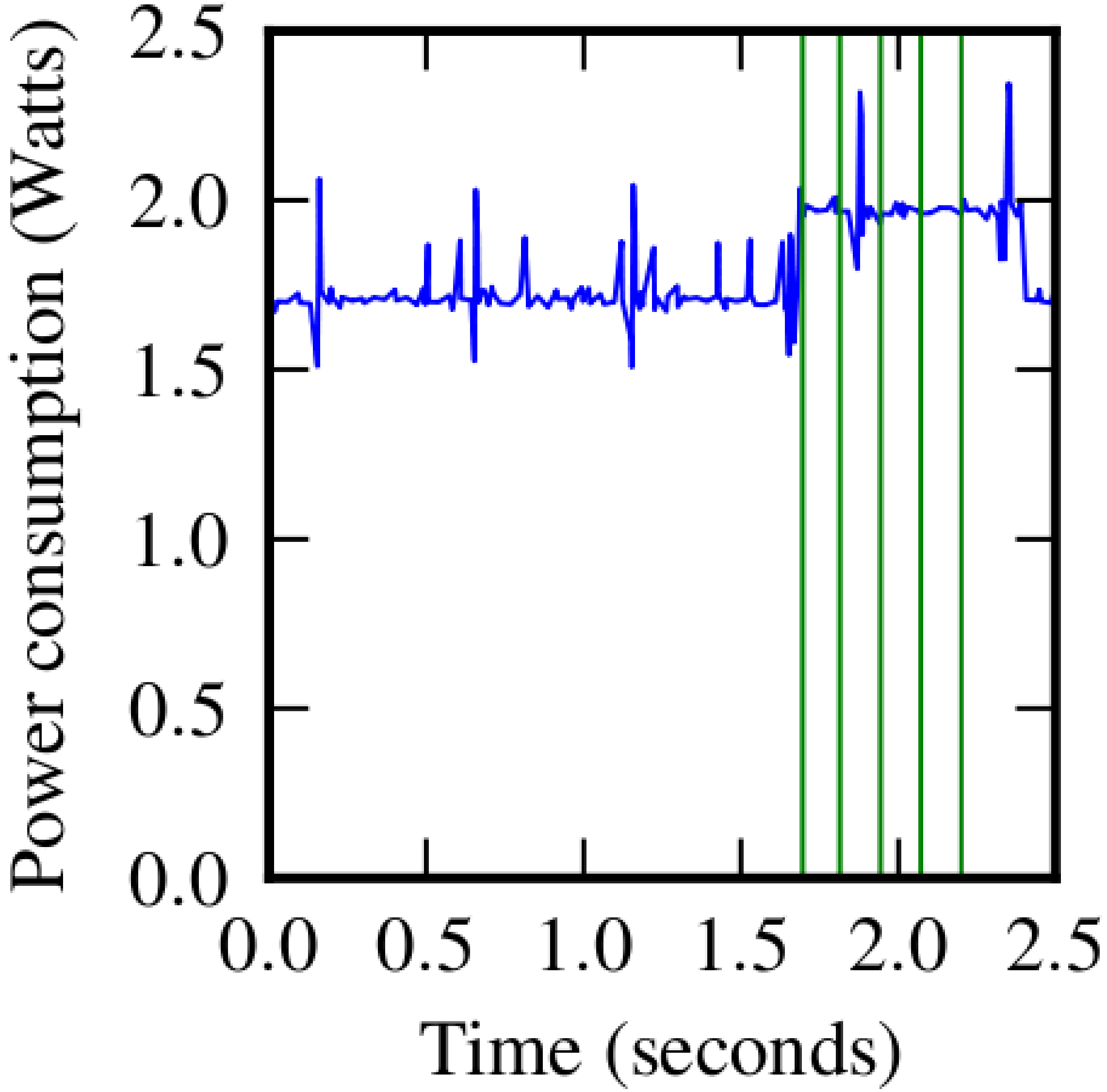
Nexus One
Send 4 packets
384ms interval
Android 2.2

Measure sending costs by sending UDP packets



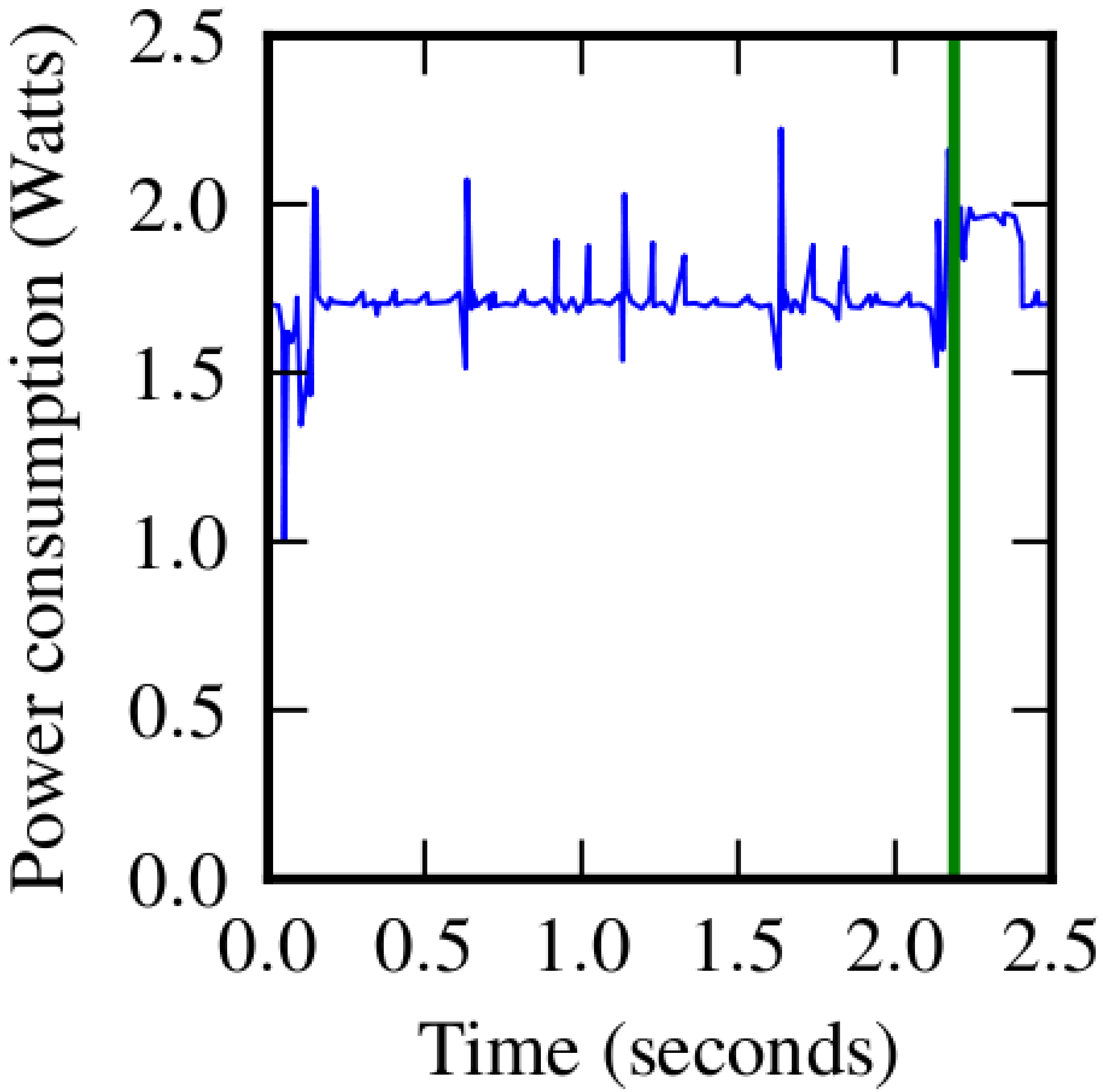
Nexus One
Send 4 packets
224ms interval
Android 2.2

Measure sending costs by sending UDP packets

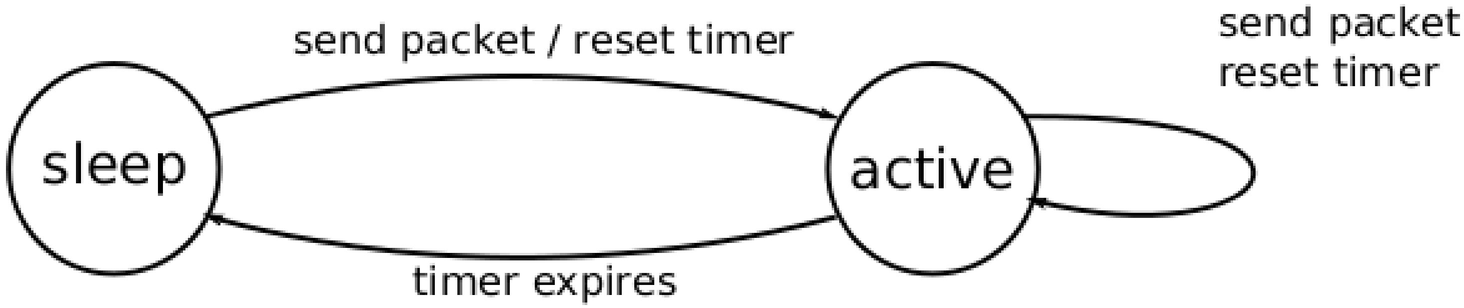


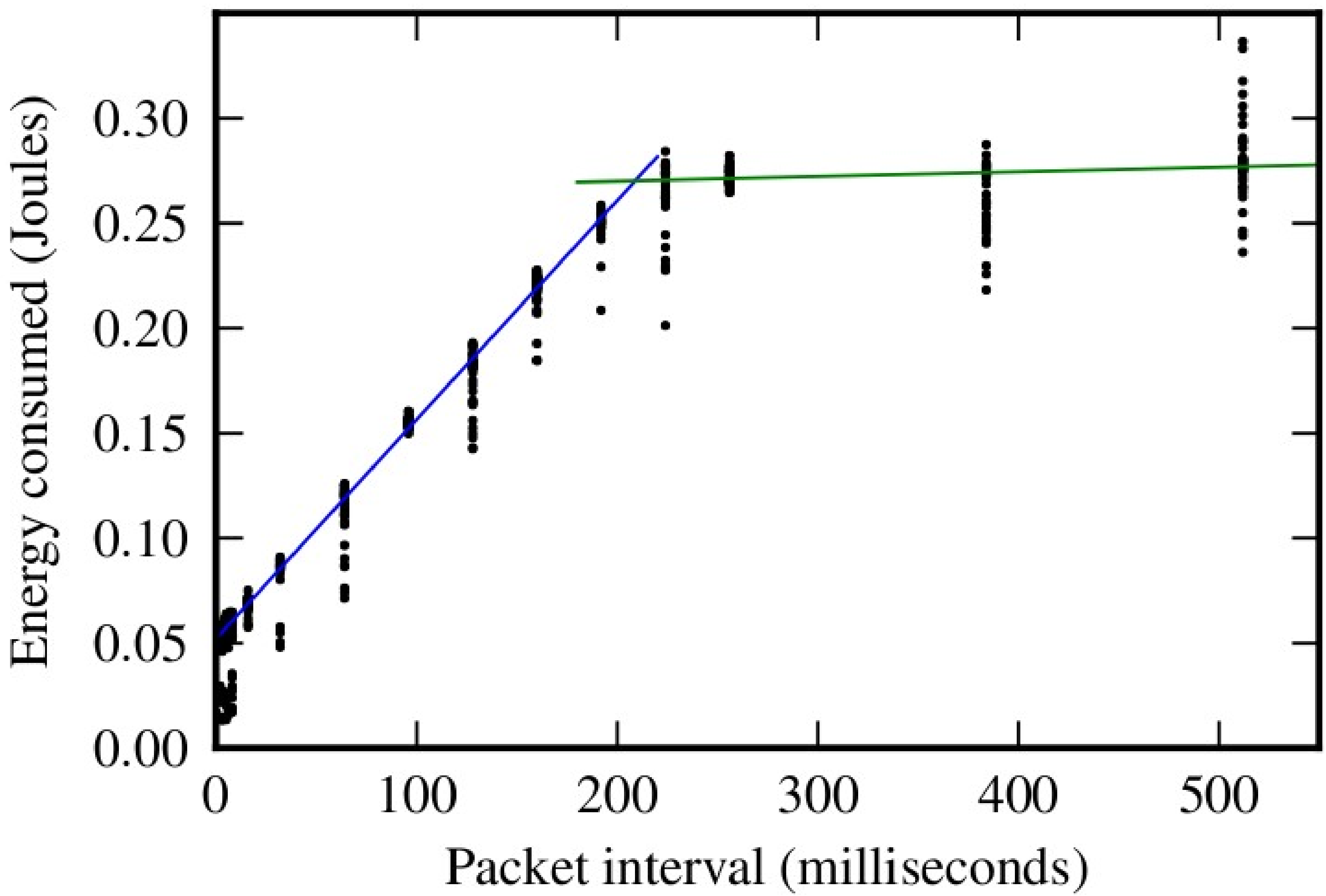
Nexus One
Send 4 packets
128ms interval
Android 2.2

Measure sending costs by sending UDP packets



Nexus One
Send 4 packets
8ms interval
Android 2.2

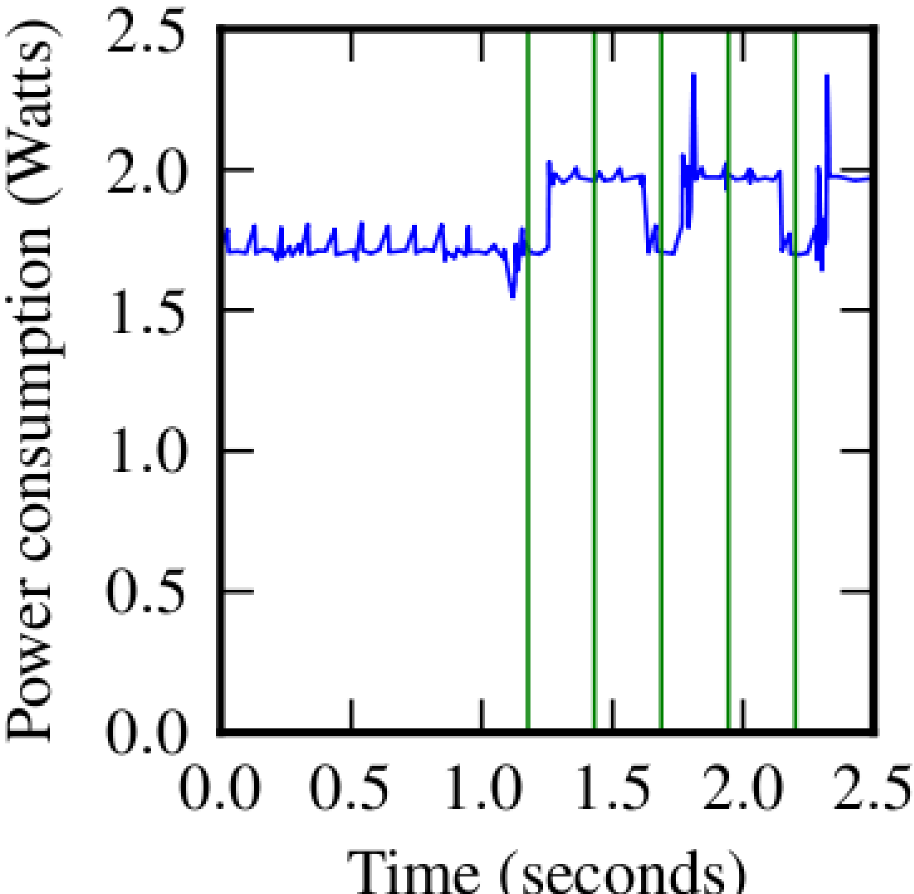

send packet / reset timer
send packet
reset timer
sleep
active
timer expires


Energy consumed (Joules)
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
100
200
300
400
500
Packet interval (milliseconds)

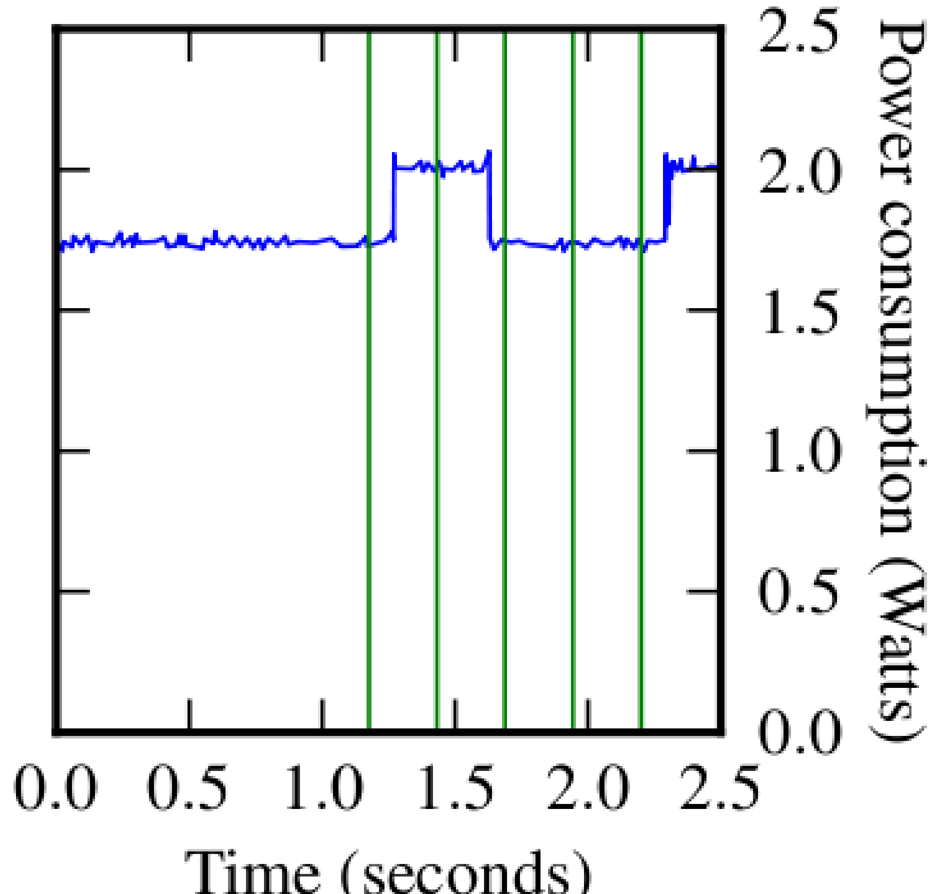
Co-scheduling packets between applications would save energy

- (Some) Applications already wait for opportunistic use of the network
- Operating system / library support needed to do better

TCP additionally needs to receive packets – more complex

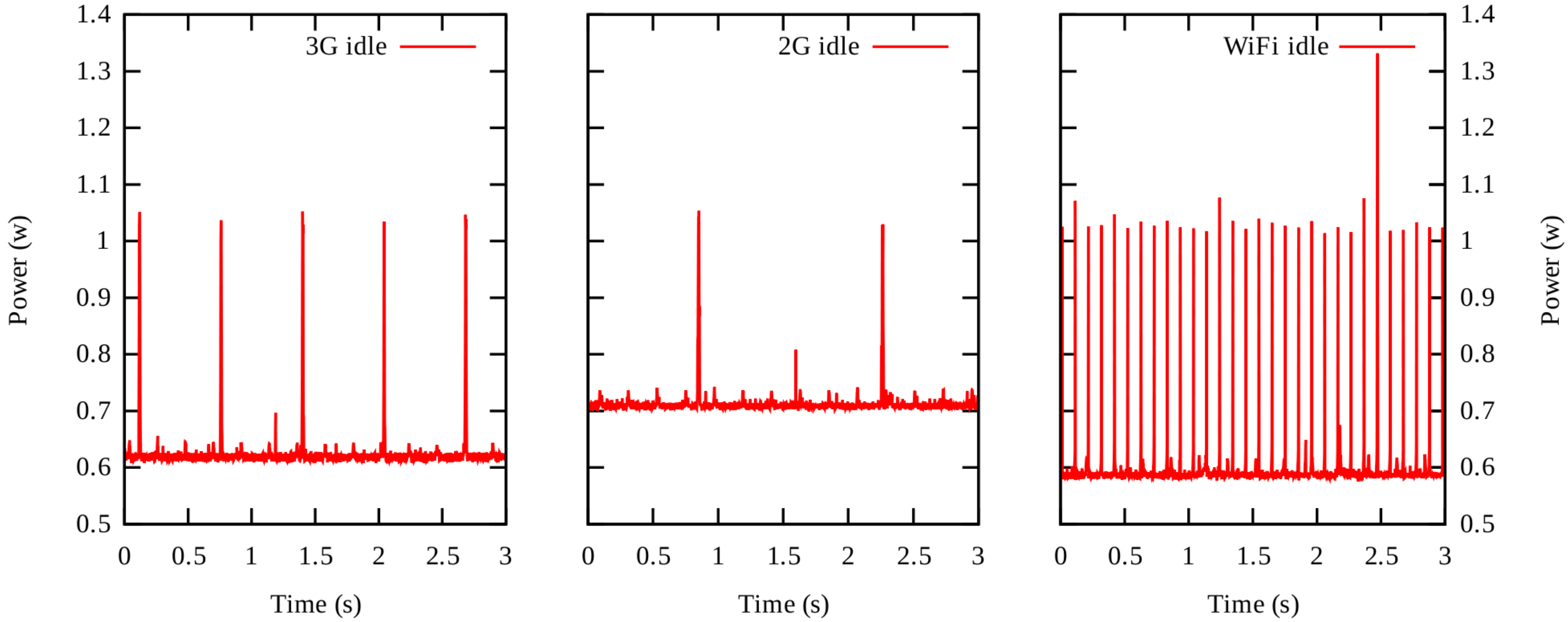


DTIM=1



DTIM=10

2G consumes more idle power than 3G (in my office)



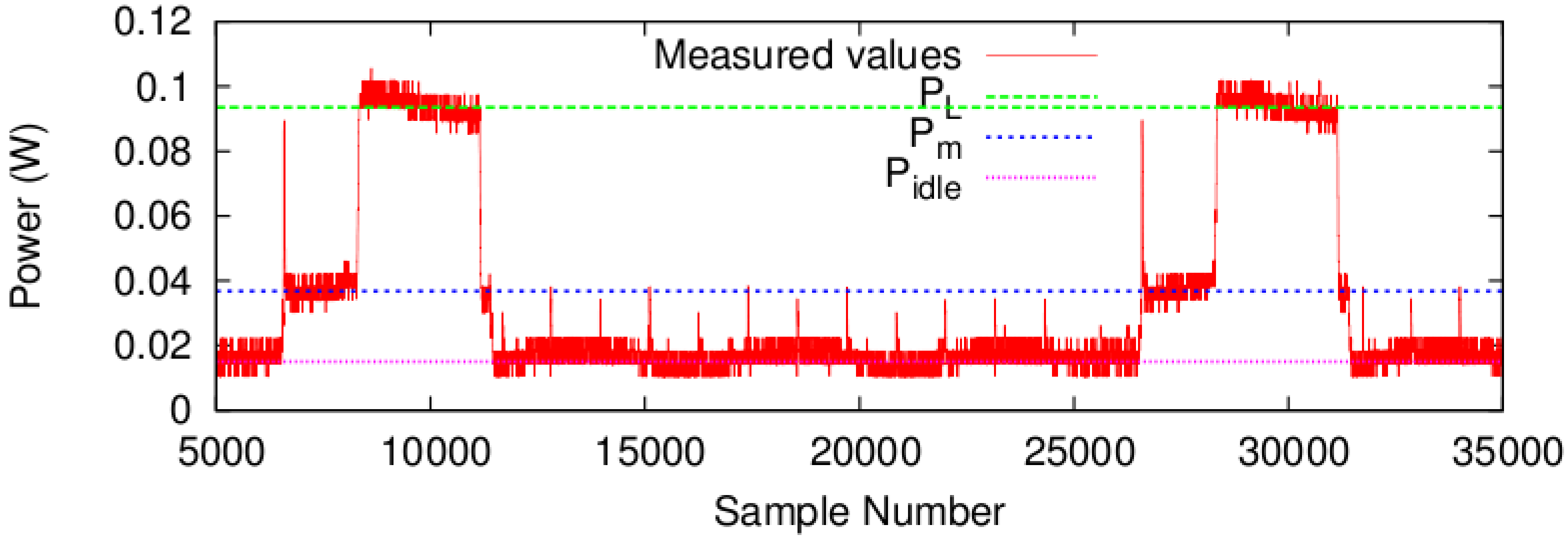
HTC G1 (or Magic) running Android 1.1

Bluetooth power consumption also shows this 'tail energy' effect

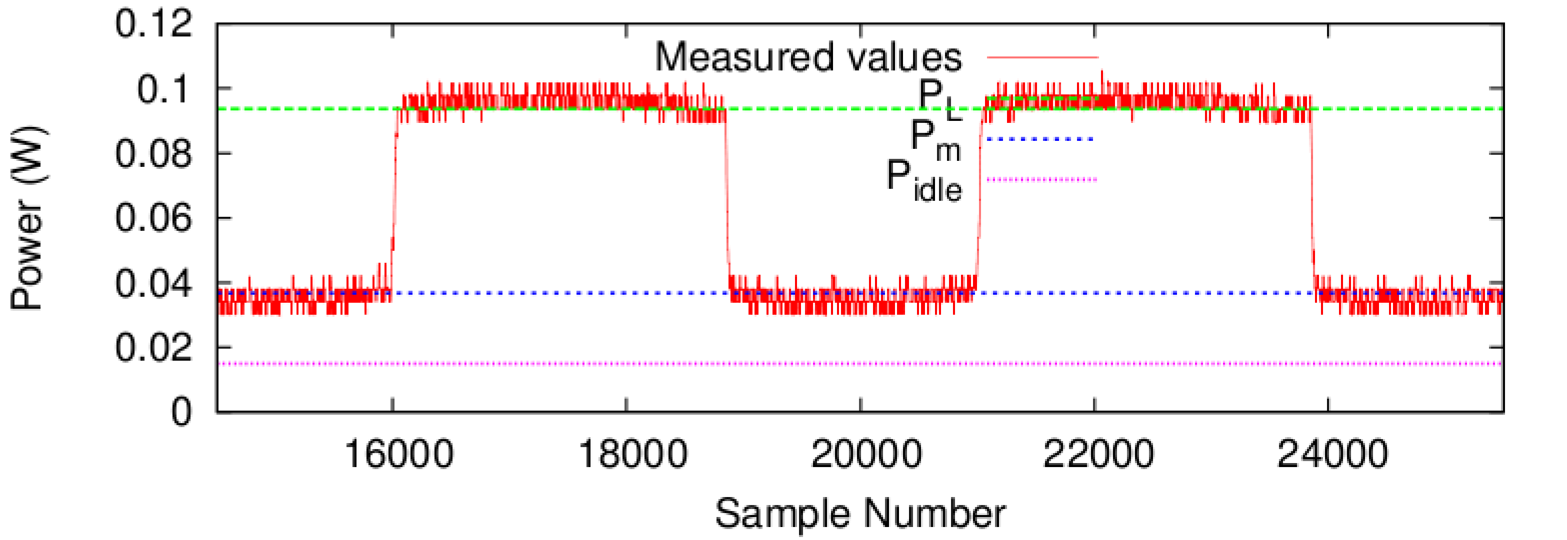
Assume that you want to make a connection to a known device

It has to listen periodically for you attempting to contact it

More frequent listening => quicker connection but more power



18 window
+
32 interval



18 window
+
18 interval

We can model fit these two modes as expected