An *AggreGATE* Network Abstraction for Mobile Devices

Ganesh Ananthanarayanan and **David Zats**

Motivation (1)

- Mobile devices have multiple interfaces with different characteristics
  - Cellular

    - 3G: 255 – 497 kbps, Edge: 36 – 182 kbps
    - Function of location, ubiquitous
  - Wi-Fi

    - 625 – 1700 kbps
    - Depends on backhaul, spotty coverage
  - Bluetooth

    - 335 – 450 kbps
    - Connect to laptop when in vicinity (e.g., CoolSpots)

Motivation (1I)

- Increased proliferation of mobile devices
  - Often in range of each other
- Rich opportunity for nearby devices to collaborate
  - Device can use neighbor's connectivity to browse the web
  - Or collaboratively stream movies from *Hulu*

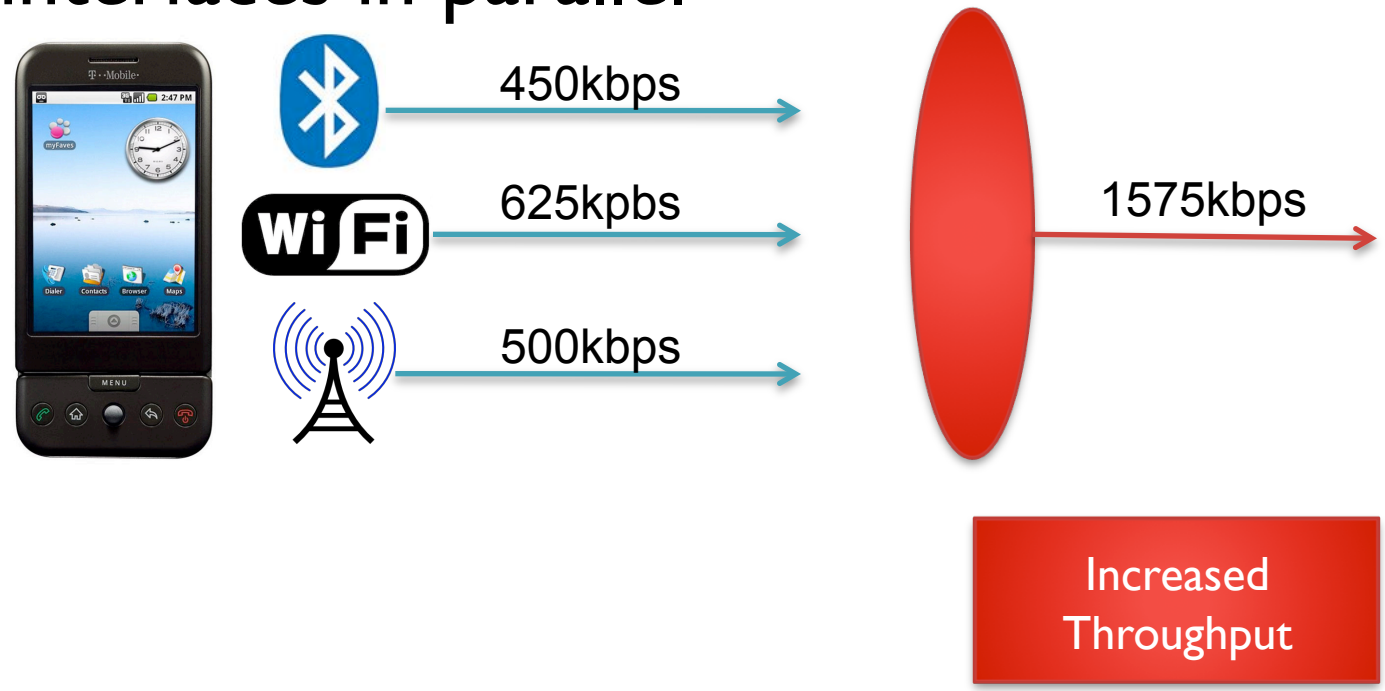
Problem Statement

- *Mobile devices should focus on the collection of connectivity options*

- Design a system that will:
  - Seamlessly and simultaneously use the available connectivity options

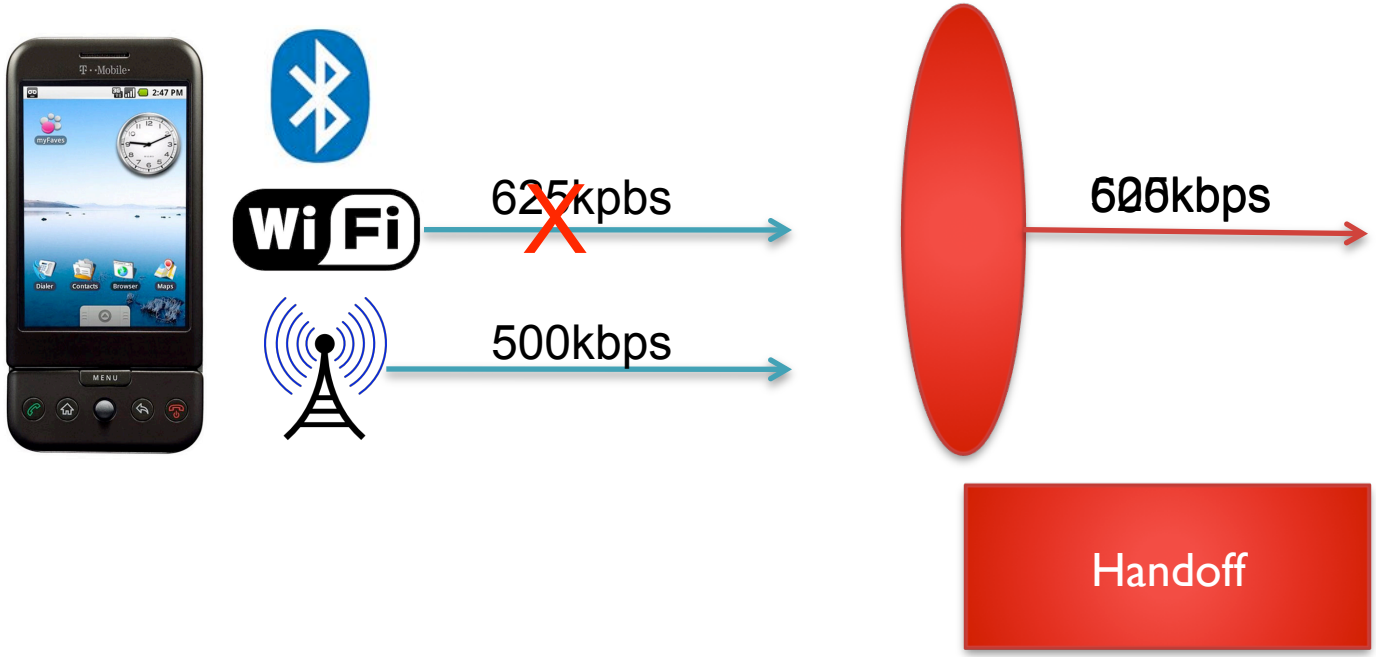
Scenario (1)

- Device uses Wi-Fi, 3G, and Bluetooth interfaces in parallel



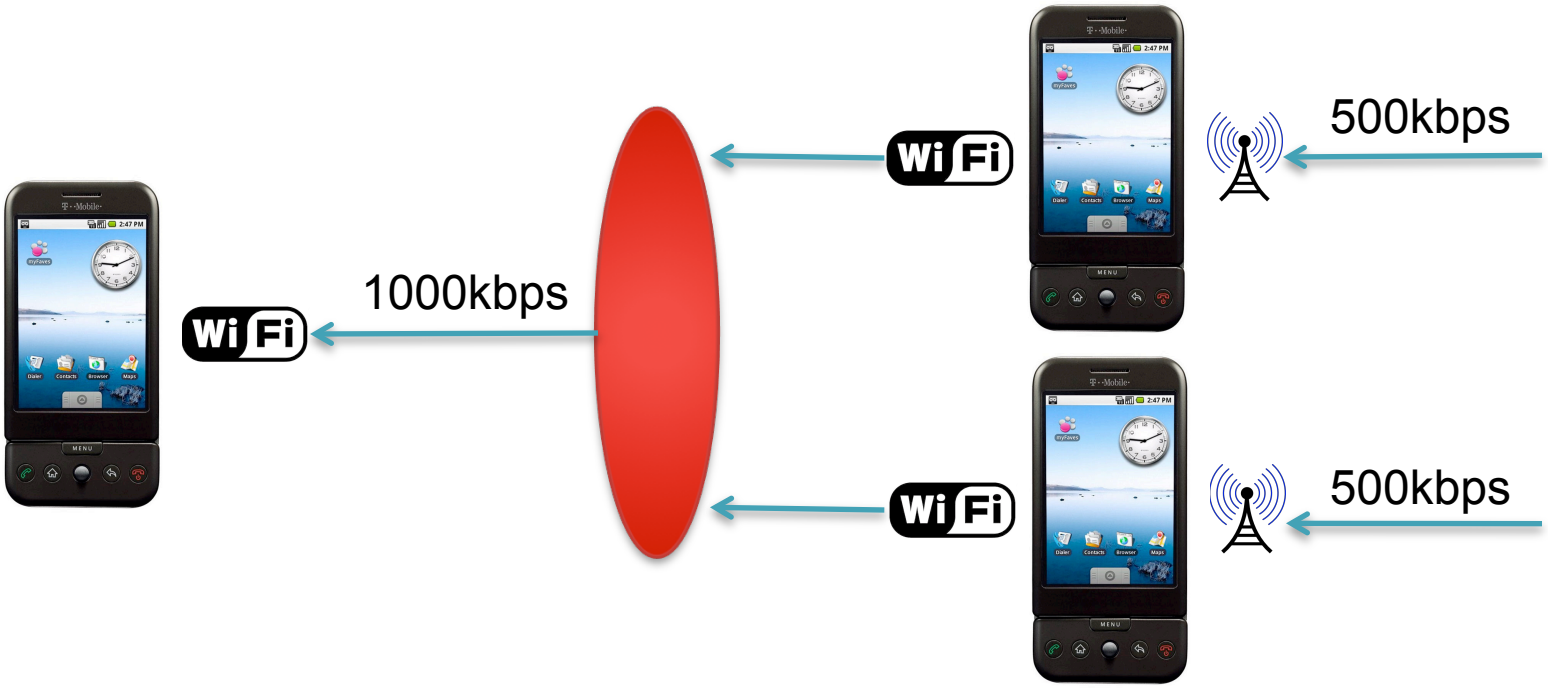
Scenario (II)

- Device seamlessly migrates to 3G connection when it goes out of range of Wi-Fi



Scenario (III)

- Device collaborates with nearby devices over ad-hoc Wi-Fi to parallelize downloads



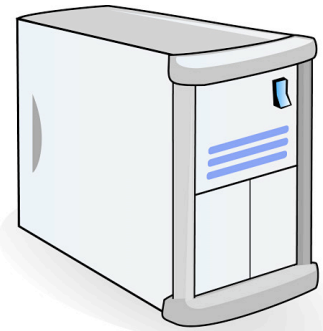
Outline:

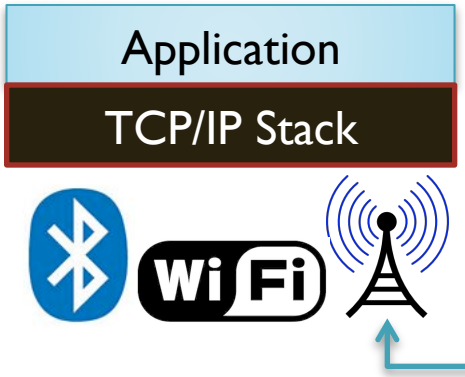
- System Architecture
- Scheduling Algorithm
- Seamless Handoffs
- Evaluation
- Future Work

System Architecture (I)

- Current approach:

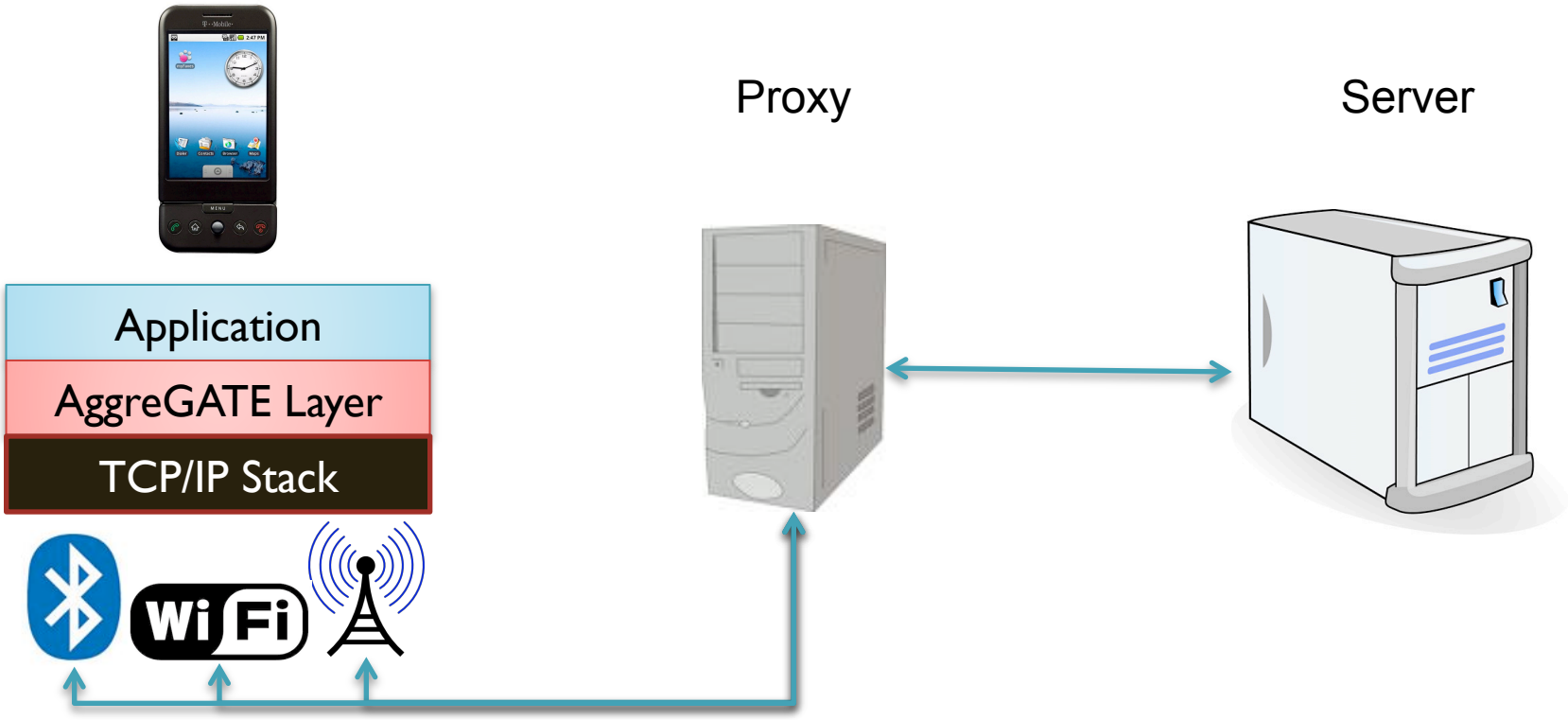
Server





System Architecture (II)

- *AggreGATE approach:*



System Architecture (III)

- AggreGATE Layer
  - Programming interface (from NetAPI):
    - open(), put() and get()
  - Application Data Units (ADU)
- AggreGATE Proxy
  - Receives and maintains connections from mobile device.
- Advantages
  - Avoids modifications to TCP/IP stack
  - Avoids modifications to server

Scheduling Algorithm (I)

- Achieve *optimal* throughput given *dynamically* changing network conditions
  - Device moves farther away from Wi-Fi network
  - New collaborator joins group
- Approach
  - Allocate ADUs in batches, proportional to throughput
  - Dynamically measure throughputs
    - ACK for every ADU batch
    - Weighted moving average of throughput for every connection

Scheduling Algorithm (II)

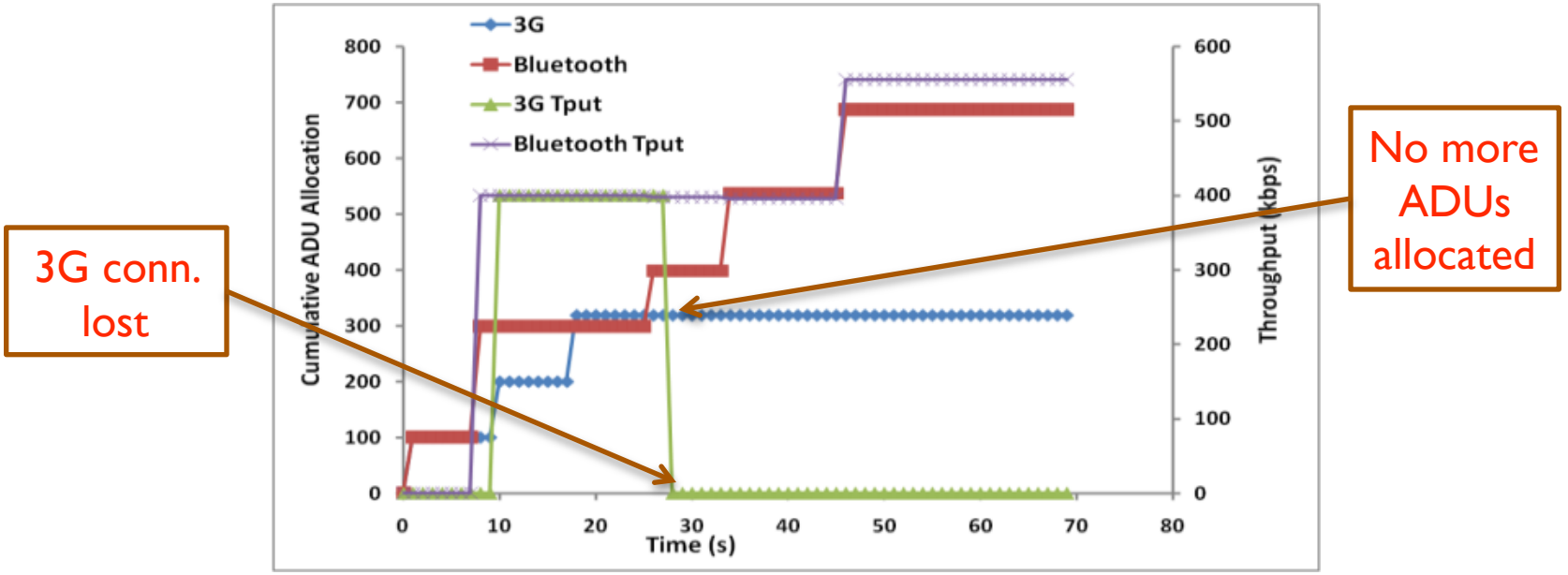
- Handling stragglers
  - Estimate completion time of batch using measured throughputs
  - Obtain progress report for *slow connection*
    - Request last-received ADU sequence number from proxy.
  - Reallocate remaining ADUs to best available connection based on fastest completion time

Handoffs

- Goal: provide uninterrupted service when connections become unavailable
- Approach
  - Extend scheduling: same as handling stragglers, but measured throughput is zero
- Mobility-based preemptive opening/ checking of connections (under progress)

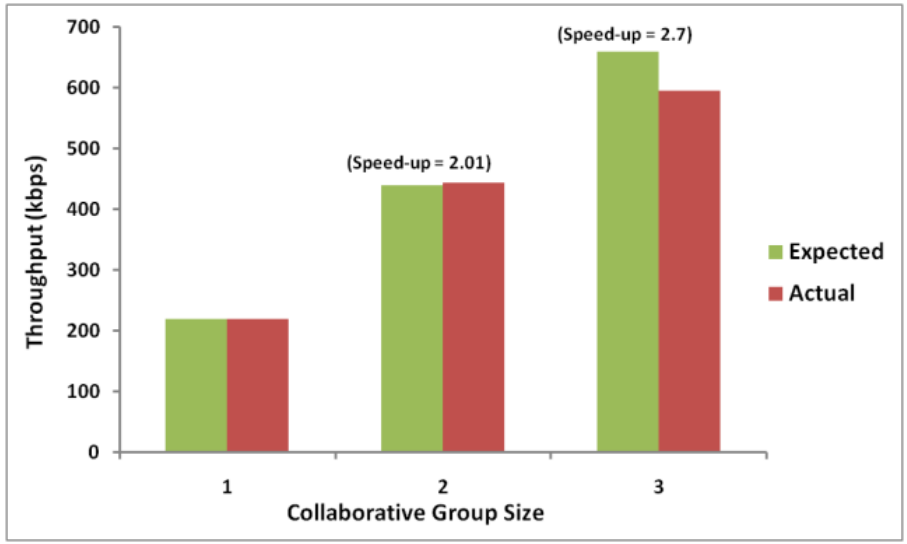
Evaluation (1)

- Single device
  - Uploads to server through proxy using Bluetooth and 3G interfaces
  - Experience 2x speedup and robust handoffs



Evaluation (II)

- Collaborating devices
  - Each mobile device registers its 3G connection with the proxy; devices collaborate through ad hoc Wi-Fi
  - Near-linear speedup



Future Work

- Mobility detection for more informed scheduling decisions
- Incorporate scheduling policies other than maximizing throughput
  - Power, Cost ($$)
- Extensive evaluation