BlueTorrent: P2P content sharing with Bluetooth

Prof. Mario Gerla
Network Research Lab.
UCLA

People-2-People content sharing

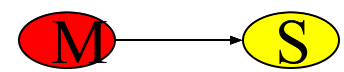
- Scenarios of interest
  - Downloading newspaper, news clips, music on the way to the subway
    - 7 degrees of Separation (Columbia Univ.)
  - Proximity Advertisement
    - Listen to music - Nokia-EMI
    - Advertisement - WideRay
    - “Reading” billboards – CBS
  - Exchanging songs, pictures, ads, movie clips
  - Social networking - Nokia Sensor

Target scenario

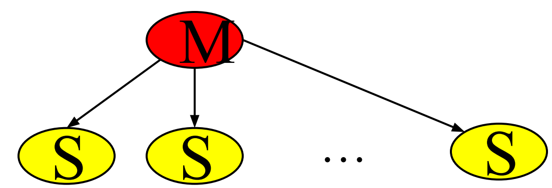
- Airport Corridor, Subway platform
- Multiple Bluetooth Access Points
- Proximity data transfers

P2P Data transfers in Bluetooth

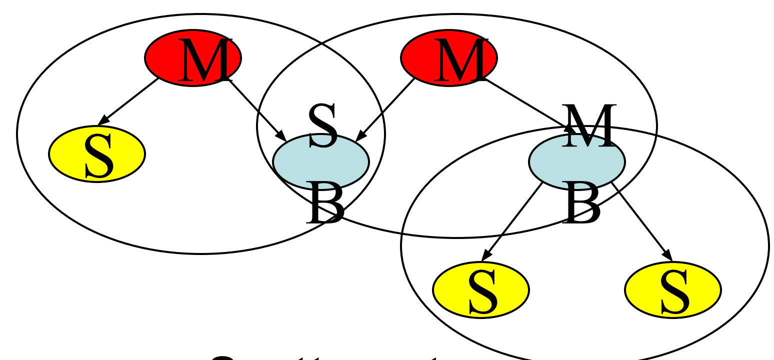
- Piconets
  - 1-to-1 connection for P2P
  - up to 7 slaves
- Scatternet (?)
  - Hardware Limitation
    - Some chips support only limited scatternet
  - Software Limitation
    - No specification
  - Mobility problem
    - Disconnection, Reconfiguration
- Bluetooth Overlays



Piconet (1-to-1, Peer-to-peer)



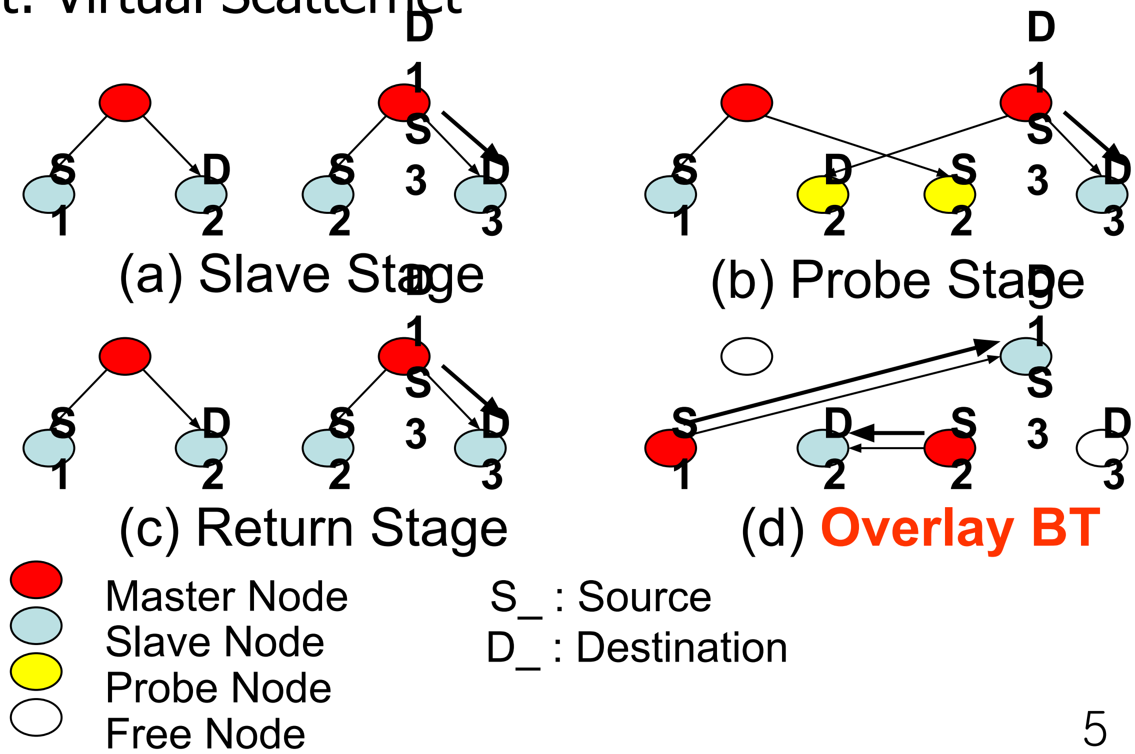
Piconet



Scatternet

The overlay concept

- Piconet Members moves together (Group Mobility)
- Each Piconet represents a “nomadic warrior”
- Works also with single node Piconets
- Opportunistic neighbor Piconet merge => Overlay BT
- Result: Virtual Scatternet



Goal of this study

- Problem definition
  - High penetration rate of Bluetooth devices (cell phones and PDA)
  - Mobile user must go/stop/wait for full download when AP-BT transfer
  - The bandwidth of AP is limited
  - The transmission range of AP is short (10 meters)
- Goal:
  - provide an “effective” content sharing mechanism for Bluetooth users

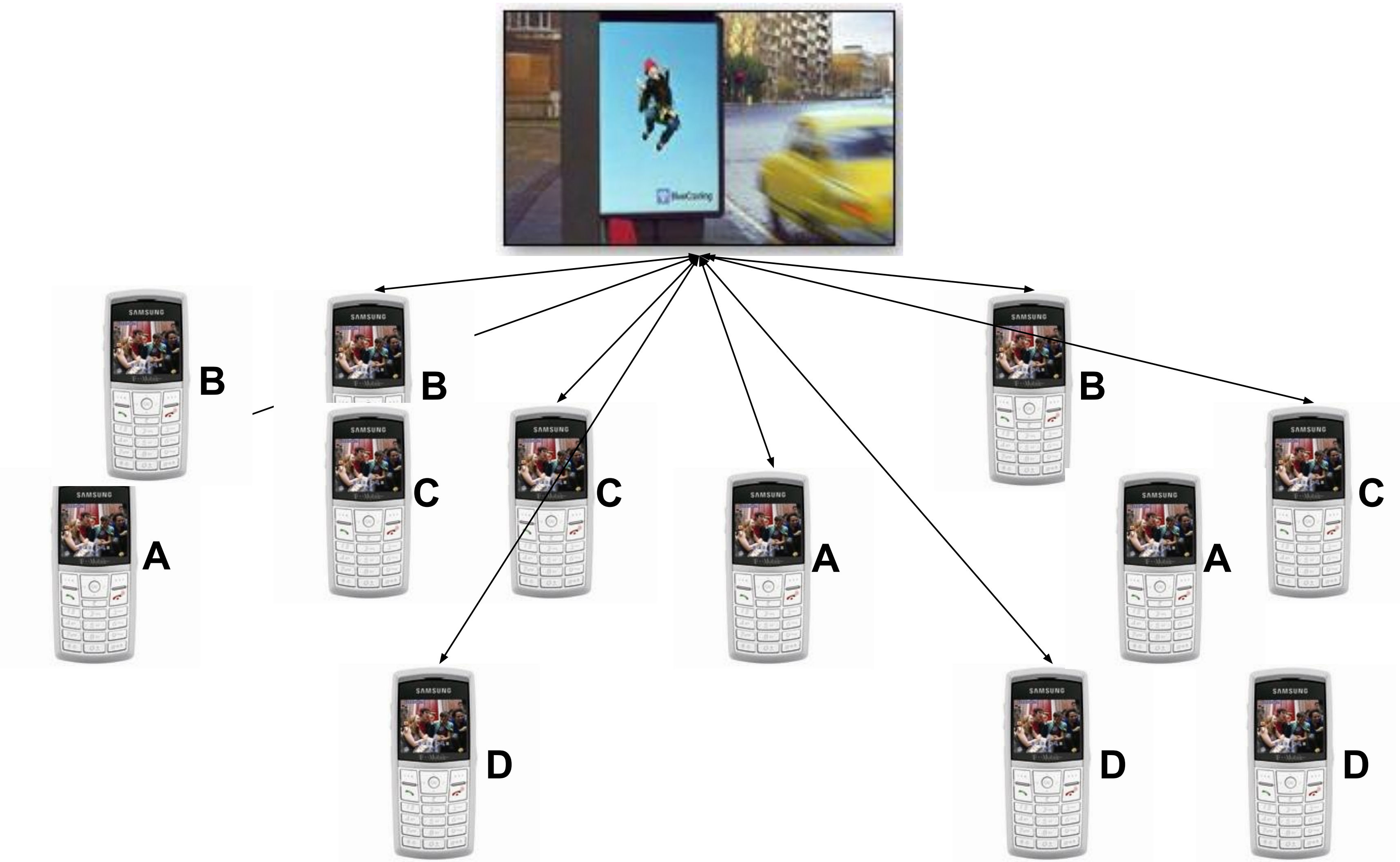
Enter: BlueTorrent

- BlueTorrent
  - Bluetooth P2P Application
  - Sharing small size audio/video ad files (<10MB)
  - Download data from digital billboards on the street with BT-AP transfer
  - Exchange data with BT-BT transfer after receiving from AP

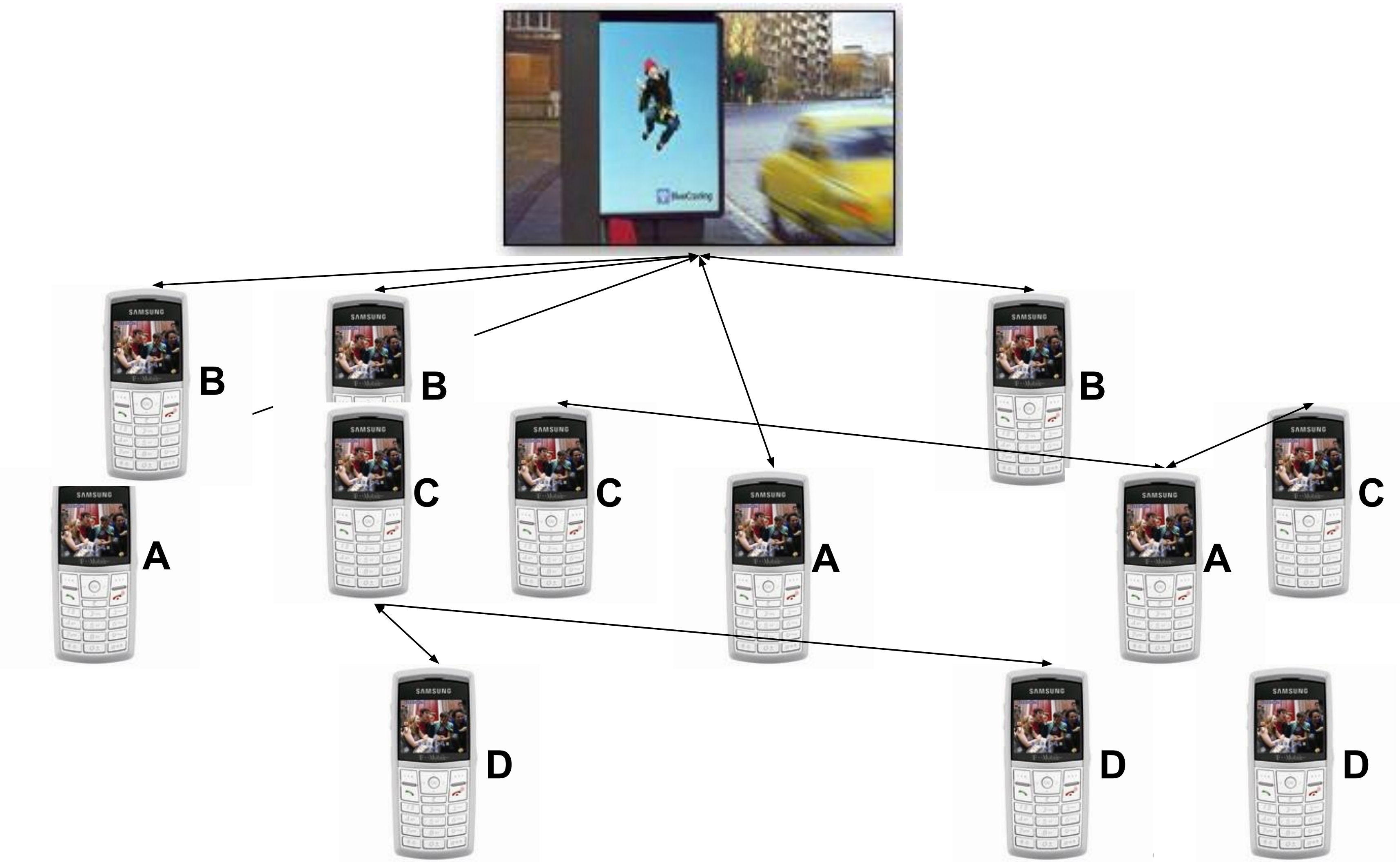
BlueTorrent (cont)

- New contribution:
  - P2P transfer (commercial products support only AP-BT transfer - example Bluecasting)
  - Incentive: to complete download, must help others (same as in Bit Torrent)
- Performance measures:
  - Download percentage
  - Download Finish time

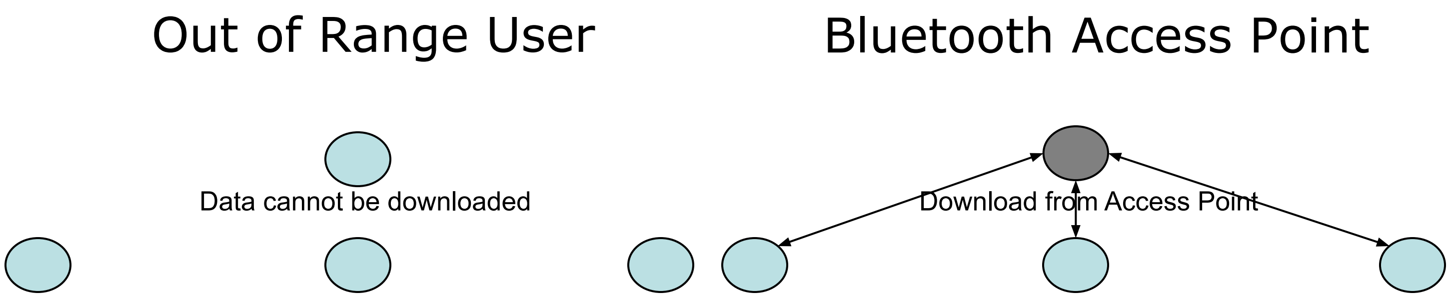
BlueCasting



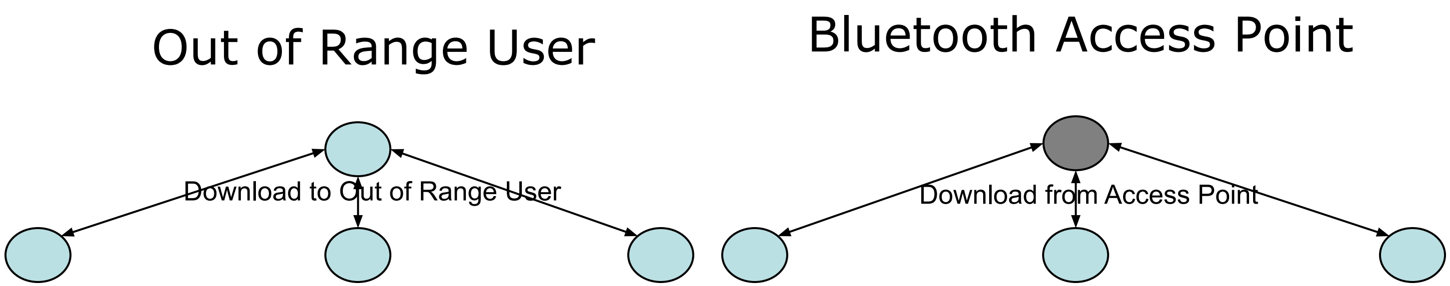
BlueTorrent



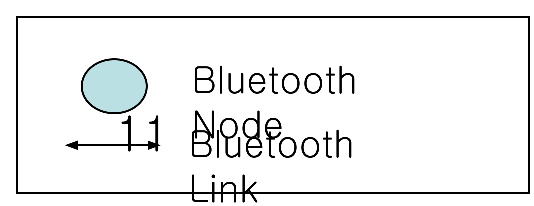
BlueTorrent vs Bluecasting



(A) BLUECASTING = AP only Transfer

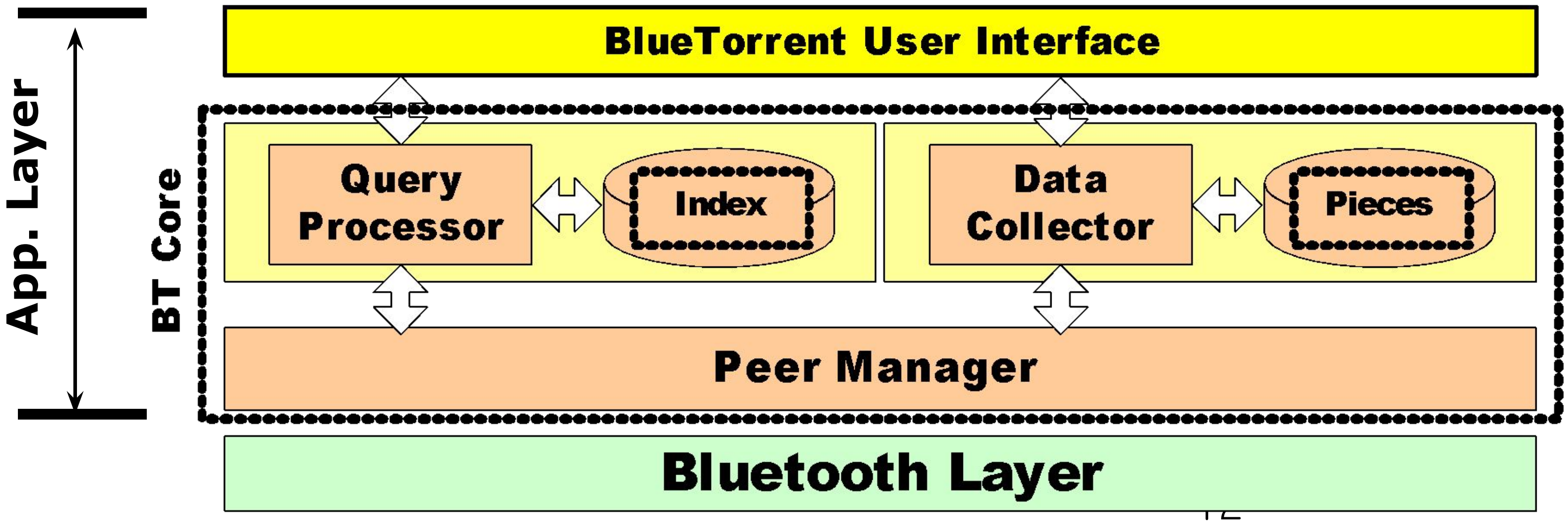


(B) BlueTorrent P2P Transfer



BlueTorrent Architecture

- BlueTorrent core components
  - Query processor
  - Data collector
  - Peer manager
- BlueTorrent user interface



Query Processor

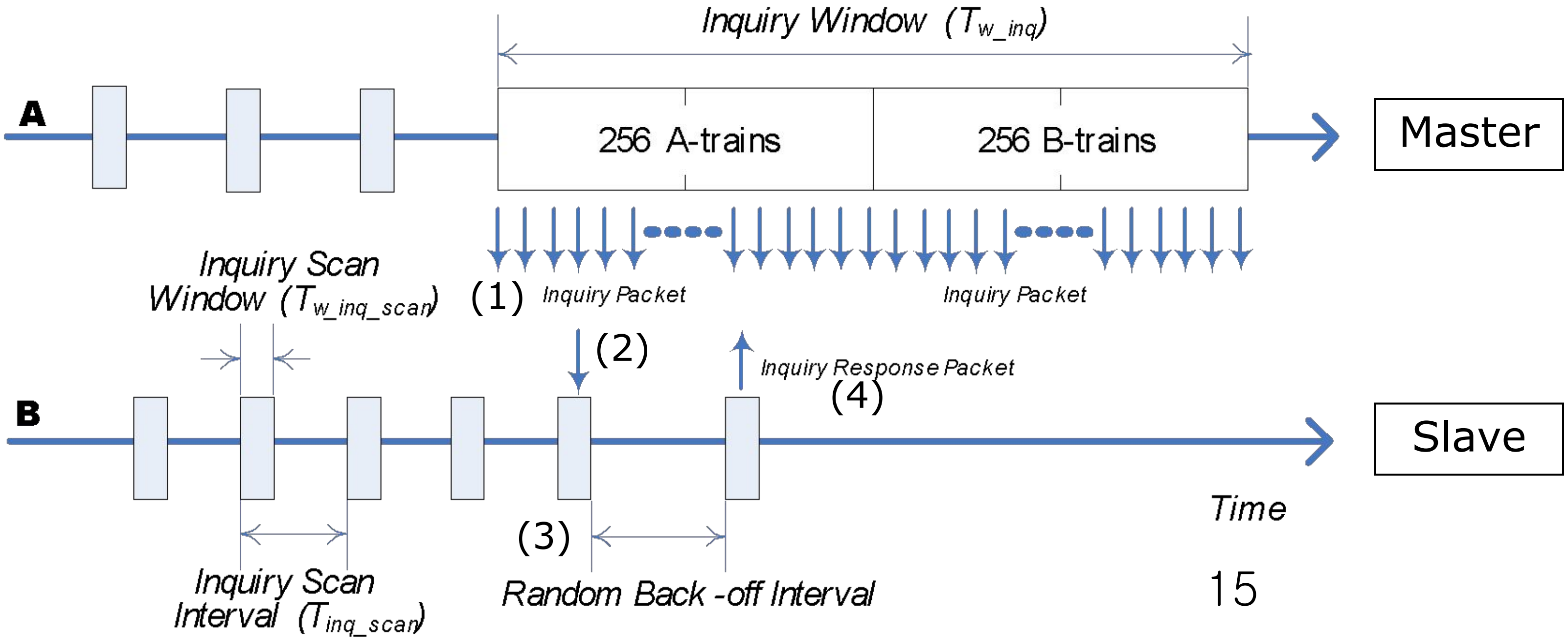
- Query types (push/pull)
  - Only APs can “push” the index to users passing by
  - Regular users send “pull” type query to find the index info. of the interested file
- Index information
  - Unique file ID (e.g., 32bit hash)
  - Title, producer, media type
- User interface allows to send queries to neighbors
  - E.g., title: “Pirates & Caribbean” media type: avg/mpg

Data Collector/Peer Manager

- Data Collector
  - BitTorrent-style file swarming
    - A file is divided into “k” pieces
  - Procedure
    - A new connection is informed to the data collector
    - Exchange bitmap vector to find missing pieces
    - Download missing pieces
- Peer Manager
  - Run periodic inquiry procedure to find peers
  - Find the best peer to download based on connection history

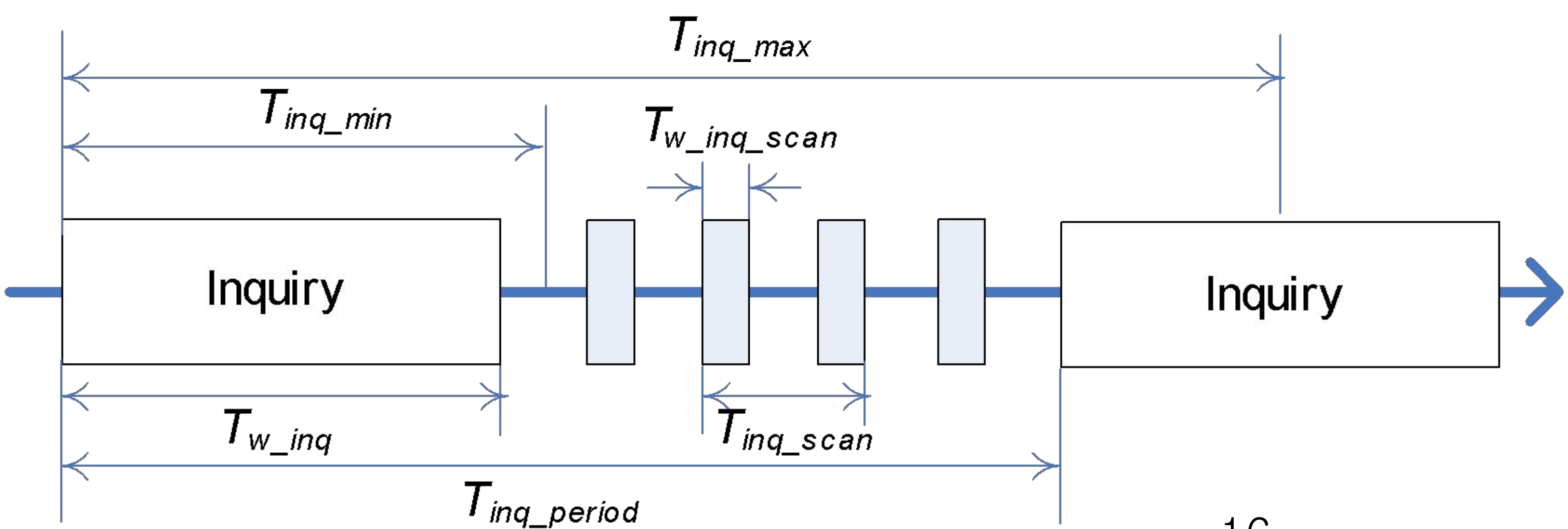
Peer Discovery Procedure

- Inquiry (master) and scan (slave) pair to make a connection
  - Bluetooth was originally developed for "cable replacement"
- Inquiry discovery procedure
  1) A sends inquiry packet trains (window size is multiple of 1.28s)
  2) B receives an inquiry packet
  3) B backs off a random interval over [0,1023]
  4) B sends back an "inquiry response packet"



Peer Discovery Procedure (Cont)

- Periodic inquiry mode for P2P discovery
  - Peers randomly switch their roles to find each other
  - *Periodic_Inquiry_Mode* HCI function
    - T_{w_inq}: fixed length
    - Variable length of the scan period
      - Uniform over $[T_{inq\_min}-T_{w\_inq}, T_{inq\_max}-T_{w\_inq}] = [T_{min}, T_{min}-T_{diff}]$
    - The units of all parameters are multiple of 1.28s

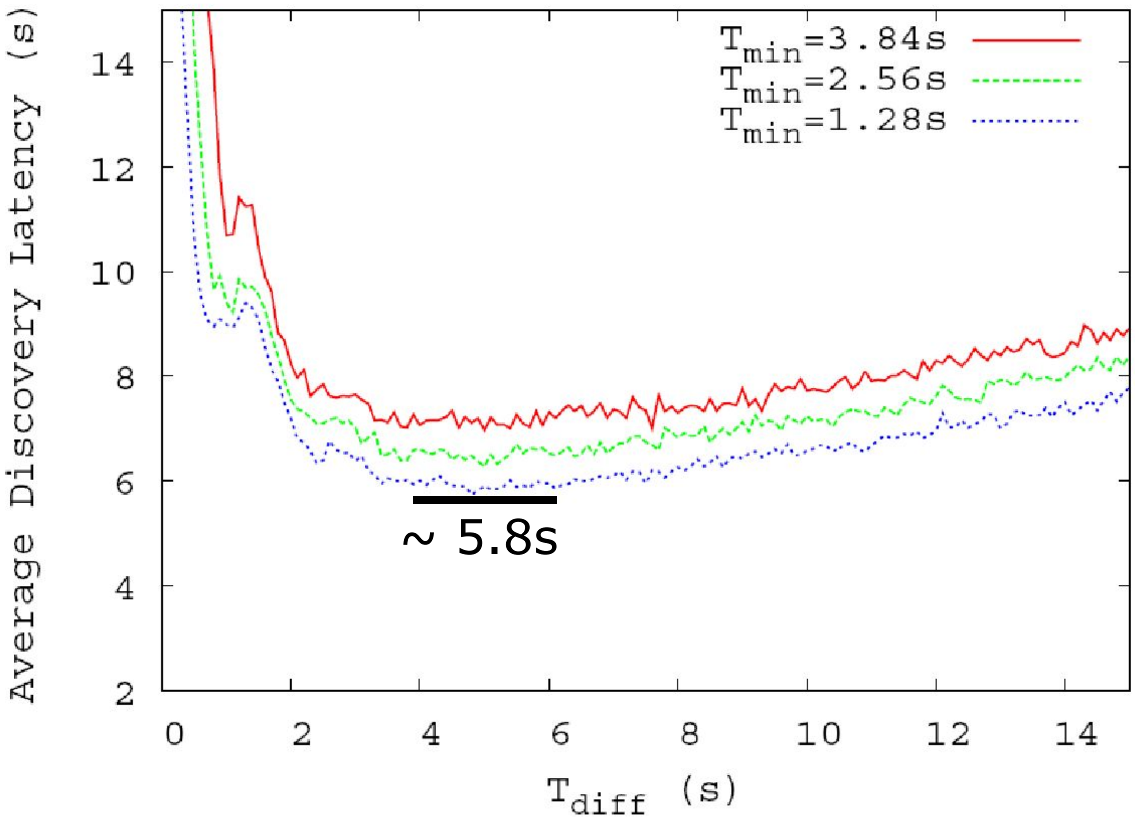


Periodic Inquiry Mode Evaluation

- *InqSim*
  - Slot-level discrete time, event-driven simulation
  - Simulate a random encounter by warming up 100s
  - Simulate two nodes and measure the latency for a peer to find the other
- Parameters of interest
  - Scan period: $[T_{min}, T_{min} + T_{diff}]$
  - Inquiry scan interval: T_{inq_scan}
    - Default: 1.28s
  - Random back-off interval: $[0, T_{max\_bo}]$
    - Default is 1023 (640ms), but the actual value depends on chipset (vary from 0 to the default value)
- Simple relationship
  - Scan period must be larger than the inquiry scan period: i.e., $T_{min} \geq T_{inq\_scan}$
  - Efficiency of scan period depends on how many “scans” happen during that period

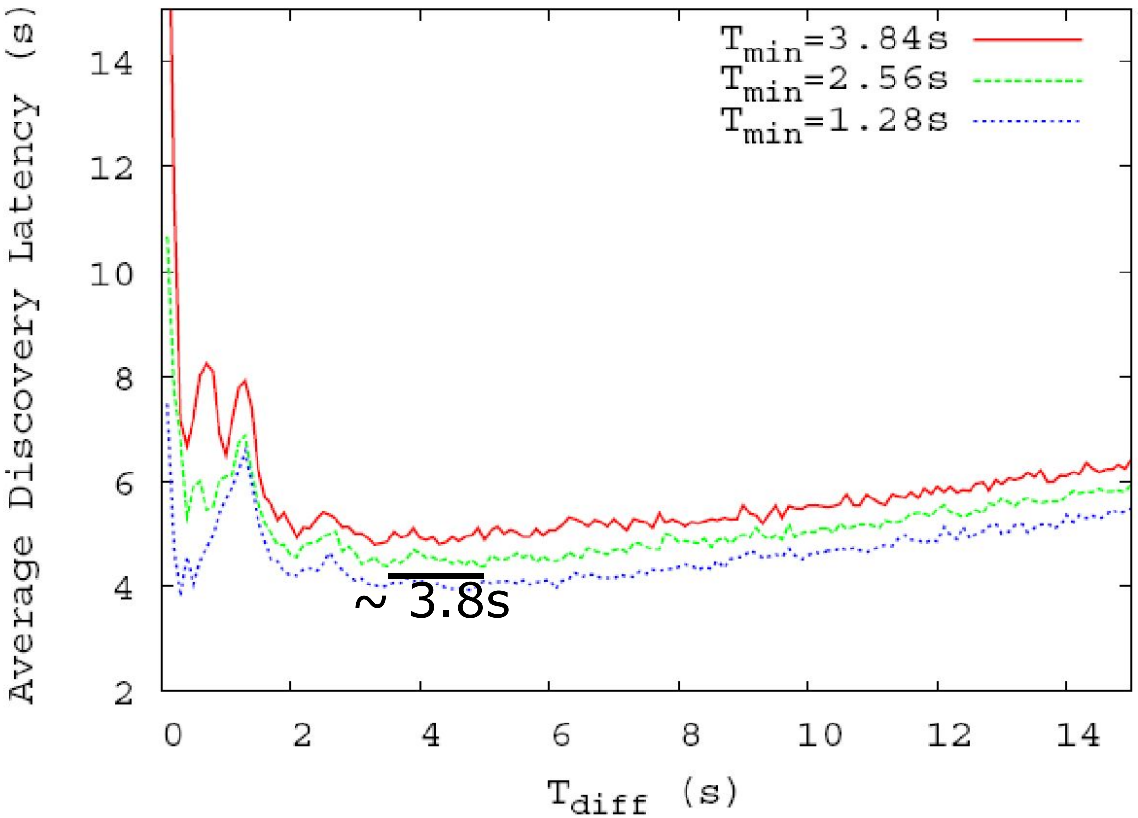
Periodic Inquiry Mode Result (1)

- Discovery latency with T_{max_bo} = 1023 slots (640ms)
  - Efficiency of scan period is important



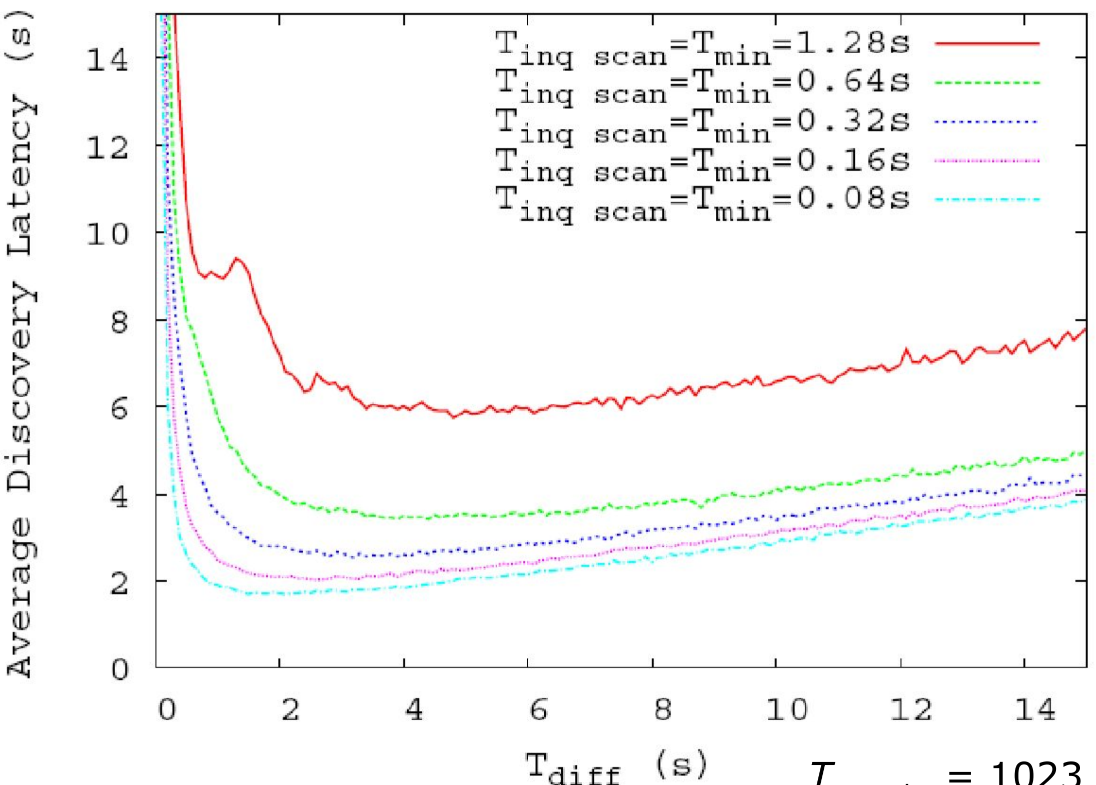
Periodic Inquiry Mode Result (2)

- Discovery latency with T_{max_bo} = 127 slots (80ms)
  - Smaller back-off reduces the latency (5.8s □ 3.8s)



Periodic Inquiry Mode Result (3)

- Discovery latency vs. inquiry scan interval
  - Smaller scan interval greatly reduces the latency (but more energy consumption)
  - Inquiry scan interval is important for determining the latency



)

T_{max_bo} = 1023 slots (640ms)

Periodic Inquiry Mode Evaluation Summary

- Latency depends
  - Maximum back-off size (chipset dependent)
  - Inquiry scan intervals (energy/delay tradeoff)
- P2P discovery is "expensive"!!
  - ~6s on avg. (based on spec.)
  - ~4s on avg. (by halving the inquiry scan period)
- Coarse granular *Periodic_Inquiry_Mode* HCI function parameters (multiple of 1.28s) lead to sub-optimal avg. delay
  - Application layer function with fine granular parameters can minimize the avg. delay

Simulation Setup

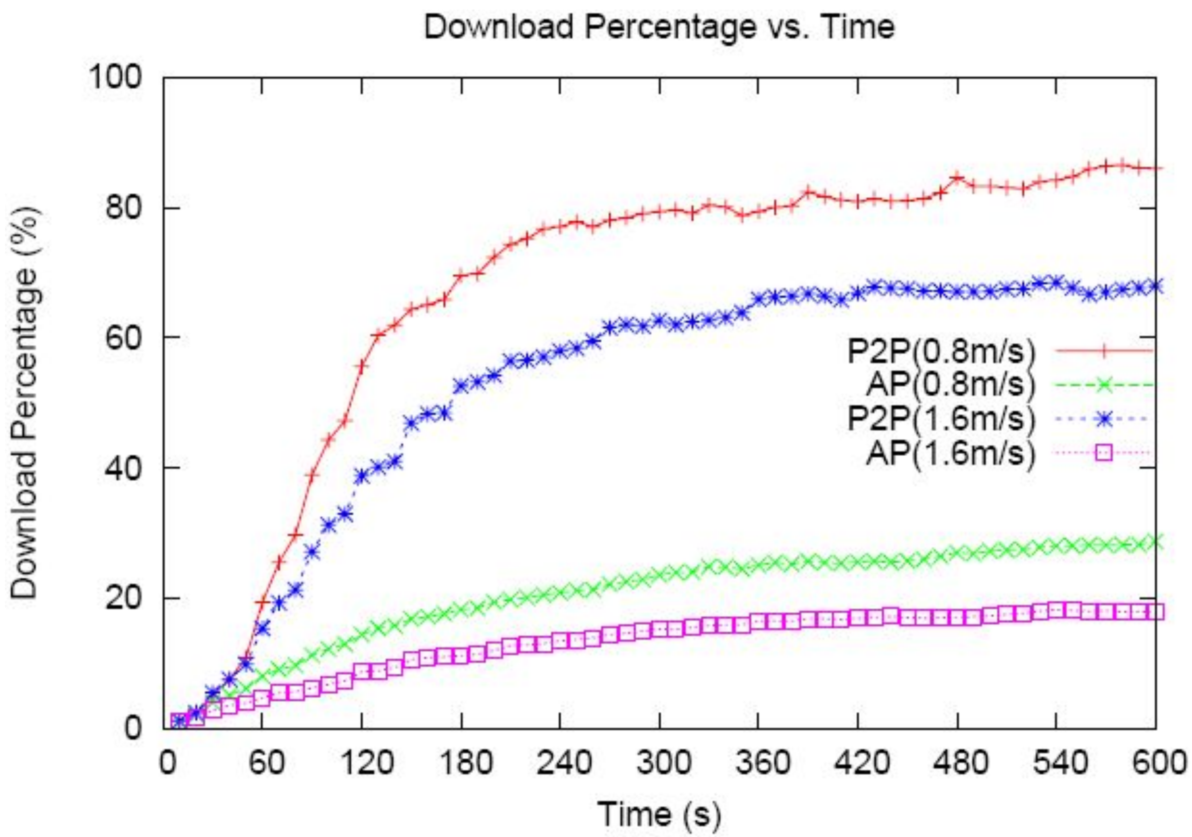
- NS-2 + UCBT extension
- Corridor mobility model:
  - Rectangle Area (length >> width)
  - Two directions (West -> East, East -> West)
  - Constant speed randomly selected over [0, V_{max}]
- When reaching to bound
  - North or South: nodes are mirrored back to the area
  - West or East bound: nodes are restarted
    - Reset mode: user data is cleared (acting as a new node)
    - No-reset mode: user data is remained (re-enter the area)
- Mobility Setting:
  - # of nodes: 25, 50, 75, 100 (Default: 50 nodes)
  - V_{max} =0.0 (static), 0.4, 0.8, 1.2, 1.6 m/s
  - Area: [25, 50, 100] x [3,5] m^2 (Default: 100x5 m^2)

Simulation Setup (Cont)

- Test scenarios: AP mode vs. P2P mode
  - AP mode: data is only transferred from AP to nodes
- Periodic inquiry mode with inquiry scan interval (0.64s) and scan period [0.64s, 4s]
- Distribute 1.2MB files
  - Divided by 50, 100, 200 blocks (# of blocks)
- Metrics
  - Download percentage of all the nodes that have passed the simulated area (time avg.)
    - No-reset: the number of nodes is the same as the number of nodes in the network
    - Reset: the number of nodes is increasing as time passes

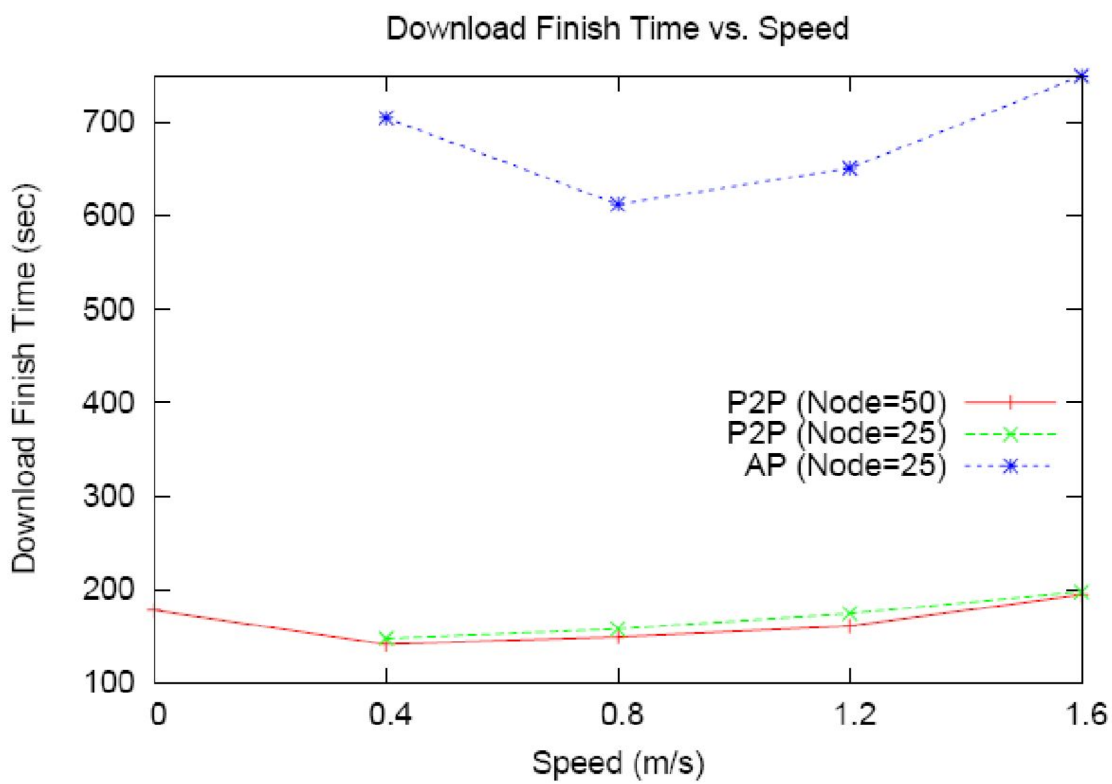
Simulation Result (1)
Download Percentage vs. Time (reset)

- Reset – a new node enters after a node get out of area
  - Time avg. of download percentage shows the effectiveness of the area
- Speed is critical: as speed increases, download %age decreases



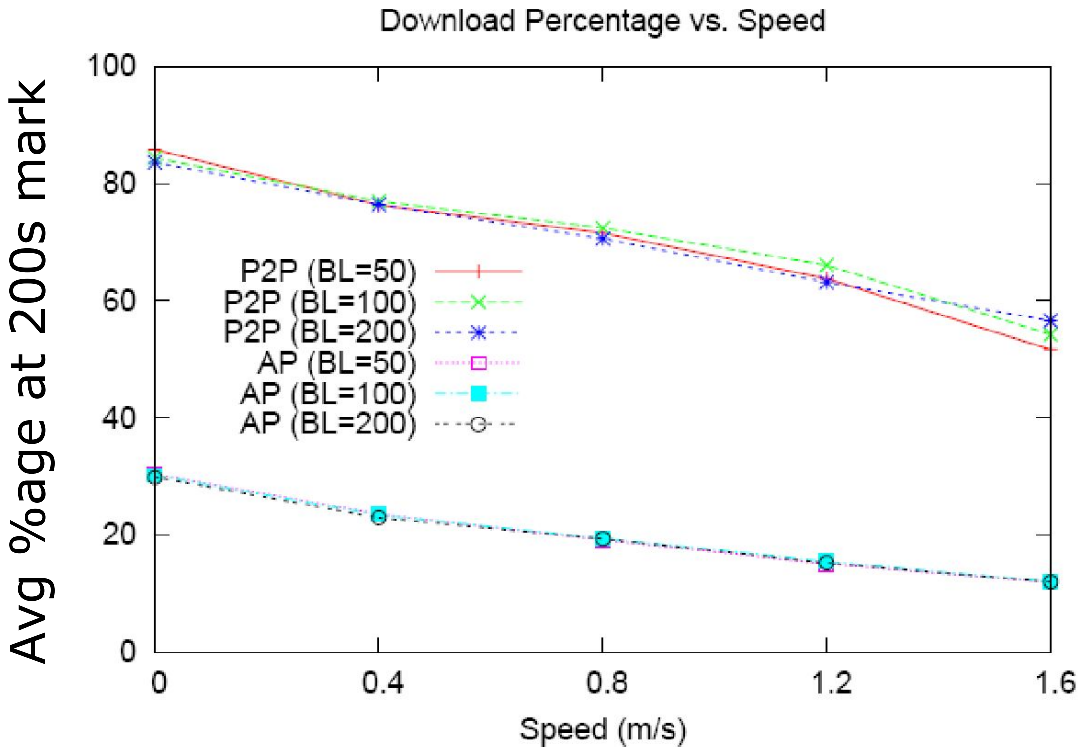
Simulation Result (2)
Finish Time vs. Speed (no-reset)

- AP performs best at 0.8m/s
  - Idle period of AP decreases (0.4m/s => 0.8m/s)
  - If one moves too fast, usefulness (trans/(discover+trans)) of a connection decreases
- P2P increases connectivity (esp. w/ low density)
  - After a certain threshold, density is not critical impact (only speed)



Simulation Result (3)
Number of Blocks (reset)

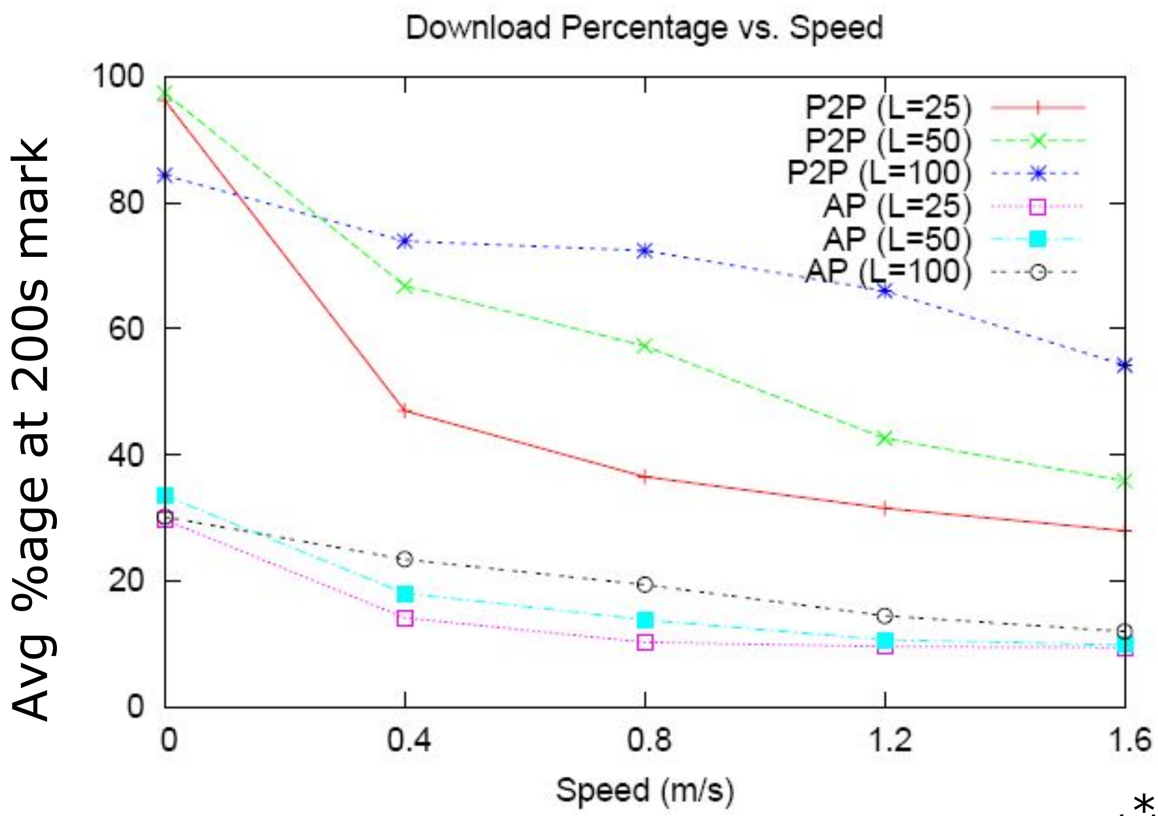
- Download percentage is not sensitive to # of blocks
  - Overhead of L2CAP layer is not significant
- Mobility has a greater impact
  - Too large block causes performance loss since non-complete blocks are flushed (esp, more frequent in the P2P mode)



* 50 nodes
* 100x5 m^2 corridor

Simulation Result (4)
Corridor Length (reset)

- Length of corridor affects the node density
- Longer corridor is more resilient to speed



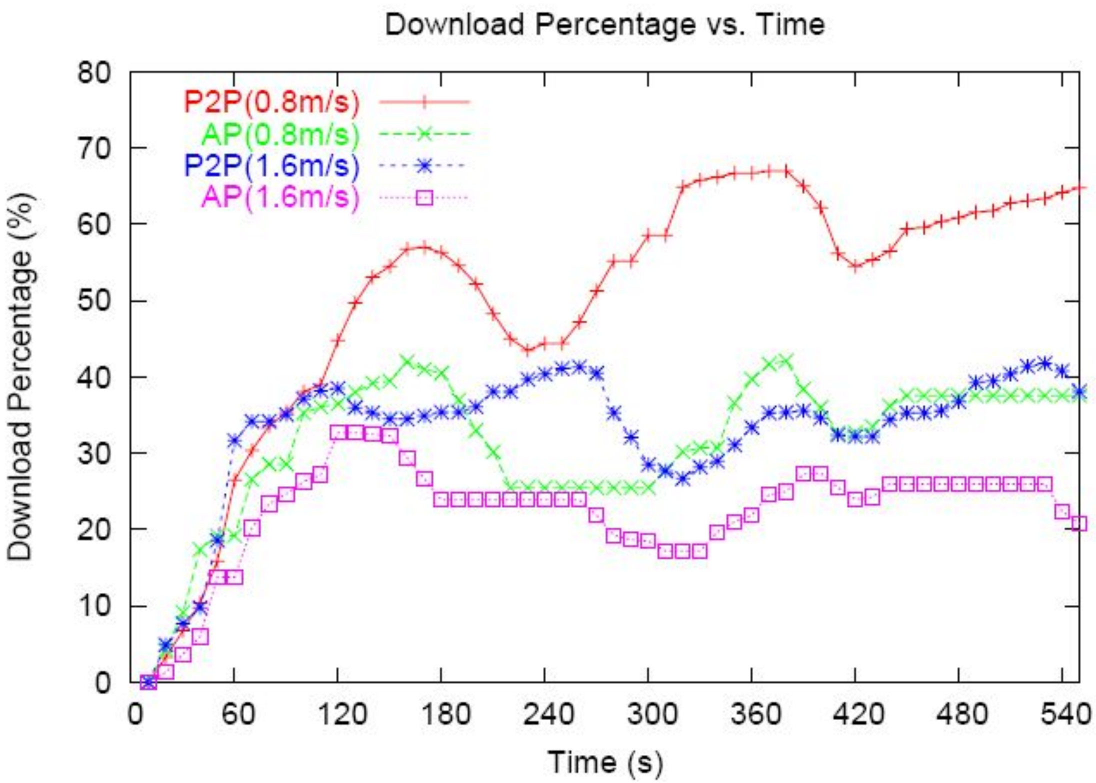
* 50 nodes
* $100x5\ m^2$ corridor

Experiment Setup

- BlueZ Bluetooth protocol stack for Linux
- Bluetake BT009Si (Silicon Wave, Bluetooth v1.2)
- 3 desktops and 5 laptops (Pentium IV/512MB RAM)
- Mobility emulation:
  - AP is up for a certain period of time; to simulate a node moves out of the AP's range (1 AP vs 7 users)
    - Move 20m (max AP's range) at a speed 0.8m/s (=25s)
    - Only P2P mode can transfer data in AP down period
  - Speed: 0.8m/s, 1.6m/s, corridor length: 100m
  - Reset period (i.e., lifetime) is determined by the speed and corridor length
    - e.g. for 0.8m/s a node is reset after 100m/0.8m/s=125s

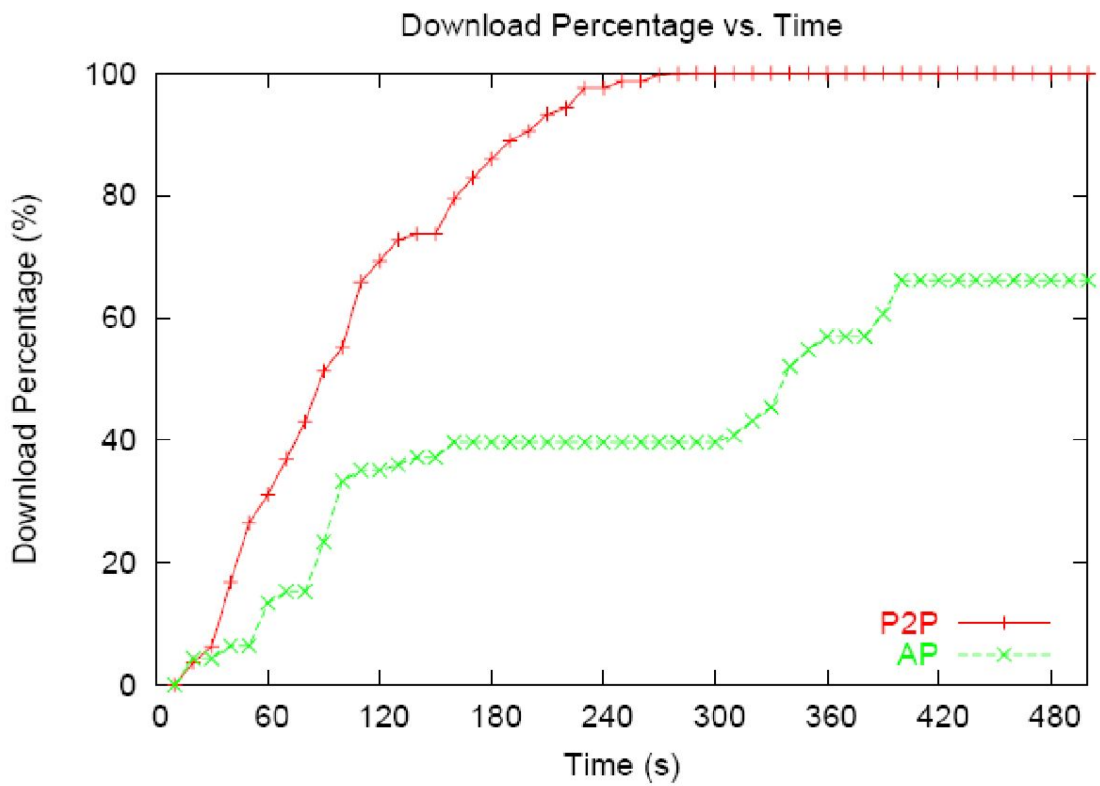
Experiment Result (1)
Download Percentage vs. Speed (reset)

- Flat line: AP down period (in the case of AP only mode)
- Time avg. drops due to Node Reset (Number of node++)
- Overall, P2P mode outperforms AP only mode



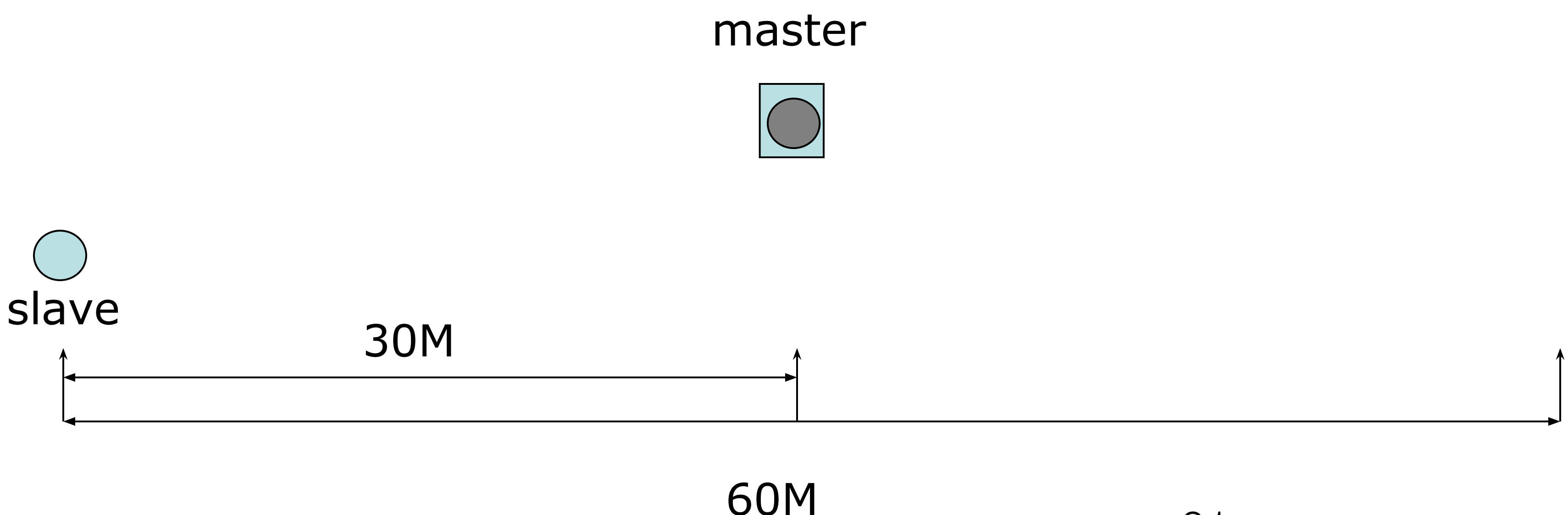
Experiment Result (2)
Download Percentage vs. Speed (no-reset)

- P2P mode is faster than AP only mode

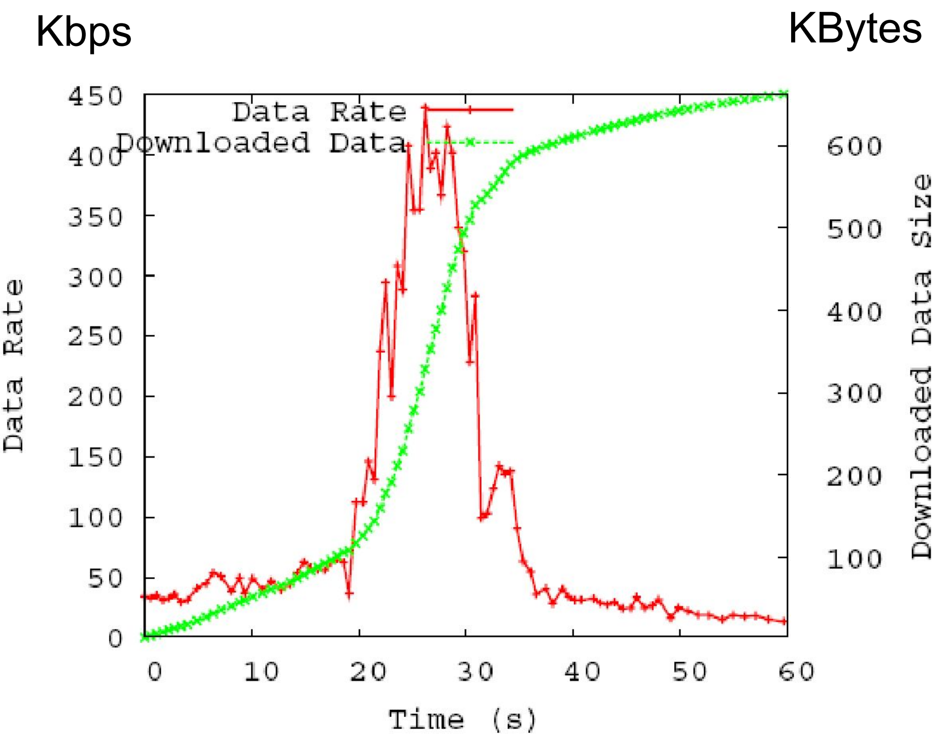


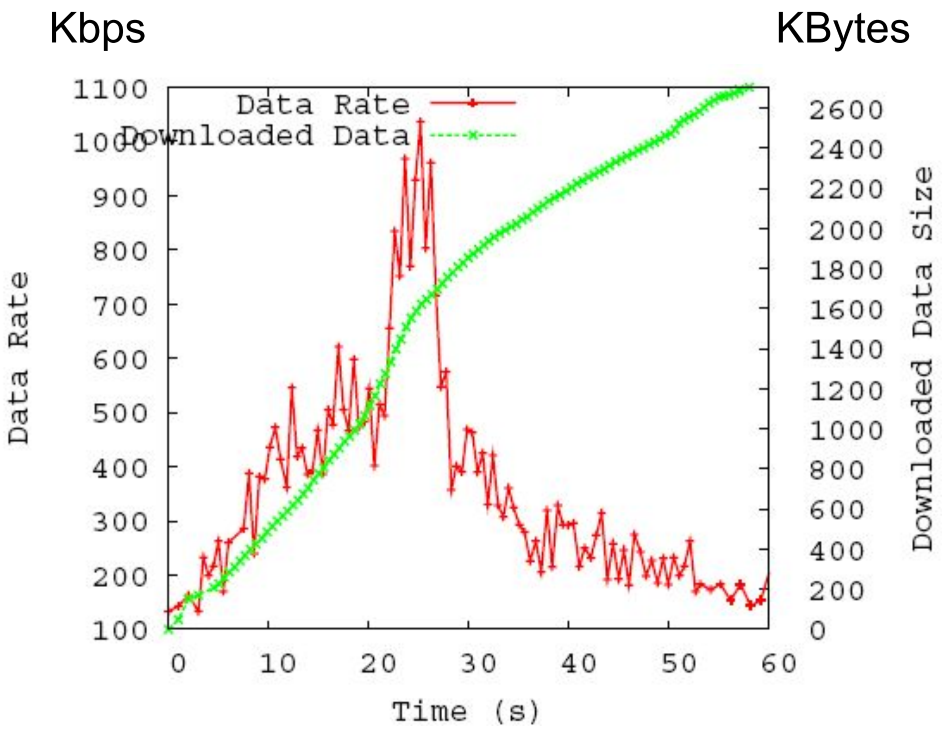
Real Environment Experiment Setup

- 2 Laptops (Master, Slave)
- Ackerman Union (w/ Interference, Obstacle)
- Speed 1 m/s (5 meter marks)
- BT 1.1 ⇔ 1.1
- BT 2.0 ⇔ 2.0



Real Environment Experiment Result





BT 1.1 ⇔ 1.1

BT 2.0 ⇔ 2.0

Connection Can be made in 30m distance

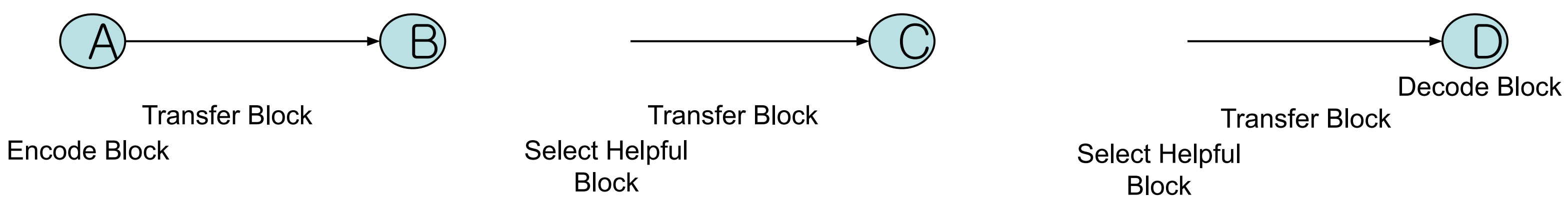
Encoding Method
(Data Transfer Options)

- Normal (Non-coding) Data Transfer
  - Exchange segment map (shows list of current segments)
  - Prepare missing segment list
  - Randomly choose one segment from missing list
- Network Coding Data Transfer
  - Encode Code Block
  - Transfer Code Block
  - If received Code Block is helpful, decode received Code Block
- Rateless (Erasure) Coding Data Transfer
  - Encode Code Block beforehand
  - Maintain missing data list
  - Randomly choose one segment from missing list

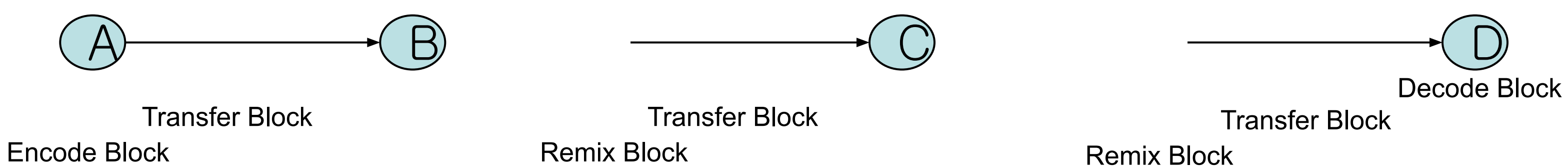
Encoding Method
(Data Transfer Options) (Cont)

Normal (Non-coding) Data Transfer

Rateless Coding (End-to-End Coding)

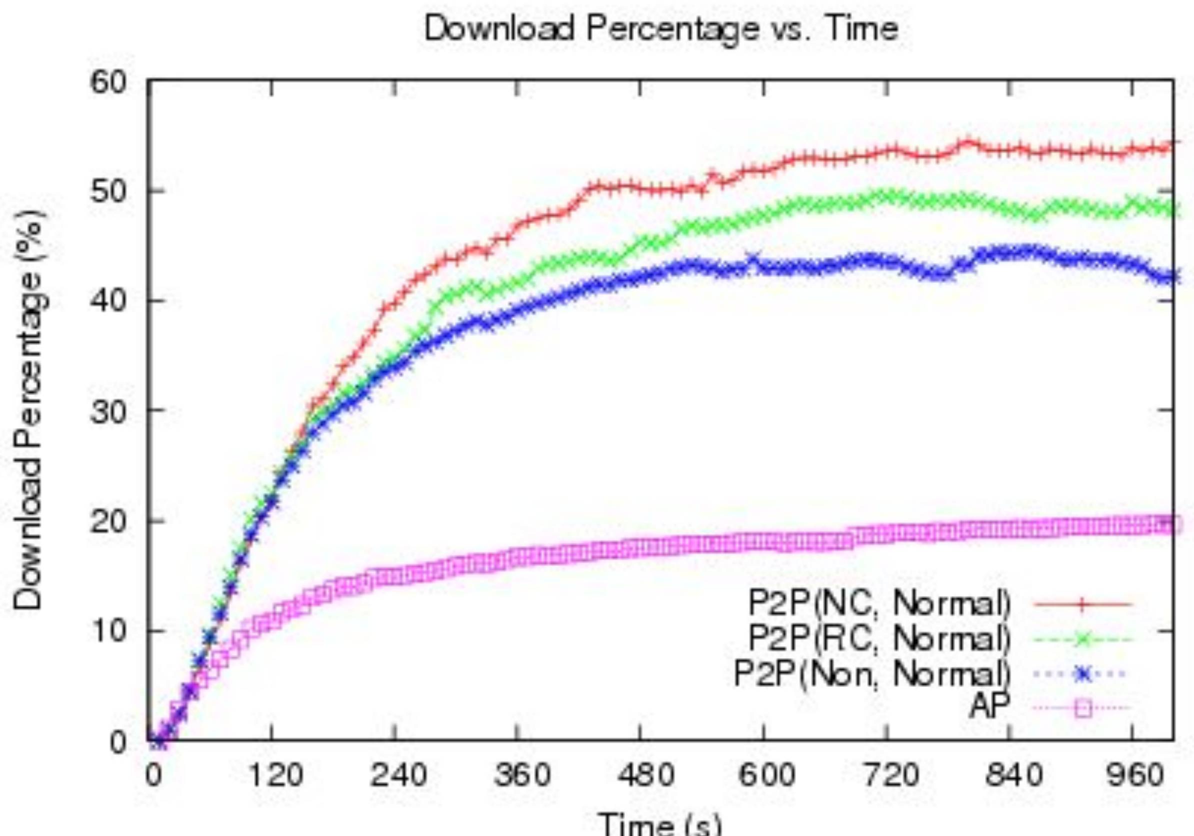


Network Coding



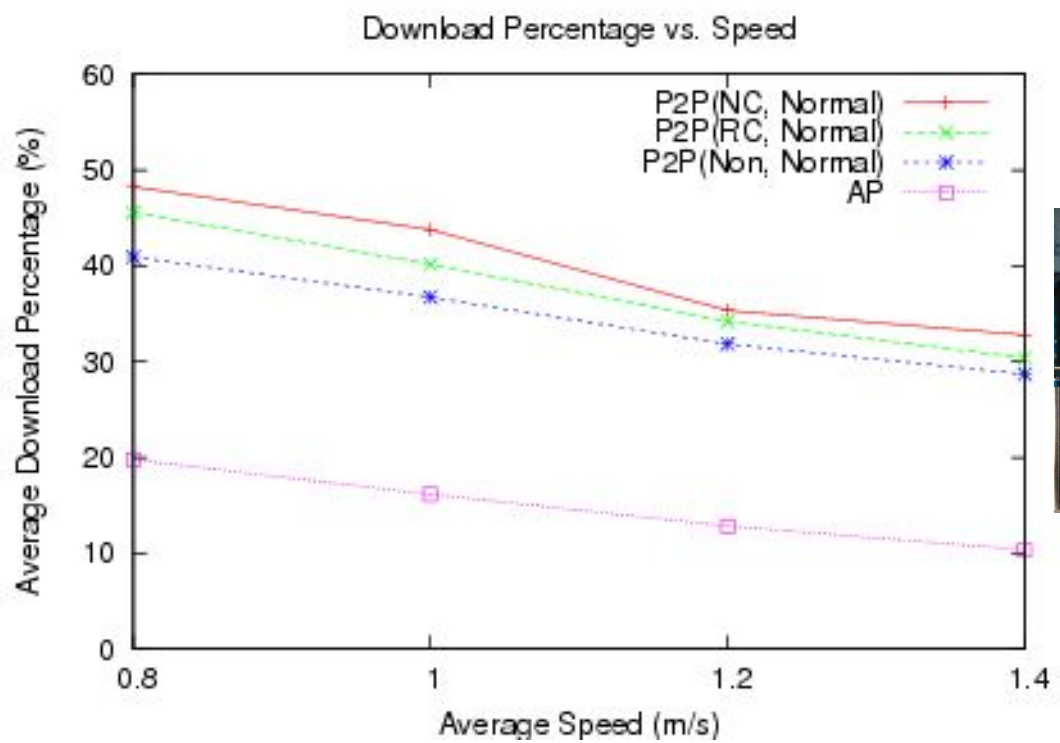
Encoding Method Result
Download vs encoding scheme

- Network Coding > Rateless Coding > Non-coding



Encoding Method Result
Download vs speed

- Network Coding > Rateless Coding > Non-coding
- Speed affects all methods



Conclusion

- Designed and implemented BlueTorrent
  - Peer manager, Query processor, Data collector
- Found the optimal parameter setting for periodic inquiry mode
- Showed that P2P networking outperforms the conventional client-server mode (i.e., AP mode)
- Feasible in the walking speed range
- Performance enhancements using:
  - Network Coding
  - Rateless (Erasure) Coding

Future work

- Multiple simultaneous downloads (service discovery, scheduling)
- Incentives/security
- Advertising (e.g., embedding ads in files, like Google)
- Urban sensing; data collection
- BT vs ZigBee vs WiFI in smart phones