Content Delivery over Wireless Mesh Networks

John N. Daigle

wcdaigle@olemiss.edu

[1] Department of Electrical Engineering
University of Mississippi

Overview

Objective Statement

Our objective is to explore content delivery in wireless mesh networks.

- Wireless mesh network
- Content delivery
- RaptorQ -based alternatives
- Useful protocols or networking environment
- Quantify performance

Wireless Mesh Networks

A wireless mesh network is a wireless network that

- Consists of a collection of interconnected wireless stations (STAs) sharing a common channel
- Facilitates multi-hop communications
  - among the wireless STAs of the wireless network
  - between wireless station of the network and one or more gateways to other networks, including the Itnternet
- Is self-organizing
- Implements the wireless mesh standard (IEEE 802.11s)

Content

Content is the digital information delivered over a network. It can consist of things like

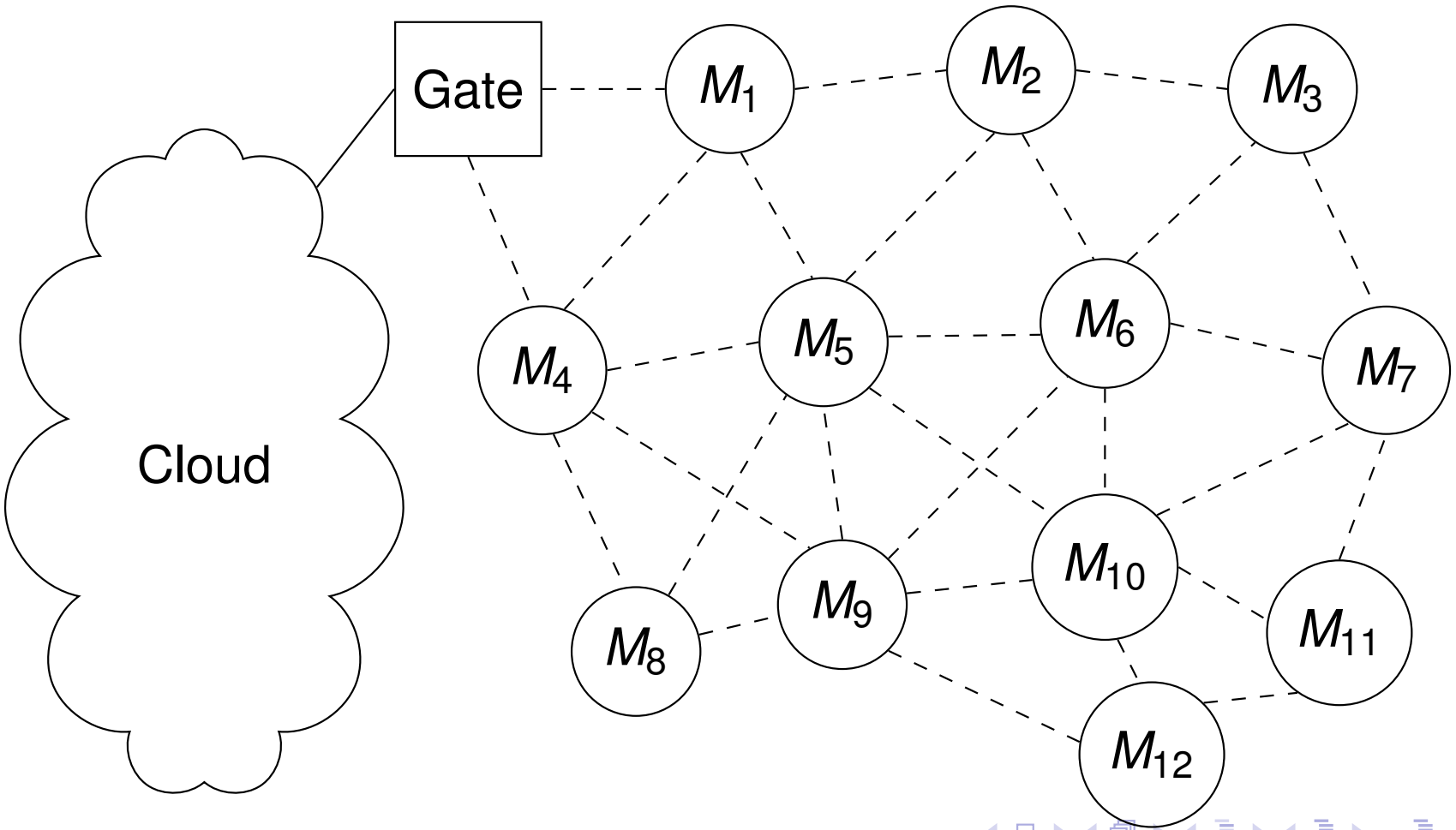
- Live or on-demand streaming media
- Downloadable files such as
  - movies
  - software such as programs and operating system updates
  - documents
  - images
- Web objects

Specific Focus of this Talk

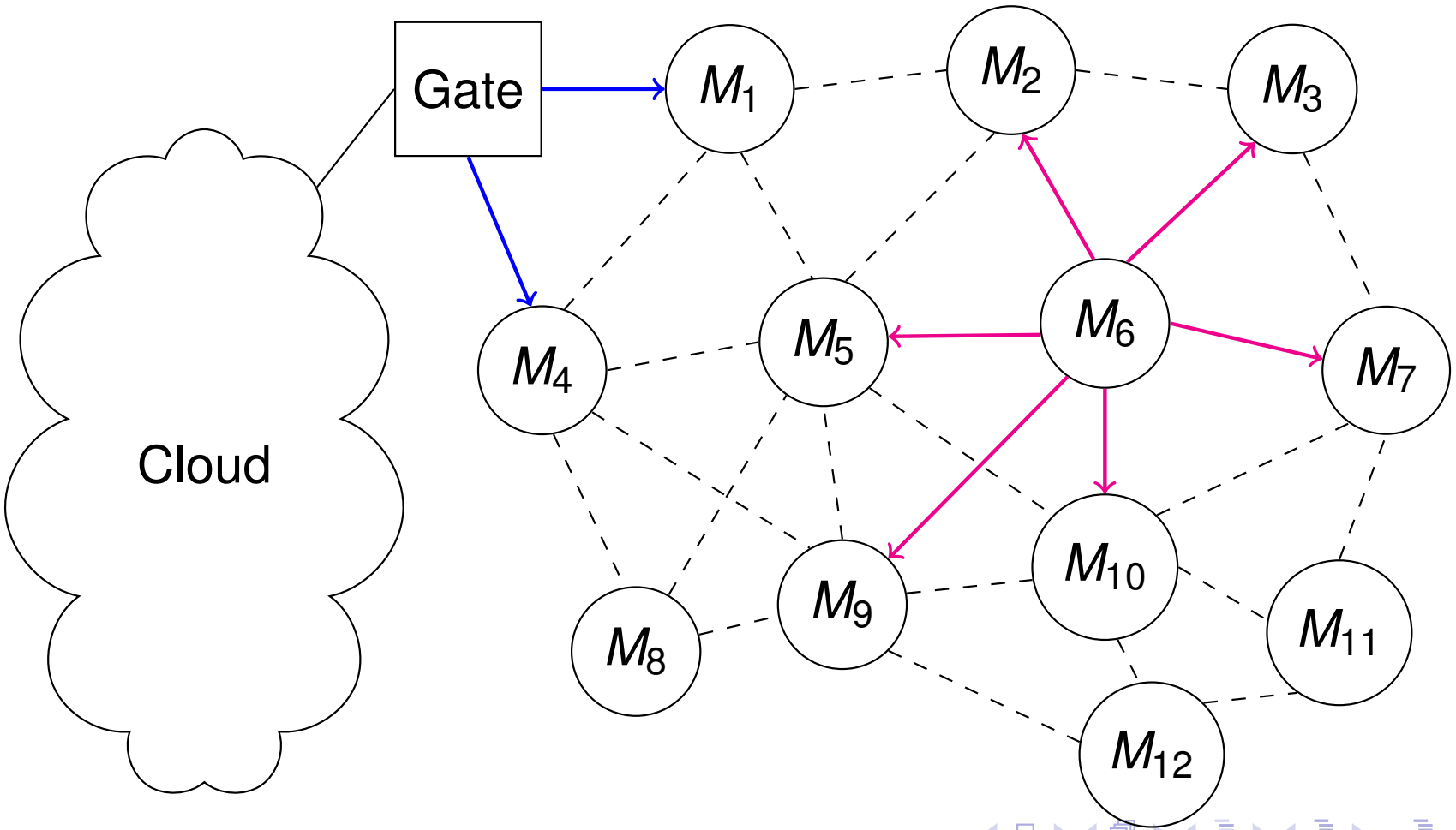
Characteristics of the specific content delivery problem of interest

- The content of interest is a large file
- The mess network of interest is a general mesh network in which each station is a mesh STA
- There is exactly one point at which content enters the mesh network
- The number of stations is large
- Multi-hop communications are required
- The content is to be delivered to every STA on the network

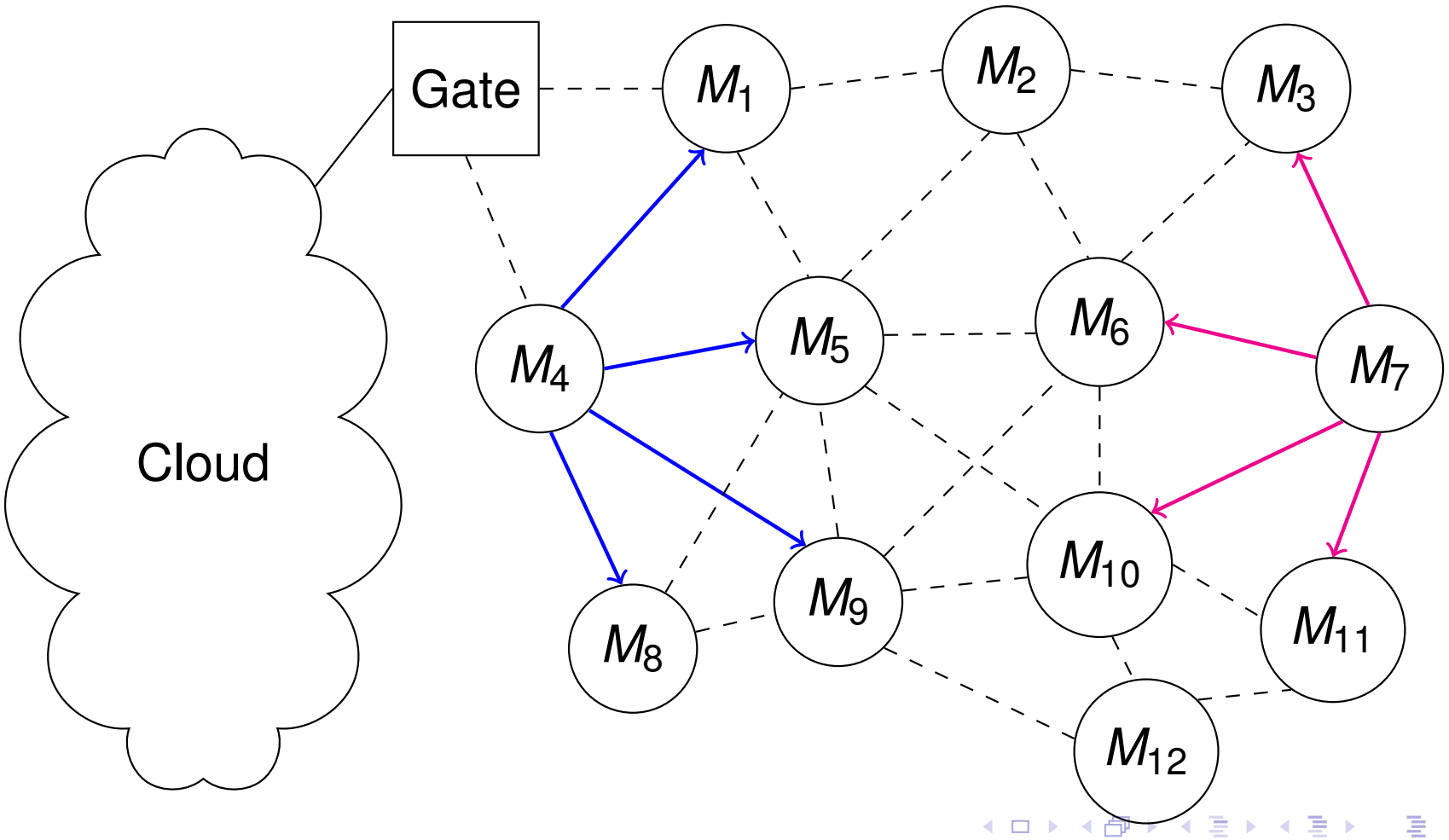
Example Mesh Network



Transmission in a Mesh Network



Transmission in a Mesh Network (Concluded)



High Level Objectives

- Achieve reliable content delivery
- Exploit multicast nature of transmissions
- Maximize effectiveness of transmissions
- Accommodate dynamic end system entry and exit
- Minimize delivery times

Content Delivery using Luby transform (LT) coding

- Content broken into chunks called *Source Symbols* (SSs)
- *Encoding Symbols* (ESs) formed by taking random linear combinations (LCs) of SSs
- ESs and LCs are sent over network to receiver
- Receiver
  - Collects ESs and LCs
  - Solves linear system (in a clever way) to recover SSs

LT Coding Simple Math

- The set of source symbols can be written as a vector: $S = [s_0, s_1, \cdots, s_{k-1}]$.
- Each encoding symbol is a binary linear combination of the source symbols, $E_i = b_i S$, where b_i is known by the receiver.
- The set of received symbols forms a set of linear equations

$$\begin{bmatrix} E_{j_1} \\ E_{j_2} \\ \vdots \\ E_{j_n} \end{bmatrix} = S \begin{bmatrix} b_{j_1} \\ b_{j_2} \\ \vdots \\ b_{j_n} \end{bmatrix} = SB_R$$

- the E_{j_ℓ} and b_{j_ℓ} are known.
- Any set of k linearly independent columns of B_R defines S

LT Coding Key Points

- Encoding symbols are generated randomly implies a random number of encoding symbols are needed to obtain k linearly independent equations.
- Received symbols containing errors are discarded
- Probabilistically, any set of n received encoding symbols is as good as any other

Many-to-many file transfer paradigm

- Source file may be available at multiple servers
- Each source prepares the file for transfer in the identical way
- A server controller receives the file request and delivers requests to a selected subset of the servers
- Each server generates encoding symbols independently drawing degrees from the same distribution
- Each destination collects encoding symbols and decodes independently
- Each receiver independently reports completion of file transfer to the server controller
- Many-to-many can be implemented with each server sending to multiple possibly different destinations

Raptor code snapshot

- Encoding
  - Erasure coding of k SSs to form n (slightly greater than k) intermediate symbols (ISs)
  - LT encode ISs to N (as many as you want) ESs and transfer along with source symbols
- Two phase decoding process
  - Use any k received ESs form the N transmitted ESs to decode all ISs
  - LT encode any missing source symbols from the ISs
- Basically changes the problem from decoding the k specific input symbols to receiving any k ESs out of N.
- RaptorQ version complexity is linear in the source block size

Advantages of RaptorQ-Based Protocols

- Complexity is linear in the size of file
- Acknowledgment free reliable delivery of content
  - Decoding requires only a specific minimum number of distinct received symbols
  - Order symbols received is irrelevant
  - Data containing errors can be discarded without requiring retransmission
  - Potential to generate basically infinite number of encoded symbols
- Accommodates dynamic network
  - End systems can join and leave group at any time
  - No special end systems in the network

Project Organization and Research Plan

- Five-member team (2 faculty and 3 graduate students)
- Develop Ruby API to Qualcomm's RaptorQ SDK
- Develop algorithms for optimal scheduling of transmissions in network
- Test ideas on a modest mesh network test bed
  - Collection of 20+ Raspberry Pi and Odroid-U2 single-board computers
  - Mac OS and MS Windows (laptops)
  - Interconnection through IEEE802.11N wireless dongles

Progress to Date

- Developed understanding of intricacies of RaptorQ Protocol and API
- Designed efficient and reliable peer–to-peer file delivery protocol
- Developed a benchmark based on the BitTorrent protocol
- Tested ideas on standalone local area network
- Conducted simulations of our new protocol versus BitTorrent
- Prepared paper for submission to IEEE ICC'15
- Developed Ruby interface to Qualcomm RaptorQ SDK
- Conducted basic tests with wireless mesh networks

Acknowledgements

The RaptorQ-Based Protocol Development Team at the University of Mississippi gratefully acknowledges the support of Qualcomm, Inc., who provides us with up-to-date version of the Qualcomm's RaptorQ Development kit. We specifically thank Michael Luby and Christian Foisy for many fruitful discussions and Valerie Wentworth for providing a pain-free channel between our team and Qualcomm.