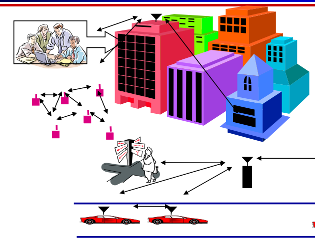
EE360: Lecture 17 Outline
Cross-Layer Design

- Announcements
  - Project poster session March 15 5:30pm (3rd floor Packard)
  - Next HW posted, due March 19 at 9am
  - Final project due March 21 at midnight
  - Course evaluations available; worth 10 bonus points
- QoS in Wireless Network Applications
- Network protocol layers
- Overview of cross-layer design
- Example: video over wireless networks
- Network Optimization
- Layering as optimization decomposition
- Distributed optimization
- Game theory

Future Network Applications



Internet (for the Z generation)
"Cellular"
Entertainment
Commerce
Smart Homes/Spaces/Structures
Sensor Networks
Automated Highways/Factories
…

Applications have hard delay constraints, rate requirements, energy constraints, and/or security constraints that must be met

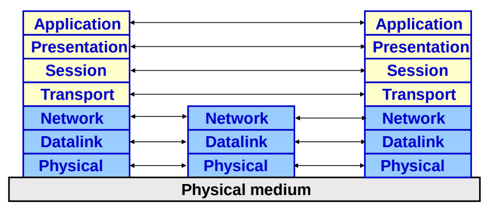
These requirements are collectively called QoS

Challenges to meeting QoS

- Underlying channels, networks, and end-devices are heterogenous
- Traffic patterns, user locations, and network conditions are constantly changing
- Hard constraints cannot be guaranteed, and average constraints can be poor metrics.
- No single layer in the protocol stack can support QoS: cross-layer design needed

A Brief Introduction to Protocol Layers

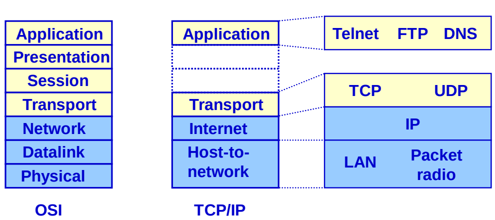
Premise: Break network tasks into logically distinct entities, each built on top of the service provided by the lower layer entities.



Example: OSI Reference Model

OSI vs. TCP/IP

- OSI: conceptually define services, interfaces, protocols
- Internet: provides a successful implementation



Layer Functionality

- Application
  - Compression, error concealment, packetization, scheduling, …
- Transport
  - End-to-end error recovery, retransmissions, flow control, …
- Network
  - Neighbor discovery and routing
- Access
  - Channel sharing, error recovery/retransmission, packetization, …
- Link
  - Bit transmission (modulation, coding, …)

Layering Pros and Cons

- Advantages
  - Simplification - Breaking the complex task of end-to-end networking into disjoint parts simplifies design
  - Modularity – Protocols easier to optimize, manage, and maintain. More insight into layer operation.
  - Abstract functionality –Lower layers can be changed without affecting the upper layers
  - Reuse – Upper layers can reuse the functionality provided by lower layers
- Disadvantages
  - Suboptimal: Layering introduces inefficiencies and/or redundancy (same function performed at multiple layers)
  - Information hiding: information about operation at one layer cannot be used by higher or lower layers
  - Performance: Layering can lead to poor performance, especially for applications with hard QoS constraints

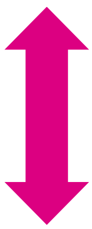
Key layering questions

- How should the complex task of end-to-end networking be decomposed into layers
  - What functions should be placed at each level?
  - Can a function be placed at multiple levels?
  - What should the layer interfaces be?
- Should networks be decomposed into layers?
  - Design of each protocol layer entails tradeoffs, which should be optimized relative to other protocol layers
- What is the alternative to layered design?
  - Cross-layer design
  - No-layer design

Crosslayer Design:

Information Exchange Across Layers

- Application
- Transport
- Network
- Access
- Link



End-to-End Metrics

Substantial gains in throughput, efficiency, and QoS can be achieved with cross-layer design

Information Exchange

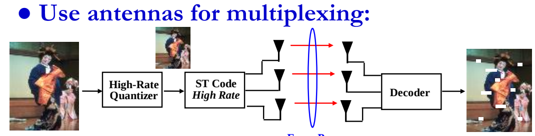
- Applications have information about the data characteristics and requirements
- Lower layers have information about network/channel conditions

Crosslayer Techniques

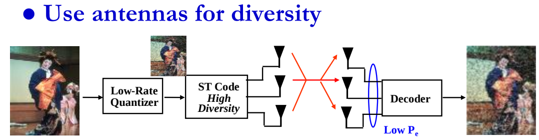
- Adaptive techniques
  - Link, MAC, network, and application adaptation
  - Resource management and allocation
- Diversity techniques
  - Link diversity (antennas, channels, etc.)
  - Access diversity
  - Route diversity
  - Application diversity
  - Content location/server diversity
- Scheduling
  - Application scheduling/data prioritization
  - Resource reservation
  - Access scheduling

Example: Video over Networks with MIMO links

- Use antennas for multiplexing:

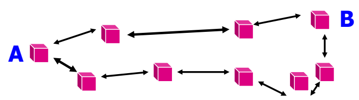


- Use antennas for diversity



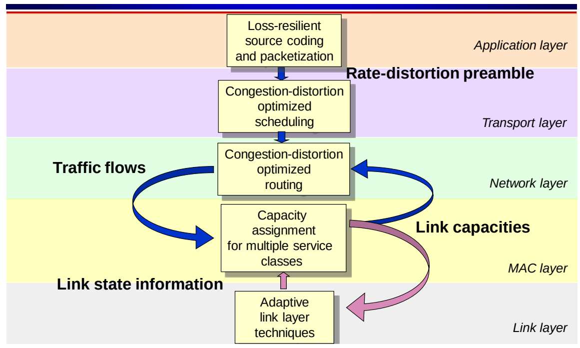
Diversity/Multiplexing/Delay Tradeoff at Links with ARQ

Delay/Throughput/Robustness across Multiple Layers

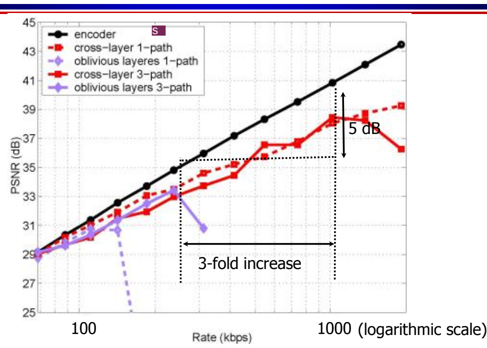


- Multiple routes through the network can be used for multiplexing or reduced delay/loss
- Application can use single-description or multiple description codes
- Can optimize optimal operating point for these tradeoffs to minimize distortion

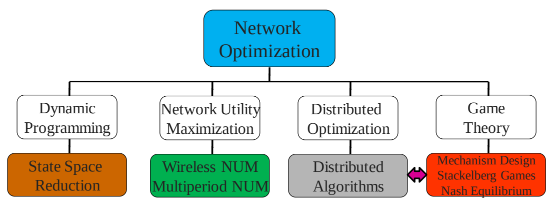
Cross-layer protocol design for real-time media



Video streaming performance



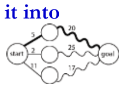
Approaches to Network Optimization*



*Much prior work is for wired/static networks

Dynamic Programming (DP)

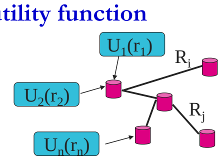
- Simplifies a complex problem by breaking it into simpler subproblems in recursive manner.



  - Not applicable to all complex problems
  - Decisions spanning several points in time often break apart recursively.
  - Viterbi decoding and ML equalization can use DP
- State-space explosion
  - DP must consider all possible states in its solution
  - Leads to state-space explosion
  - Many techniques to approximate the state-space or DP itself to avoid this

Network Utility Maximization

- Maximizes a network utility function
- Assumes
  - Steady state
  - Reliable links
  - Fixed link capacities



- Dynamics are only in the queues

$$\max \sum U_k(r_k) \quad s.t \;\; Ar \le R$$

flow k — *routing* — *Fixed link capacity*

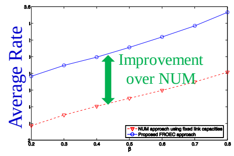
Optimization is Centralized

Wireless NUM

- Extends NUM to wireless networks
  - Random lossy links
  - Error recovery mechanisms
  - Network dynamics
- Network control as stochastic optimization

$$
\begin{aligned}
\max \quad & E[\sum U(r_m(G))] \\
\textbf{st} \quad & E[r(G)] \le E[R(S(G),G)] \\
& E[S(G)] \le \bar{S}
\end{aligned}
$$

- Can include
  - Adaptive PHY layer and reliability
  - Existence convergence properties
  - Channel estimation errors



Rethinking Layering

- How to, and how not to, layer? A question on architecture
- Functionality allocation: who does what and how to connect them?
  - More fuzzy question than just resource allocation but want answers to be rigorous, quantitative and simple
- How to quantify benefits of better modulation-codes-schedule-routes… for network applications?

The Goal

A Mathematical Theory of Network Architectures

"Layering As Optimization Decomposition:
A Mathematical Theory of Network Architectures"
By Mung Chiang, Steven H. Low, A. Robert Calderbank, John C. Doyle

Layering As Optimization Decomposition

The First unifying view and systematic approach

Network: Generalized NUM

Layering architecture: Decomposition scheme

Layers: Decomposed subproblems

Interfaces: Functions of primal or dual variables

Horizontal and vertical decompositions

NUM Formulation

- Objective function: What the end-users and network provider care about
  - Can be a function of throughput, delay, jitter, energy, congestion…
  - Can be coupled, eg, network lifetime
- Variables: What're under the control of this design
- Constraint sets: What're beyond the control of this design. Physical and economic limitations. Hard QoS constraints (what the users and operator must have)

Layering

Give insights on both:

- What each layer can do (Optimization variables)
- What each layer can see (Constants, Other subproblems' variables)

Connections With Mathematics

- Convex and nonconvex optimization
- Decomposition and distributed algorithm

Primal Decomposition

Simple example: $x + y + z + w \le c$

Decomposed into:

$$\begin{aligned} x + y &\le \alpha \\ z + w &\le c - \alpha \end{aligned}$$

New variable α *updated by various methods*

Interpretation: Direct resource allocation (not pricing-based control)

Dual-based Distributed Algorithm

NUM with concave smooth utility functions:

Convex optimization with zero duality gap

Lagrangian decomposition:

$$\begin{aligned} L(\mathbf{x}, \boldsymbol{\lambda}) &= \sum_s U_s(x_s) + \sum_l \lambda_l \left(c_l - \sum_{s: l \in L(s)} x_s \right) \\ &= \sum_s \left[U_s(x_s) - \left(\sum_{l \in L(s)} \lambda_l \right) x_s \right] + \sum_l c_l \lambda_l \\ &= \sum_s L_s(x_s, \lambda^s) + \sum_l c_l \lambda_l \end{aligned}$$

Dual problem:

$$\begin{aligned} &\text{minimize} && g(\boldsymbol{\lambda}) = L(\mathbf{x}^*(\boldsymbol{\lambda}), \boldsymbol{\lambda}) \\ &\text{subject to} && \boldsymbol{\lambda} \succeq 0 \end{aligned}$$

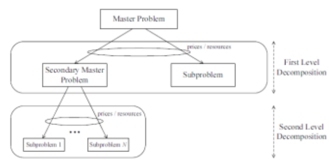
Horizontal vs Vertical Decomposition

- Horizontal Decompositions
  - Reverse engineering: *Layer 4 TCP congestion control: Basic NUM (LowLapsley99,* RobertsMassoulie99, MoWalrand00, YaicheMazumdarRosenberg00, etc.)
  - Scheduling based MAC is known to be solving max weighted matching
- Vertical Decompositions
  - Jointly optimal congestion control and adaptive coding or power control (Chiang05a)
  - Jointly optimal routing and scheduling (KodialamNandagopal03)
  - Jointly optimal congestion control, routing, and scheduling (ChenLowChiangDoyle06)
  - Jointly optimal routing, resource allocation, and source coding(YuYuan05)

Alternative Decompositions

Many ways to decompose:

- *Primal Decomposition*
- *Dual Decomposition*
- *Multi-level decomposition*
- *Different combinations*



Lead to alternative architectures with different engineering implications

Key Messages

- Existing protocols in layers 2,3,4 have been reverse engineered
- Reverse engineering leads to better design
- Loose coupling through layering price
- Many alternatives in decompositions and layering architectures
- Convexity is key to proving global optimality
- Decomposability is key to designing distributed solution
- Still many open issues in modeling, stochastic dynamics, and nonconvex formulations
- Architecture, rather than optimality, is the key

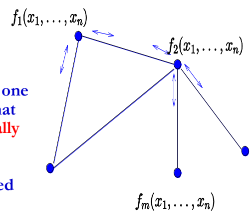
Other Extensions

- On-line learning
- Hard delay constraints (not averages)
- Traffic dynamics
- Distributed optimization

Distributed and Asynchronous Optimization of Networks

- Consider a network consisting of m nodes (or agents) that cooperatively minimize a common additive cost (not necessarily separable)

$$\text{minimize} \quad \sum_{i=1}^{m} f_i(x)$$
$$\text{subject to} \quad x \in \mathcal{R}^n,$$



- Each agent has information about one cost component, and minimizes that while exchanging information locally with other agents.
- Model similar in spirit to distributed computation model of Tsitsiklis
- Mostly an open problem. Good distributed tools have not yet emerged

Game Theory

- Game theory is a powerful tool in the study and optimization of both wireless and wired networks
  - Enables a flexible control paradigm where agents autonomously control their resource usage to optimize their own selfish objectives
  - Game-theoretic models and tools provide potentially tractable decentralized algorithms for network control
- Most work on network games has focused on:
  - Static equilibrium analysis
  - Establishing how an equilibrium can be reached dynamically
  - Properties of equilibria
  - Incentive mechanisms that achieve general system-wide objectives
- Distributed user dynamics converge to equilibrium in very restrictive classes of games; potential games is an example
- Examples: power control; resource allocation

Key Questions

- What is the right framework for crosslayer design?
- What are the key crosslayer design synergies?
- How to manage crosslayer complexity?
- What information should be exchanged across layers, and how should this information be used?
- How to balance the needs of all users/applications?

Summary: To Cross or not to Cross?

- With cross-layering there is higher complexity and less insight.
- Can we get simple solutions or theorems?
  - What asymptotics make sense in this setting?
  - Is separation optimal across some layers?
  - If not, can we consummate the marriage across them?
- Burning the candle at both ends
  - We have little insight into cross-layer design.
  - Insight lies in theorems, analysis (elegant and dirty), simulations, and real designs.

Presentation

- "Cross-Layer Wireless Multimedia Transmission: Challenges, Principles, and New Paradigms"
- By Mihaela Van Scharr, Sai Shankar
- *Presented by Chris Li*