1

A Systematic Framework for Cross-layer Optimization in Dynamic Communication Systems

Mihaela van der Schaar and Fangwen Fu

email: {mihaela, fwfu}@ee.ucla.edu

www-page: medianetlab.ee.ucla.edu

2

Networked multimedia apps

Delay-sensitive applications

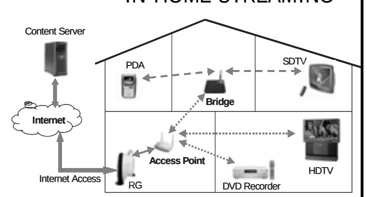
- Entertainment
- P2P TV
- Emergency services
- Surveillance
- Telemedicine
- Multi-party videoconferencing
- Telepresence
- Augmented reality
- Distributed gaming

Communication over packet-based networks

Wired: Internet, overlay networks, P2P

Wireless: Cellular, WLANs, Cognitive Radios

IN-HOME STREAMING



STARBUCKS

WAYPORT

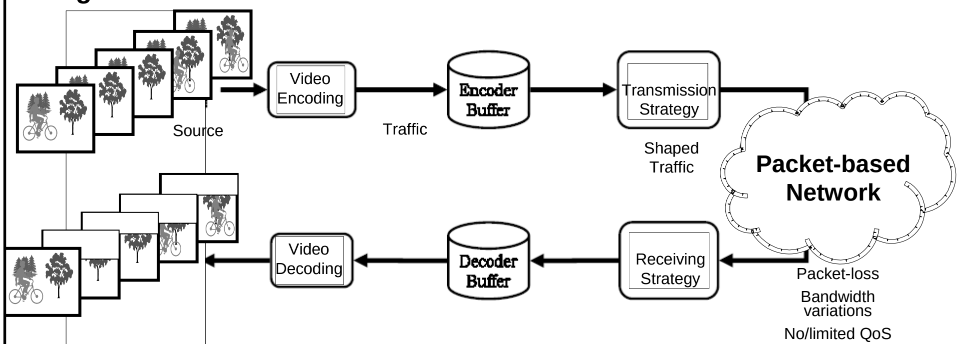
MEETINGS

3

Multimedia Streaming over Internet and wireless networks

Transmitter/Server/Proxy

Original VIDEO



Received VIDEO

Receiver/Client

4

Multimedia Streaming Challenges - Effect of Transmission Errors

Coded,
No loss

3%

5%

10%

Example: reconstructed video frames from a H.264 coded sequence, subject to packet losses

5 **Packet networks (e.g. wireless networks) are heterogeneous in bandwidth, reliability and receiver device characteristics ☹**

- Packets Loss
  - Delayed (propagation and queuing - variable, delay jitter)
    - Excessive delay = loss for real-time applications
  - Lost
    - Inevitable under “best effort” delivery
    - Packet losses can vary from 0.1% to 10% or more
    - Time varying characteristics
    - Difficult to characterize and measure
  - Discarded (at the receiver side)
    - If the complexity/power of the receiver is limited
- Bandwidth fluctuation
  - multipath fading, co-channel interference, noise, mobility, handoff, etc
  - competing traffic
- Receiver architecture heterogeneity
  - Computing capability, buffer availability, display resolution, power limitation (transmission and processing)

6 **Multimedia Streaming – Other Challenges**

Delay-sensitive applications:

•Media download/real-time streaming/interactive two-way communication

•Pre-encoded (stored) video/Interactive/real-time or non-real-time

High bandwidth requirements:

•Standard definition TV: at least 3 Mbps

•High definition TV: at least 10 Mbps (Blu-Ray: 12-14 Mbps)

Time-varying “environment” (dynamics):

•Multimedia source characteristics

•Multimedia traffic characteristics (bursty) – depends on codec used and its configuration (conventional traffic/queuing models can often not be used)

•Network/channel characteristics – wired/wireless

Heterogeneous and time-varying (dynamic) system or user constraints: power constraints, diverse usage scenarios, user preferences etc.

7 **Multimedia Streaming – Other Challenges (continued)**

• Packet-based communication networks (wired/wireless) provide **limited or no QoS** for multimedia applications

•**Coupling between users** is important (e.g. short term increase of power may be beneficial in the short term, but have detrimental effects in the long term) -> (dynamic) **multi-user interaction**

•**Tradeoff between efficiency and fairness**

•*Etc…..*

How to cope with these challenges?

Can the various layers cooperate to help us solve these challenges?

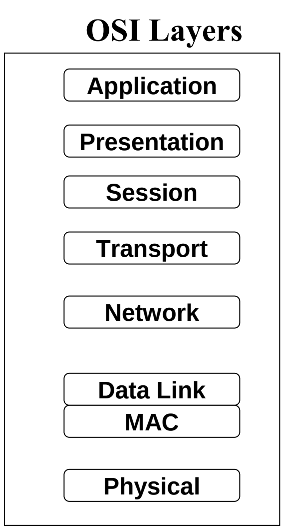
8 **Protocols**

- Protocol = a set of standards defining “message” formats & exchange rules
- In multimedia communications, protocols are used for (examples):
  - mapping bitstreams to packets
  - controlling the delivery, protection and other aspects of networked communications
- **Open Systems Interconnection (OSI)** protocol stack:
  - Physical layer (1): channel characteristics
  - Data Link layer (2): framing, error control, multi-user interaction
  - Network layer (3): addressing and routing
  - Transport layer (4): end-to-end reliability, flow control
  - Session layer (5): establishing a communication session
  - Presentation layer (6): compression, data representation
  - Application layer (7): applications: file transfer, streaming, but also end-to-end reliability (even routing!), traffic characterization, traffic shaping, prioritization, etc.

9 OSI protocol stack

- Why protocols? Important to develop a protocol/standard based on which devices can inter-operate
- Standardization – depends on the layer
  - Examples
    - MAC/PHY: 802.11 Wireless LANs (WLANs)a
    - Network Layer/Transport Layer: IETF (Internet Engineering Task Force) - IETF is the protocol engineering and development arm of the Internet.
    - Application Layer: ISO MPEG, ITU H.26x
    - Across Layers: 3GPP, ISMA
- Examples of protocols at the various layers
  - PHY: 802.11a
  - MAC: 802.11a, 802.11e
  - Network: IP
  - Transport: UDP, TCP
  - Application layer: H.264, RTP etc.
- Unfortunately, de facto standards also exist – examples?

10 Illustrative support and adaptation parameters for multimedia streaming in the various layers/protocols



- **RF**
  - Transmit power
  - Antenna direction
- **Baseband**
  - Modulation
  - Equalization
- **Link/MAC**
  - Frame length
  - Error correction coding
  - ARQ
  - Admission Control and Scheduling
  - Packetization
- **Transport/Network**
  - Signaling
  - TCP/UDP
  - Packetization
- **Application**
  - Compression strategies
  - Concealment, Post-processing etc.
  - Rate/Format adaptation
  - Channel coding/ARQ
  - Number of streams/flow
  - Scheduling
  - Packetization

11 **Can protocols alone provide optimal solutions for media communications?**

- NO!!
- Remember, a protocol = a set of standards defining "message" formats & exchange rules, but not how to select the algorithms, parameters, optimizations of the protocol!

We need cross-layer design and optimization

12 **Why cross-layer design? A motivation**

- In layered network architectures such as the Open Systems Interconnection (OSI) model, the functionality of each layer is specified in terms of the services that it receives from the layer(s) below and that it is required to provide to the layer(s) above.
- The advantage of layered architectures is that the designer or implementer of the protocol or algorithm at a particular layer can focus on the design of that layer, without being required to consider all the parameters and algorithms of the rest of the stack.
- However, in current layered network implementations, each layer often optimizes its strategies and parameters individually.

- > This generally results in sub-optimal performance for the users/applications

13 **Why cross-layer design? A motivation (cont.)**

- Many cross-layer optimization solutions have been proposed in recent years to improve the performance of network users operating in a time-varying, error-prone wireless environment.
- These solutions optimize the protocol parameters in an integrated fashion by jointly and simultaneously considering the dynamics at each layer and requiring layers to provide access to their internal protocol parameters to other layers.

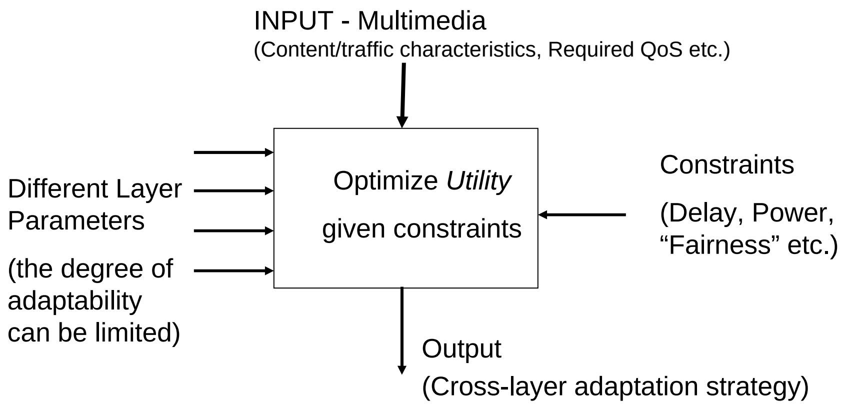
14 **Why Cross-Layer Design and Optimization? Summary**

Cross-layer design and optimization is essential because:

- it leads to improved multimedia performance over existing wireless networks;
- It provides guidelines for designing and optimizing the inter-layer message exchanges (middleware);
- it provides valuable insights on how to design the next generation algorithms and protocols for wireless multimedia systems.

15

Conceptual Framework (System View of Cross Layer Optimization)

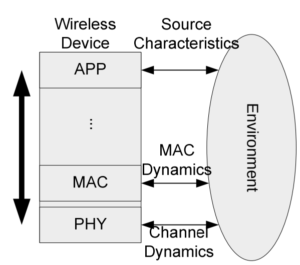


- **Utility:** multimedia quality, power, system-wide network utilization etc.
- Cross-layer problem = *Challenging Multi-Objective Optimization*

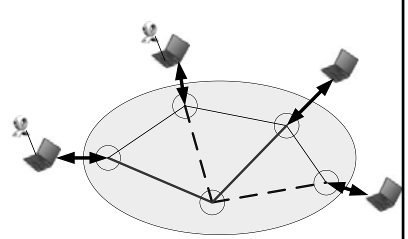
16

Classes of cross-layer optimization solutions

- User-level Cross-layer Optimization
  - Explicitly consider the user’s constraints on the adaptation
    - Adapt the application to channel
    - Adapt the channel to the application



- Network-level Cross-layer Optimization
  - Jointly consider
    - Power allocation
    - Multiple access
    - Routing selection
    - Congestion control
    - Packet prioritization
  - Network Utility Maximization (NUM)



17 **Focus of this tutorial: Systematic Framework for Cross-layer optimization in Dynamic Networks**

Motivation:
Existing theory

- **Information and coding theory** [Shannon and beyond]
  - “ideal” point-to-point communication setting
  - simplistic source models -> not accurate for multimedia coders
  - no delay constraints (concept of “streaming” is absent)
  - no resource management issues and policies such as fairness, etc.
  - system issues neglected – essential for *realistic* wireless multimedia communications
- **Optimization (Dynamic!), Control, Learning**

18 **Motivation - continues**

Concepts, theories and solutions that have dominated the information theory and communications areas need to be adapted for

- **time-varying source and channel characteristics,**
- **dynamic and delay-sensitive multimedia applications (Not all bits are created equal ☺)**
- **multi-user transmission environments.**

Conventional signal processing and source coding methods do not consider

- **environmental dynamics**
- **system/protocol (also layering) constraints**
- **multi-user transmission environments.**

Unique constraints of *multimedia applications* change fundamental communication design principles

19

This tutorial (compared to past cross-layer design tutorials)

Steady state => Dynamic behavior

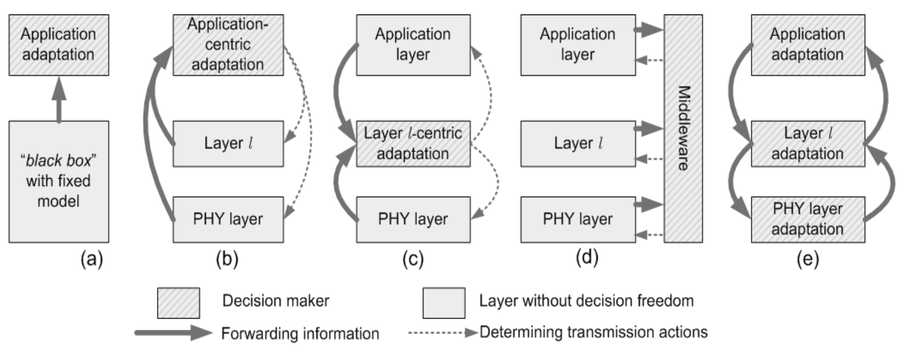
Dynamics => Foresighted rather than myopic cross-layer optimization

Repeated multi-user interaction => Stochastic multi-user interaction (considers coupling between users)

20

Conceptual illustration of cross-layer optimization methods:

(a) application adaptation;
(b) application-centric adaptation;
(c) middle layer-centric adaptation;
(d) middleware-based adaptation;
(e) autonomous adaptation with information exchange

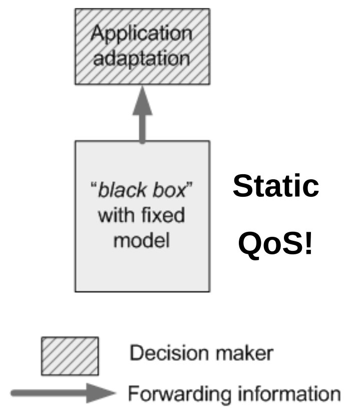


21

A simple, but sub-optimal solution: The application does it all

☺ Independently adapts parameters at each layer
☺ Obeys layered network architecture

☹ Has poor performance for delay-sensitive applications in wireless environments (lower layers do not interact with application layer, but merely "insulate" the application from the network)
☹ Poor network utilization



22

Can good old "QoS" paradigms help?

Integrated services:

- Applications can reserve end-to-end bandwidth
  - Need to deploy protocol that reserves bandwidth
  - Must modify scheduling policies in routers/access points to honor reservations
  - Application must provide the network with a description of its traffic, and must further abide to this description.
- Where implemented today? Examples?

- RSVP (Internet)
- 802.11e HCCA, 802.11a PCF

Differentiated services:

- Differentiated services for different classes
- User pays more to send/receive higher class of packets.
  - What happens is user does not pay more? Tragedy of commons!
- Packets are marked with different priority.
- Where implemented today? Examples?

- 802.11e EDCA

23 Illustrative solution for application-driven optimization

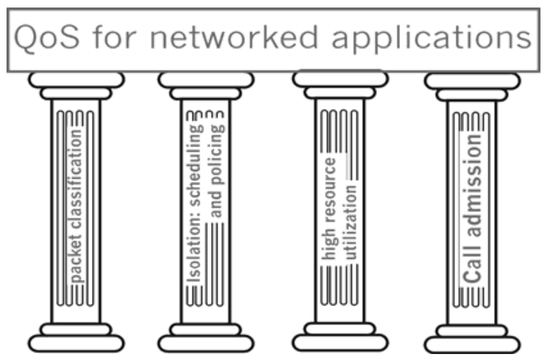
Step 1. Perform admission control/QoS request (e.g. App+ Network)

- to ensure high/fair resource utilitization – define fair/efficient resource division rules

Step 2. After application is admitted or if no admission control:

Network: Schedule and police the users

Applications: Packet classification and scheduling



24 Step 1: Admission Control

- Session must first declare its QoS requirement and characterize the traffic it will send through the network
- **T-spec**: defines the traffic characteristics
- Who generates the TSPEC parameters?
  - Can be generated by the application
  - Can generated autonomously by the network if the TSPEC parameters are not available from higher layers based on traffic measurements
- What TSPEC parameters are used for admission control?
  - Application layer (traffic) parameters
    - Mean rate ρ
    - Delay d (determined by the application)
    - Maximum burst size σ
    - Peak rate P
  - Network parameters
    - Channel capacity available C

25

How to make admission control decisions based on TSPEC parameters?

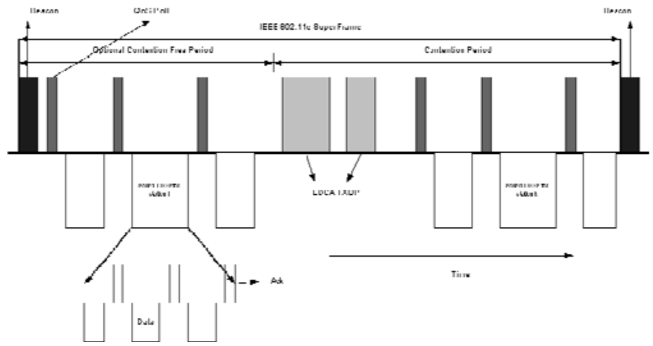
- Determine Effective Bandwidth, e_i ,of a Stream
- Make Admission Control Decision from the following equation

$$\Sigma\, e_i <= C$$

- If the new request asks for more capacity than available in channel then reject it.

802.11e

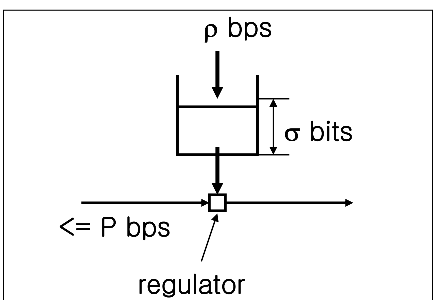
example

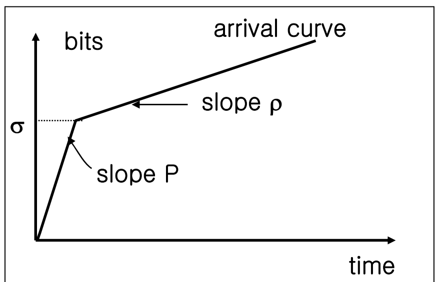


26

Flow Specification using Token Bucket

- A flow is characterized by three parameters (σ,ρ,P)
  - σ – token depth
  - ρ – average rate
  - P - peak rate

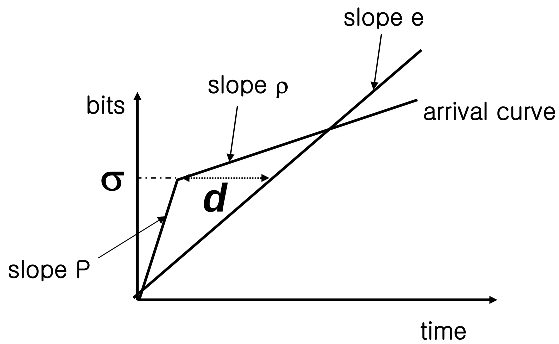




Arrival curve – maximum amount of bits transmitted by time t

27 Effective Bandwidth Calculation

- Given (σ,ρ,P) and target delay ***d***
- Determine effective bandwidth: Minimum bandwidth required for lossless multiplexing that satisfies the given QoS requirement of *d*)

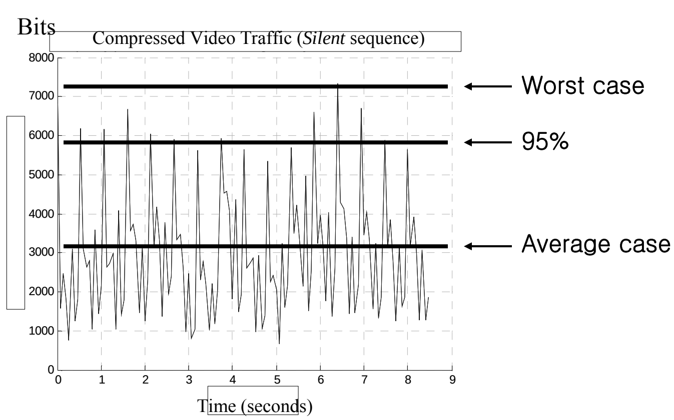


$$e = \frac{P}{(1 + d\,\frac{P-\rho}{\sigma})}$$

28 How to determine the TSPEC parameters?

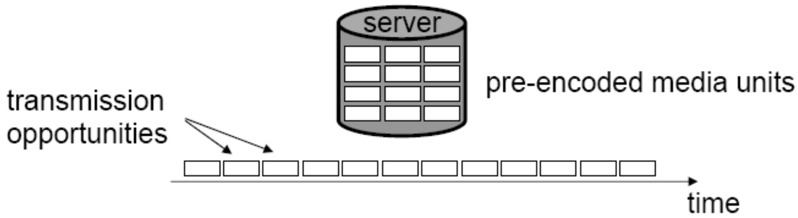
Traffic models

- Flow-based characterization of the traffic
  - determines high-level parameters of the traffic specification
  - used for flow-based admission control etc. (e.g. TSPEC)



29

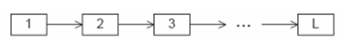
Step 2: Scheduling of multimedia packets



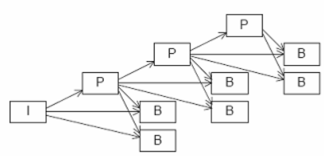
- Media unit is put into packet for transmission
- Packet may be retransmitted or sent multiple times
- Requirements
  - Meet target rate
  - Maximize reconstruction quality
- Packet scheduling problem: which packets should be selected for transmission and when?
- For rate-adaptation, scheduling (after admission, or if only Diff-Serv is implemented), error protection etc., we need a packet-based characterization of the traffic

30

Packet-based characterization of the traffic – using Direct Acyclic Graphs (DAGs)



(a) Sequential Dependencies

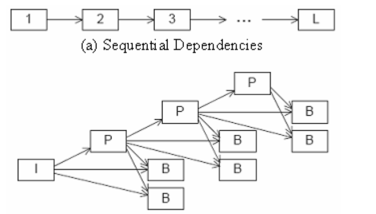


(a) Typical Hybrid Coder Dependencies (MPEG-2, H.264/AVC)

- Display deadline depends on display order
- Decoding deadline depends on DAG and display order

GOP (a)	**I**	**P**	**P**	**P**	**P**	**P**	**I**
Display	T_0	T_0 +T	T_0 +2T	T_0 +3T	T_0 +4T	T_0 +5T	T_0 +6T
Decode	T_0	T_0 +T	T_0 +2T	T_0 +3T	T_0 +4T	T_0 +5T	T_0 +6T
GOP (b)	**I**	**B**	**B**	**P**	**B**	**B**	**I**
Display	T_0	T_0 +T	T_0 +2T	T_0 +3T	T_0 +4T	T_0 +5T	T_0 +6T
Decode	T_0	T_0 +T	T_0 +2T	T_0 +T	T_0 +4T	T_0 +5T	T_0 +4T

31 Computing distortion impacts



(a) Sequential Dependencies

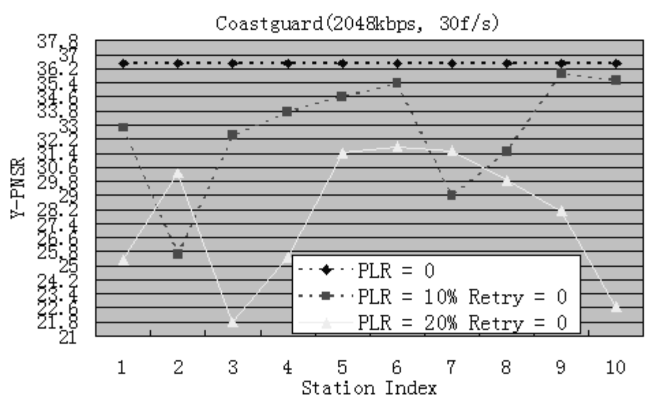
(a) Typical Hybrid Coder Dependencies (MPEG-2, H.264/AVC)

- Distortion impact of a frame depends on
  - Number of dependent frames
  - Correlation with neighboring frames

GOP (a)	**I**	**P**	**P**	**P**	**P**	**P**	**I**	**P**	**P**	**P**	**P**	**P**
No. Dependent Frames	5	4	3	2	1	0	5	4	3	2	1	0
GOP (b)	**I**	**B**	**B**	**P**	**B**	**B**	**I**	**B**	**B**	**P**	**B**	**B**
No. Dependent Frames	5	0	0	4	0	0	7	0	0	4	0	0

32 How good is the performance?

PSNR performance of different video streams.

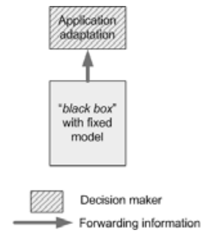


10 stations are admitted. TSPEC info is below.

Flow	Mean date rate (kbps)	Delay bound (ms)	Nominal MSDU size (octets)	Maximum MSDU size (octets)	Maximum burst size (octets)	Peak data rate (kbps)	Effective bandwidth (kbps)	TXOP (ms)	User priority
Single flow	2048	200	2048	2048	78858	2915	2295	7.5	5

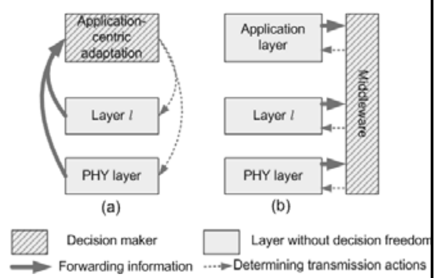
33

From single-layer optimization to cross-layer optimization



Single-layer optimization
☹Has poor performance for delay-sensitive applications in wireless environments
☹ Poor network utilization

Current cross-layer optimization
☺ Jointly optimizes parameters across multiple layers to improve performance
☹ Violates the layered architecture
☹ Companies have no freedom to design their own protocols

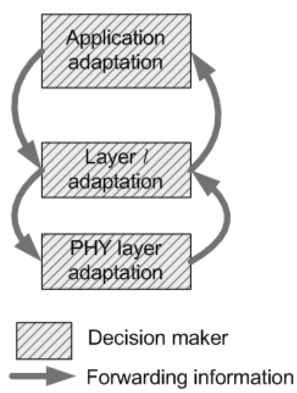


Can we achieve optimal performance while obeying the layered network architecture?

33

34

Systematic layered optimization with information exchange



Key questions to be answered:
1. How can each layer optimally configure its own parameters?
2. What information should be exchanged between layers and how?

35 **Challenges for cross-layer design and optimization**

- Decision making - coupled among layers
- Environmental dynamics at various layers (different time scales)
- Adaptation granularity (multiple time scales/granularities at different layers)
- How to exchange information among layers? What information should be exchanged?
- Which layer/layers should perform the optimization?
- Protocol compliant? Protocols are determined and controlled by different entities/companies
- Violate OSI stack?

36 **Other Challenges**

- Current cross-layer optimization violates layered architecture
  - Centralized
  - Lead to dependent layer design
  - Reduce network stability and extensibility
- Decision maker requires to know
  - All possible strategy combination
  - Dynamics from different layers
- Objective
  - Myopic performance (maximizing current utility) or (also) impact on the future performance?

Why do we care about future performance at the current time?

Current decisions impact immediate *and* future performance

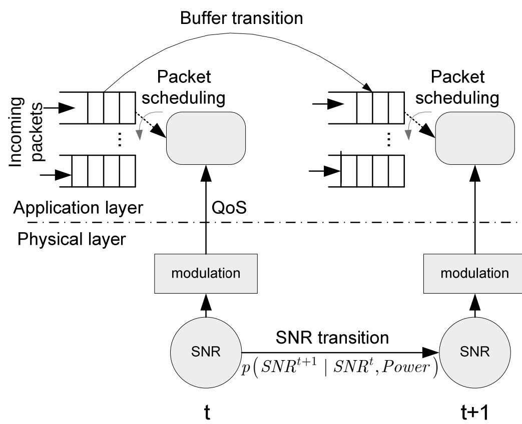
•Video Rate & Distortion: Coding decisions for current video data unit impact bit-budget, rate, and distortion of future data units

•Delay: Time to transmit current video packets impacts time available to transmit future video packets

37

Cross-layer optimization

Current action impacts both immediate as well as future reward



State: buffer fullness
Actions: packet scheduling, source coding
State transition: buffer transition
Reward: distortion reduction

State: SNR
Actions: modulation, power
State transition: SNR transition
Cost: consumed power

Dramatic improvements in network resource utilization can be achieved using a *foresighted* (long-term) optimization.

38

How to make optimal foresighted decisions? Markov decision process (MDP)

- Discrete-time stochastic *control* process
- Extension of Markov chains
- Differences:
  - Addition of actions (choice)
  - Addition of rewards (goal)
- If the actions are fixed, an MDP reduces to a Markov chain

39 **Discrete MDP model**

Discrete MDP model:

- Time t is discrete.
- State space S.
- Set of actions A.
- Reward function $R : S \times A \rightarrow \mathbb{R}$.
- Transition model $p(s'|s, a)$, $S \times A \rightarrow \Delta(S)$.
- Initial state s_0 is drawn from $\Delta(S)$.

The Markov property entails that the next state s_{t+1} only depends on the previous state s_t and action a_t:

$$p(s_{t+1}|s_t, s_{t-1}, \ldots, s_0, a_t, a_{t-1}, \ldots, a_0) = p(s_{t+1}|s_t, a_t). \quad (1)$$

40 **Rewards and optimality criterion**

Agent should maximize

$$E\Big[\sum_{t=0}^{h} \gamma^t R_t\Big], \quad (2)$$

where

- h is the planning horizon, can be finite or ∞
- γ is a discount rate, $0 \leq \gamma < 1$ Myopic vs. foresighted decisions

Reward hypothesis (Sutton and Barto, 1998):
All goals and purposes can be formulated as the maximization of the cumulative sum of a received scalar signal (reward).

Example?

41 **Why do we care about future performance at the current time?**

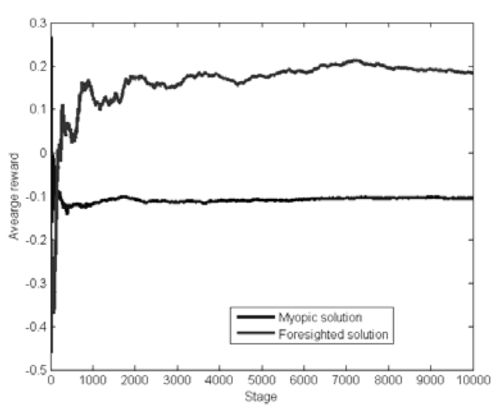
- Current decisions impact immediate *and* future performance
  - Video Rate & Distortion: Coding decisions for current video data unit impact bit-budget, rate, and distortion of future data units
  - Delay: Time to transmit current video packets impacts time available to transmit future video packets

Other examples?

Dramatic improvements in network resource utilization can be achieved using a *foresighted* (long-term) optimization.

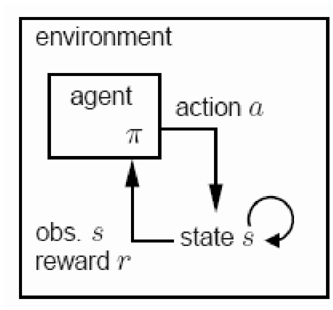
42 **Foresighted & myopic solutions**

- Foresighted & myopic



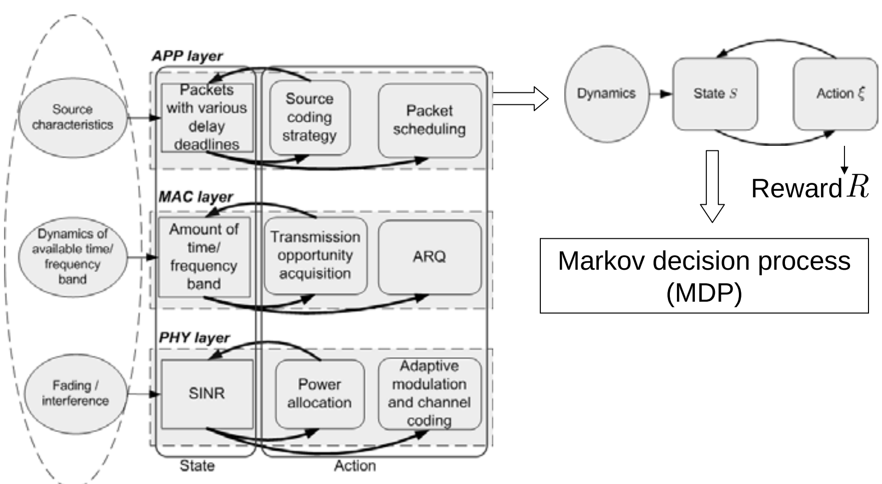
43 **Solution to an MDP = Policy π**

- Gives the action to take from a given state regardless of history
- Goal: Find a policy that maximizes the cumulative discounted sum of rewards



44

MDP for cross-layer optimization - Example



45 **Brief summary of concepts**

- The *agent* and its *environment* interact over a sequence of discrete time steps.
- The specification of their interface defines a particular task:
  - the *actions* are the choices made by the agent;
  - the *states* are the basis for making the choices;
  - the *rewards* are the basis for evaluating the choices.
- A *policy* is a stochastic rule by which the agent selects actions as a function of states.
- The agent's objective is to maximize the amount of reward it receives over time.

46 **Policies and value**

An agent acts according to its policy

$$\pi : S \rightarrow A. \qquad (3)$$

A common way to characterize a policy is by its $\boxed{\text{value function:}}$

Quantifies how good it is to be in a state

$$V^{\pi}(s) = R(s, \pi(s)) + E\Big[\sum_{t=1}^{\infty} \gamma^t R(s_t, \pi(s_t))\Big]. \qquad (4)$$

State-value function

47 **Policies and value**

Extracting a policy π from a value function V is easy:

$$\pi(s) = \arg\max_{a \in A} \left[R(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V(s') \right]. \qquad (6)$$

Bellman (1957) equation:

$$V^*(s) = \max_{\pi} V^{\pi}(s)$$

$$V^*(s) = \max_{a \in A} \left[R(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^*(s') \right], \qquad (7)$$

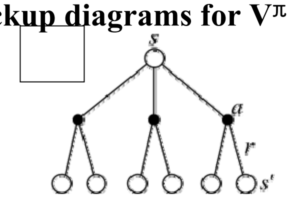
Optimal state-value function **Known**

48 **Bellman optimality equation**

$$V^*(s) = \max_{a \in \mathcal{A}(s)} \sum_{s'} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma V^*(s') \right]$$

- Expresses the relationship between the value of a state and the values of its successor states.

Backup diagrams for V^{π}



49 **Bellman optimality equation**

- For finite MDPs, the Bellman optimality equation has a unique solution independent of the policy.
- The Bellman optimality equation is actually a system of equations, one for each state, so if there are N states, then there are N equations in N unknowns.
- If the dynamics of the environment are known, then in principle one can solve this system of equations for V* using any one of a variety of methods for solving systems of nonlinear equations.
- Once one has V*, it is relatively easy to determine an optimal policy.
- For each state s, there will be one or more actions at which the maximum is obtained in the Bellman optimality equation. Any policy that assigns nonzero probability only to these actions is an optimal policy.
- You can think of this as a one-step search. If you have the optimal value function, V*, then the actions that appear best after a one-step search will be optimal actions.
- Another way of saying this is that any policy that is *greedy* with respect to the optimal evaluation function V* is an optimal policy.

50 **Bellman optimality equation**

- The beauty of V* is that if one uses it to evaluate the short-term consequences of actions--specifically, the one-step consequences--then a greedy policy is actually optimal in the long-term sense in which we are interested because V* already takes into account the reward consequences of all possible future behavior.
- By means of V*, the optimal expected long-term return is turned into a quantity that is locally and immediately available for each state.
- Hence, a one-step-ahead search yields the long-term optimal actions.

$$V^*(s) = \max_{a\in A}\left[R(s,a) + \gamma \sum_{s'\in S} p(s'|s,a)V^*(s')\right]$$

$$V^*(s) = \max_{\pi} V^{\pi}(s)$$

$$\pi(s) = \arg\max_{a\in A}\left[R(s,a) + \gamma \sum_{s'\in S} p(s'|s,a)V(s')\right]$$

51

Brief summary of concepts

- A policy's *value function* assigns to each state the expected return from that state given that the agent uses the policy.
- The *optimal value function* assigns to each state the largest expected return achievable by any policy.
- A policy whose value functions are optimal is an *optimal policy*.
- The optimal value functions for states are unique for a given MDP, but there can be many optimal policies.
- Any policy that is *greedy* with respect to the optimal value functions must be an optimal policy.
- The *Bellman optimality equations* are special consistency conditions that the optimal value functions must satisfy and that can, in principle, be solved for the optimal value functions, from which an optimal policy can be determined with relative ease.

52

How do we compute the optimal state-value function and the optimal policy?

- Problem:
  - Given:
    - Transition probability function: $p\left(s' \mid s, a\right)$
    - Reward function: $R(s, a)$
  - Determine:
    - Optimal state-value function: V^*
    - Optimal policy: π^*

Solution:

Dynamic Programming

53

Dynamic Programming

- The term dynamic programming (DP) refers to a collection of algorithms that can be used to compute optimal policies given a perfect model of the environment
  - Can be used to solve Markov decision processes.
  - Use value functions to organize and structure the search for good policies.
  - Turn Bellman equations into update policies.
- The classical view is not so useful in practical multimedia communication problems since we rarely have a perfect environment model
  - However, it provides foundation for other methods
- Not practical for large problems

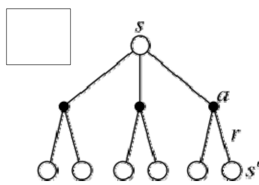
54

DP methods

- *Policy evaluation* refers to the (typically) iterative computation of the value functions for a given policy.
- *Policy improvement* refers to the computation of an improved policy given the value function for that policy.
- Putting these two computations together, we obtain *policy iteration* and *value iteration*, the two most popular DP methods.
  - Either of these can be used to reliably compute optimal policies and value functions for finite MDPs given complete knowledge of the MDP.

55 **Full backups**

- Classical DP methods operate in sweeps through the state set, performing a *full backup* operation on each state.
  - Each backup updates the value of one state based on the values of all possible successor states and their probabilities of occurring.
  - Full backups are closely related to Bellman equations: they are little more than these equations turned into assignment statements.
  - When the backups no longer result in any changes in value, convergence to values that satisfy the corresponding Bellman equation has occurred.



Backup diagram for V^π

$$V(s) \leftarrow \sum_a \pi(s,a) \sum_{s'} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma V(s')\right]$$

56 **Policy evaluation**

- Iterative *policy evaluation* using full backups

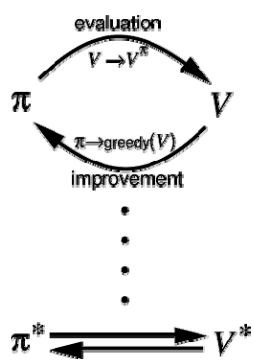
Input π, the policy to be evaluated
Initialize $V(s) = 0$, for all $s \in \mathcal{S}^+$
Repeat
  $\Delta \leftarrow 0$
  For each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_a \pi(s,a) \sum_{s'} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma V(s')\right]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$ (a small positive number)
Output $V \approx V^\pi$

57 Policy improvement

- When should we change the policy?
- If we pick a new action α from state *s* and thereafter follow the current policy and $V(\pi') >= V(\pi)$, then picking α from state *s* is a better policy overall.
- Results from the *policy improvement theorem*

58 Iterative DP algorithms

- All iterative algorithms for computing the optimal state-value function and optimal policy have two basic steps:
  - 1. Policy Improvement
  - 2. Policy Evaluation



2 basic steps

$$\pi(s) := \arg\max_a \sum_{s'} P_a(s,s')V(s') \quad 1$$

$$\boxed{V(s) := R(s) + \sum P_{\pi(s)}(s,s')V(s')} \quad 2$$

Value Function

Next, several algorithms that apply these two basic steps
(in different orders) for computing optimal value function and optimal policy

59 **Value iteration**

can be used to compute optimal policies and value functions

Value iteration: successive approximation technique.
The optimal value function V_0^*

$$V_0^*(s) = \max_{a \in A} R(s, a). \tag{8}$$

In order to consider one step deeper into the future, i.e., to compute V_{n+1}^* from V_n^* we can turn (7) into an update:

$$V_{n+1}^*(s) = \max_{a \in A} \left[R(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) V_n^*(s') \right], \tag{9}$$

known

which is known as a Bellman backup H, allowing us to write (9) as

$$V_{n+1}^* = HV_n^*. \tag{10}$$

arbitrary initial value, e.g. optimal myopic value

Note that policy improvement and evaluation in (9) are combined (simultaneous)

60 **Value iteration**

- *Value iteration* using full backups

Initialize V arbitrarily, e.g.,$V(s) = 0$, for all $s \in \mathcal{S}^+$

Repeat
  $\Delta \leftarrow 0$
  For each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \max_a \sum_{s'} \mathcal{P}_{ss'}^a \left[\mathcal{R}_{ss'}^a + \gamma V(s')\right]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$ (a small positive number)

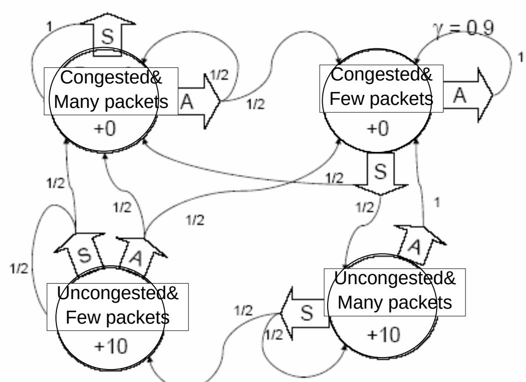
Output a deterministic policy, π, such that
  $\pi(s) = \arg\max_a \sum_{s'} \mathcal{P}_{ss'}^a \left[\mathcal{R}_{ss'}^a + \gamma V(s')\right]$

61 Value iteration discussion

- As $n \to \infty$, value iteration converges.
- Value iteration has converged when the largest update Δ in an iteration is below a certain threshold θ
- Exhaustive sweeps are not required for convergence: arbitrary states can be backed up in arbitrary order, provided that in the limit all states are visited infinitely often (Bertsekas and Tsitsiklis, 1989).
- This can be exploited by backing up the most promising states first, known as prioritized sweeping (Moore and Atkeson, 1993; Peng and Williams, 1993).

62 Value iteration - Example

$$V_{k+1}^{*}(s) = \max_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s' \mid s,a) V_k^{*}(s') \right]$$



k	V^k(PU)	V^k(PF)	V^k(RU)	V^k(RF)
1	0	0	10	10
2	0	4.5	14.5	19
3	2.025	8.55	16.525	25.075
4	4.75875	12.195	18.3475	28.72
5	7.62919	15.0654	20.3978	31.1804
6	10.2126	17.4643	22.6122	33.2102
7	12.4546	19.5402	24.7711	35.1201
8	14.3977	21.4086	26.7516	36.951
9	16.1128	23.1069	28.5172	38.6662
10	17.6489	24.6506	30.0835	40.2325
11	19.0347	26.0466	31.4796	41.6422

$$\gamma = 0.9$$

For V^2(RU):

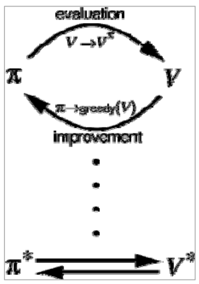
- For action S: [(.5)(10) + (.5)(0)] = 5
- For action A: [(.5)(0) + (.5)(0)] = 0

•So, V^2(RU) = 10 + (.9)(5) = 14.5

63

Policy iteration

- Continue improving the policy π and recalculating $V(\pi)$
- A finite MDP has a finite number of policies, so convergence is guaranteed in a finite number of iterations



1. Initialization
   $V(s) \in \Re$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Repeat
   $\Delta \leftarrow 0$
   For each $s \in \mathcal{S}$:
   $v \leftarrow V(s)$
   $V(s) \leftarrow \sum_{s'} \mathcal{P}_{ss'}^{\pi(s)} \left[\mathcal{R}_{ss'}^{\pi(s)} + \gamma V(s') \right]$
   $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   $b \leftarrow \pi(s)$
   $\pi(s) \leftarrow \arg\max_a \sum_{s'} \mathcal{P}_{ss'}^{a} \left[\mathcal{R}_{ss'}^{a} + \gamma V(s') \right]$
   If $b \neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop; else go to 2

64

Policy iteration discussion

- Converges in finite number of steps (only finite number of stationary policies), when using exact policy evaluation.
- Policy evaluation can be done iteratively (as shown before) or by solving the system of linear equations.
- When state space is large, this can be expensive.

65 **Relationship between policy iteration and value iteration**

Recall value iteration...

Initialize V arbitrarily, e.g., $V(s) = 0$, for all $s \in \mathcal{S}^+$

Repeat
  $\Delta \leftarrow 0$
  For each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \max_a \sum_{s'} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma V(s')\right]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$ (a small positive number)

Output a deterministic policy, π, such that
  $\pi(s) = \arg\max_a \sum_{s'} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma V(s')\right]$

Truncates policy iteration by combining one sweep of policy evaluation and one of policy improvement in each of its sweeps.

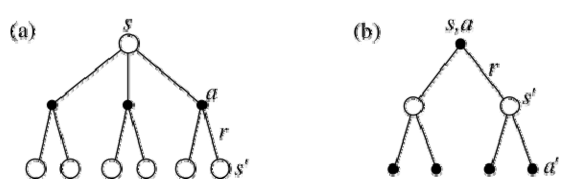
66 **Action-value function – another value function**

- Q^π is the expected return (value) starting from state *s*, taking action *a*, and thereafter following policy π

Action-value function for policy π

$$Q^\pi(s,a) = E_\pi\left[\sum_{t=0}^{\infty} \gamma^t R_t \mid s_0 = s, a_0 = a\right]$$
$$= R(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s' \mid s,a) V^\pi(s')$$

$$V^*(\boldsymbol{s}) = \max_{\boldsymbol{a} \in \boldsymbol{\mathcal{A}}} Q^*(\boldsymbol{s},\boldsymbol{a}) \qquad \pi^*(\boldsymbol{s}) = \arg\max_{\boldsymbol{a} \in \boldsymbol{\mathcal{A}}} Q^*(\boldsymbol{s},\boldsymbol{a})$$



Backup diagrams for (a) V^π and (b) Q^π

67 **Complete knowledge**

- In problems of *complete knowledge*, the agent has a complete and accurate model of the environment's dynamics.
- If the environment is an MDP, then such a model consists of the one-step *transition probabilities* and *expected rewards* for all states and their allowable actions.
- In problems of *incomplete knowledge*, a complete and perfect model of the environment is not available.

68 **What happens if the environment is unknown?**

- Model-based
  - Learn reward and transition probabilities
  - Then compute optimal value function and corresponding policy
  - Example: RTDP
- Model-free
  - Learn value function or action value function directly

69 **Asynchronous Dynamic Programming**

- Conventional dynamic programming methods (e.g. value iteration, policy iteration, etc.) makes exhaustive sweep over the state space, i.e. every state is backed up within each round of the iteration (Synchronous back up).
- Synchronous back up is not a necessary condition for convergence, asynchronous back up can achieve the same convergence result provided that in the limit all the states are visited infinitely often. With asynchronous back up, only a *subset* of the states are backed up at each stage *k*.
  (Bertsekas and Tsitsiklis, 1989)

$$V^{k+1}(s) = \begin{cases} \max_a \sum_{s'} \mathcal{P}_{ss'}^a \left[\mathcal{R}_{ss'}^a + \gamma V^k(s') \right], & if \ \ s \in S_k \\ V^k(s), & otherwise \end{cases}$$

70 # Real-time dynamic programming

- Real-time dynamic programming is a special case of asynchronous dynamic programming, in which the stages of the optimal value function being approximated and the stages of the decision problem being solved are related. At each stage, only one state is backed up, i.e. S_k is a singleton for all *k.*
  (Barto et al., 1995)
- The DP and control processes interact as follows:
  - Control decisions are based on the most up-to-date information from DP computation;
  - The state sequences generated during control influence the selection of states to which the DP back up operation is applied and whose approximated value functions have to be stored.
- No knowledge is required a priori in Real-time DP, on-line estimation technique is usually employed to estimate environment dynamics.
- Semi-uniform strategies (e.g. epsilon-greedy strategy) are used to control the state visit path in the control process, making sure every state is visited infinitely often.

71

A comparison

	Knowledge requirement	Computation	State back up
RTDP	No knowledge	On-line	Asynchronous
VI	Complete knowledge	Off-line	Synchronous

	Policy	Convergence	Complexity
RTDP	Dynamic	Yes	$O(\|S\|\|A\|)$
VI	Stationary	Yes	$O(\|A\|)$

72

Why do we need model-free on-line learning?

- If the transition probabilities are known, finding the optimal policy becomes a straightforward computational problem, however…
- If the transition probabilities and/or rewards are unknown, then this is a problem for *reinforcement learning* (RL)

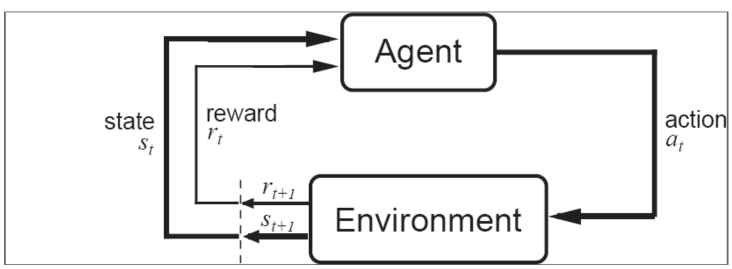
73 **RL – the big picture**

Types of Machine Learning

- Supervised learning: learn from labeled examples
  - The task of the supervised learner is to predict the value of the function for any valid input object after having seen a number of training examples (i.e. pairs of input and target output). To achieve this, the learner has to generalize from the presented data to unseen situations in a "reasonable" way
- Unsupervised learning: cluster unlabeled examples
  - Unsupervised learning encompasses many techniques which seek to summarize and explain key features of the data.
  - It is distinguished from supervised learning and reinforcement learning in that the learner is only given unlabeled examples.
  - One form of unsupervised learning is data clustering (e.g. K-means).
- Reinforcement learning: learn from interaction

74 **Agent-environment interaction**

By "state" we mean whatever information is available to the agent

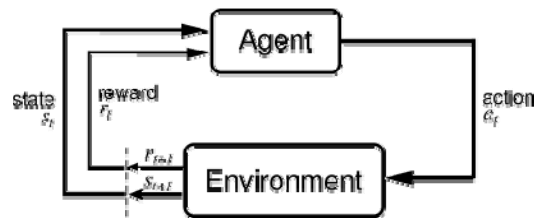


Everything inside the agent is completely known and controllable by the agent; everything outside is incompletely controllable but may or may not be completely known.

In our setting, who are the agent and the environment? What are the states, actions, rewards?

75 **Typical Agent**

- In reinforcement learning (RL), the agent observes a state and takes an action.
- Afterward, the agent receives a reward.



Goal: Optimize Reward

- Rewards are calculated in the environment
- Used to teach the agent how to reach a goal state
- Must signal what we ultimately want achieved, not necessarily subgoals
- May be *discounted* over time
- In general, seek to maximize the *expected return*

Examples of reinforcement learning techniques: Q-learning, Sarsa, actor critic etc.

76 **Q-learning**

- Reinforcement-learning techniques learn from experience, no knowledge of the model is required.
- Policy is often represented as state-action value function:

$$Q : S \times A \rightarrow \mathbb{R} \qquad (11)$$

and the policy as

$$\pi(s) = \arg\max_{a \in A} Q(s, a) \qquad (12)$$

- Q-learning update (Watkins, 1989): experience tuple (s, a, r, s')

$$Q(s, a) = (1 - \beta)\, Q(s, a) + \beta \Big[R(s, a) + \gamma \max_{a' \in A} Q(s', a') \Big], \qquad (13)$$

77 **Q-learning**

Initialize $Q(s, a)$ arbitrarily
repeat
  Initialize s
  repeat
    Choose a from s using policy derived from Q (e.g., ϵ-greedy)
    Take action a, observe r, s'
    $Q(s, a) = (1 - \beta)\, Q(s, a) + \beta \Big[r + \gamma \max_{a' \in A} Q(s', a') \Big]$
    $s \leftarrow s'$
  until s is terminal
until

78 **Q-learning**

Q-learning discussion:

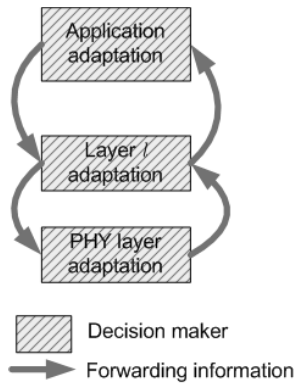
- Q-learning is guaranteed to converge to the optimal Q-values if all $Q(s, a)$ values are updated infinitely often Watkins and Dayan (1992).
- In order to make sure all actions will eventually be tried in all states exploration is necessary.
- A common exploration method is to execute a random action with small probability ϵ, which is known as ϵ-greedy exploration Sutton and Barto (1998).

Discussion: when can RL be used in cross-layer optimization problems?

78

79

Systematic layered optimization with information exchange



Key questions to be answered:
1. How can each layer optimally configure its own parameters?
2. What information should be exchanged between layers and how?

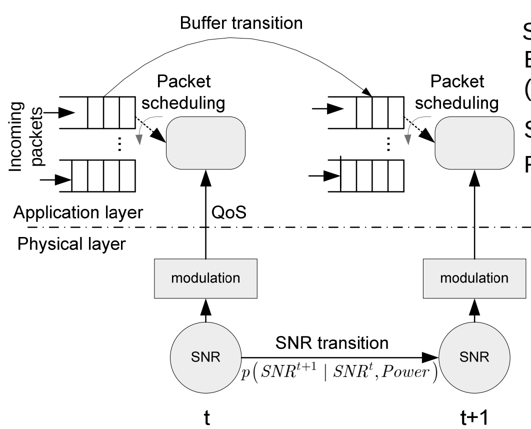
1

A systematic framework for cross-layer optimization

- Layered solution for cross-layer optimization
  - Explore the structure of cross-layer optimization
  - Decompose into multiple single-layer optimizations
  - Design optimal message exchange mechanisms across layers
- Dynamic cross-layer optimization with dynamic modeling
  - Dynamically model traffic and channel states
  - Develop structural results on optimal solutions
  - Decompose multi-user cross-layer optimization into multiple single-user optimizations
  - Design optimal message exchange mechanisms across users

2

Cross-layer optimization – formalized as MDP



State: Buffer fullness
External action:
(packet scheduling, source coding)
State transition: buffer transition
Reward: distortion reduction

State: SNR
Internal action: modulation
External action: Power
State transition: SNR transition
Cost: consumed power

Coupling between two layers:
internal actions at lower layers cooperatively determine the reward.

3

Centralized Markov Decision Process (MDP) formulation

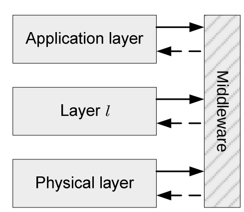
- Tuple $(S, \mathcal{X}, p, R)$
  - state space
    $s = (buffer\ length,\ SNR) \in S$
  - action space
    $\xi = (power,\ modulation,\ packet\ scheduling,\ source\ coding) \in \mathcal{X}$
  - transition probability
    $p(s|s,\xi) = p(s^{k+1} = s|s^k = s, \xi^k = \xi)$
  - $R(s,\xi)$ immediate reward at state s (e.g. application quality) when performing action ξ

- Goal is to maximize some cumulative function of the rewards

$$\max_{\xi^t \in \mathcal{X}} \left\{ \sum_{t=1}^{\infty} \alpha^t R\left(s^t, \xi^t\right) \right\}$$

4

Centralized DP operator



- Value iteration algorithm

$$V_n(s) = \max_{\xi \in X} \left\{ R(s,\xi) + \alpha \sum_{s' \in S} p\left(s' \mid s,\xi\right) V_{n-1}\left(s'\right) \right\}$$

- Policy iteration algorithm
  - Policy evaluation

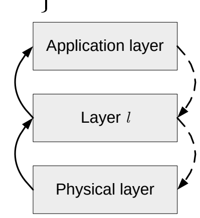
$$V^{\pi_n} = R(s, \pi_n(s)) + \alpha \sum_{s' \in S} p\left(s' \mid s, \pi_n(s)\right) V^{\pi_n}\left(s'\right)$$

  - Policy improvement

$$\pi_{n+1}(s) = \arg\max_{\xi \in \mathcal{X}} \left\{ R(s,\xi) + \alpha \sum_{s' \in S} p\left(s' \mid s,\xi\right) V^{\pi_n}\left(s'\right) \right\}$$

- Key step: Dynamic programming (DP) operator

$$\max_{\xi \in \mathcal{X}} \left\{ R(s,\xi) + \alpha \sum_{s' \in S} p\left(s' \mid s,\xi\right) V\left(s'\right) \right\}$$



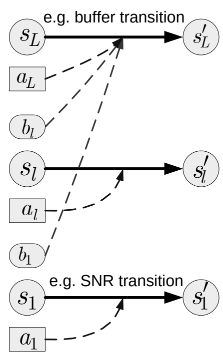
5 **Structure of cross-layer optimization**

- Transition probability

$$p(s' \mid s, \xi) = \prod_{l=1}^{L-1} p(s_l' \mid s_l, a_l) p(s_L' \mid s, a_L, \boldsymbol{b}_{1,\dots,L-1})$$

- Utility function

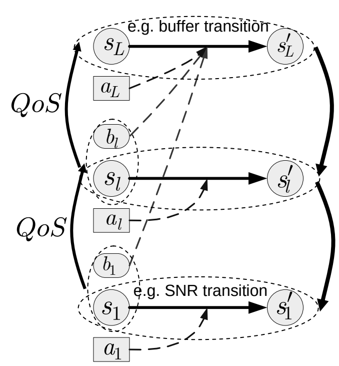
$$R(s,\xi) = \underbrace{g_L(s, a_L, \boldsymbol{b}_{1,\cdots,L-1})}_{\text{Application utility}} - \lambda^b \underbrace{d(s, a_L, \boldsymbol{b}_{1,\cdots,L-1})}_{\text{internal cost}} - \sum_{l=1}^{L-1} \lambda_l^a \underbrace{c_l(s_l, a_l)}_{\text{external cost}}$$



6 **Key ideas of layered DP operator**

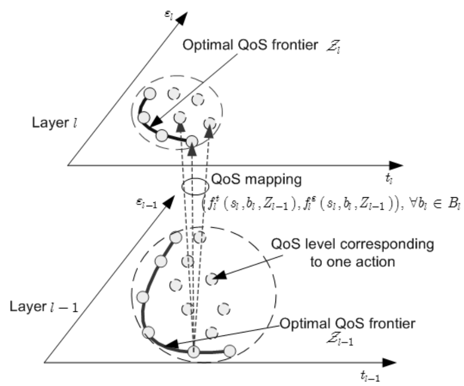
- Define instantaneous QoS exchange to determine internal actions – upperward message exchange

- Design layered DP operator to determine external actions – downward message exchange



7 Instantaneous QoS

- Instaneous QoS definition: $Z_l = (\varepsilon_l, \tau_l, v_l)$
  - packet loss probability ε_l
  - transmission time per packet τ_l
  - transmission cost per packet v_l



8 Proposed layered DP operator

Centralized DP operator (existing):

$$V(s) = \max_{\xi}\left\{R(s,\xi) + \alpha\sum_{s'\in S} p(s' \mid s,\xi)V(s')\right\}$$

$$V(s_1,\cdots,s_L) =$$

$$\max_{\substack{a_1,\cdots,a_L\\ b_1,\cdots,b_{L-1}}}\left\{\underbrace{g_L(s,a_L,\boldsymbol{b}_{1,\cdots,L-1}) - \lambda^b d(s,a_L,\boldsymbol{b}_{1,\cdots,L-1})}_{\substack{\text{Computed based on } QoS\\ R_{in}(s_L,a_L,Z_{L-1})}} - \sum_{l=1}^{L-1}\lambda_l^a c_l(s_l,a_l) + \right.$$

$$\left.\alpha\sum_{s'\in S}\prod_{l=1}^{L-1} p(s_l' \mid s_l,a_l)p(s_L' \mid s,a_L,\boldsymbol{b}_{1,\ldots,L-1})V(s_1',\cdots,s_L')\right\}$$

9 Proposed layered DP operator (Cont'd)

Layered DP operator (proposed):

Layer L

$$V_{L-1}\left(s_1',\cdots,s_{L-1}',s_1,\cdots,s_L\right) = \max_{a_L,Z_{L-1}} \left[R_{in}\left(s_L,a_L,Z_{L\text{-}1}\right) - \lambda_L c_L\left(s_L,a_L\right) + \gamma \sum_{s_L\in\mathcal{S}_L} p\left(s_L' \mid s_L,a_L,Z_{L\text{-}1}\right) V_L\left(s_1',\cdots,s_L'\right)\right]$$

Downward message

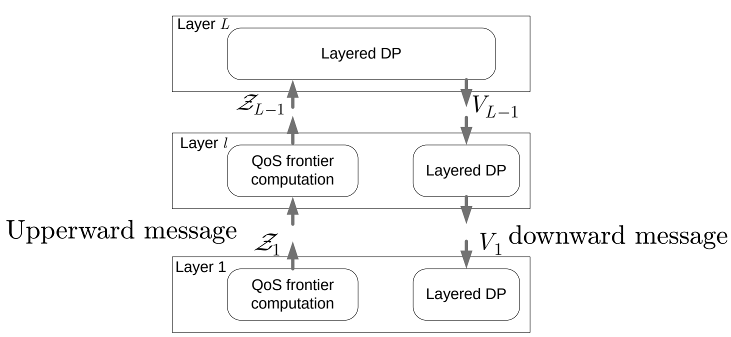
Layer l

$$V_{l-1}\left(s_1,\cdots,s_L,s_1',\cdots,s_{l-1}'\right) = \max_{a_l\in A_l}\left[-\lambda_l c_l\left(s_l,a_l\right) + \sum_{s_l'\in\mathcal{S}_l} p\left(s_l' \mid s_l,a_l\right) V_l\left(s_1,\cdots,s_L,s_1',\cdots,s_l'\right)\right]$$

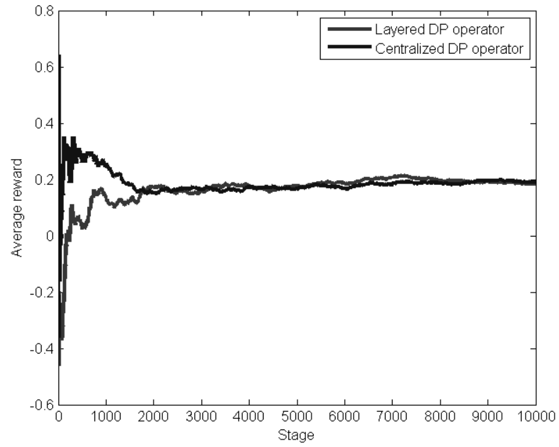
Layer 1

$$V\left(s_1,\cdots,s_L\right) = \max_{a_1\in A_1}\left[-\lambda_1 c_1\left(s_1,a_1\right) + \sum_{s_1'\in\mathcal{S}_1} p\left(s_1' \mid s_1,a_1\right) V_1\left(s_1,\cdots,s_L,s_1'\right)\right]$$

10 Message exchange & layer optimizer



11 Comparison of centralized and layered DP operator



Application quality: 32.5dB for the layered one
32.8dB for the centralized one

12 A systematic framework for cross-layer optimization

Layered solution for cross-layer optimization

- Explore the structure of cross-layer optimization
- Decompose into multiple single-layer optimizations
- Design optimal message exchange mechanisms across layers

Dynamic cross-layer optimization with dynamic modeling

- Dynamically model traffic and channel states
- Develop structural results on optimal solutions
- Decompose multi-user cross-layer optimization into multiple single-user optimizations
- Design optimal message exchange mechanisms across users

13

Past work on single-user cross-layer optimization for multimedia

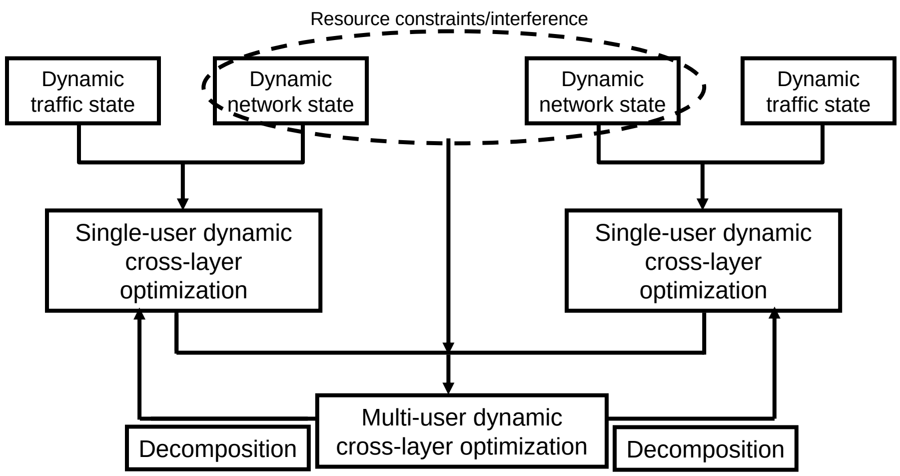
- Cross-layer optimization below APP
  - Optimizing the given QoS metrics under the resource constraints, e.g. power-delay trade-off [R. Berry, 2002].
- Cross-layer optimization including APP
  - Unequal Error Protection (UEP) [M. Luby, 1996]
  - Rate-distortion optimized packet scheduling (RaDiO) [P. Chou, 2001]
  - Online (myopic) cross-layer adaptation based on observed channel conditions [M. van der Schaar, 2005]

14

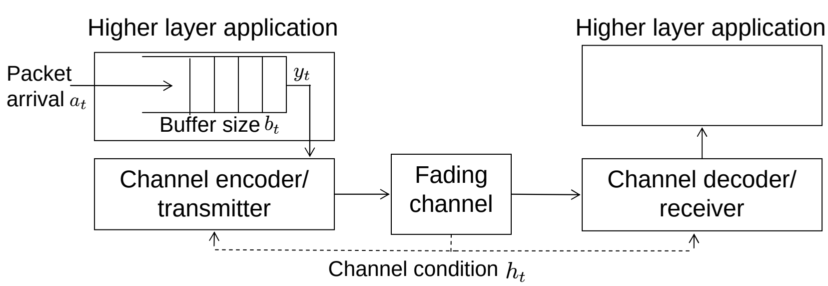
Past work on multi-user cross-layer optimization for multimedia

- Application-centric utility maximization [M. van der Schaar 2006]
  - Using static utility function to model media quality
  - Allocating resources before adapting cross-layer transmission strategies
- Network utility maximization (NUM) [M. Chiang 2007, A. Katsaggelos 2008]
  - Using static utility function to model media quality
  - Allocating resources before adapting cross-layer transmission strategies
- Wireless NUM [D. O'Neill, A. GoldSmith, S. Boyd, 2008]
  - Time-varying wireless channels
  - Considering average resource constraints but not the media characteristics.

15 **A systematic framework for cross-layer optimization**



16 **Illustrative example:**
Communication with average delay constraints



Assumptions:

•Channel condition is modeled as a finite state Markov chain
•i.i.d. packet arrival
•Packets are equally important
•Packets do not have hard delay constraints

Model: Markov decision process

State: (b_t, h_t) Action: y_t Utility: $u_t = -\underbrace{(b_t - y_t)}_{\text{Buffer size (delay)}} - \lambda \underbrace{\rho(h_t, y_t)}_{\text{Power consumption}}$

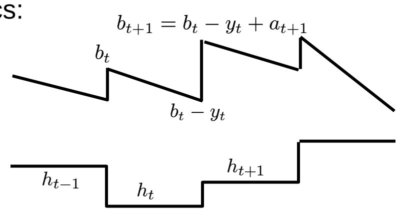
17

Illustrative example (Cont'd): Communication with average delay constraints

System dynamics:

Buffer dynamics

Channel dynamics



(depend on packet arrival and packet transmission)

Objective:

$$\max_{y_t \geq 0} \sum_{t=0}^{\infty} \alpha^t u_t \quad (0 \leq \alpha < 1)$$

discounted long-term utility

Bellman's equation:

state transition probability state-value function

$$V(b,h) = \max_{b \geq y \geq 0} \left\{ -(b-y) - \lambda \rho(h,y) + \alpha \boxed{p(h' \mid h)\, p(a')} V(b - y + a', h') \right\}$$

18

Illustrative example (Cont'd): Communication with average delay constraints

Structural properties [R. Berry, 2001]:

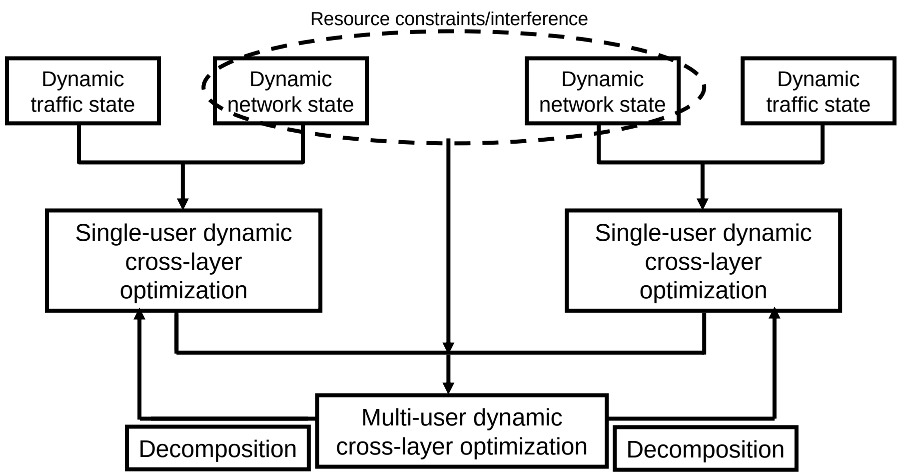
State value function is concave over buffer length

Delay-power trade-off is a convex function

Optimal packet scheduling (i.e. the amount of data transmitted at each time slot) is non-decreasing in the buffer length.

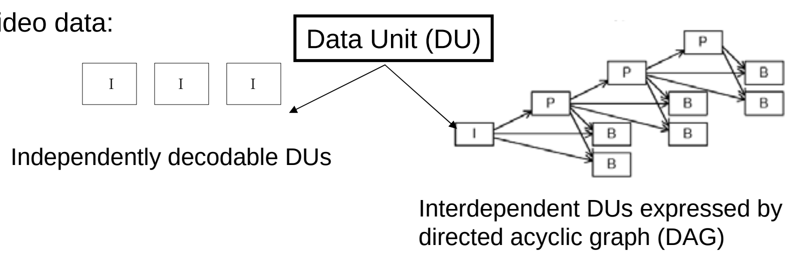
This example does not consider the traffic dynamics!

19 A systematic framework for cross-layer optimization



20 Media data representation

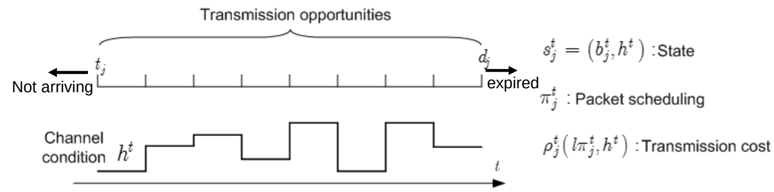
Video data:



- Each DU has the following attributes:
  - Arrival time: time at which the DU is ready for transmission: t_i
  - Delay deadline: d_i
  - Distortion impact: q_i per packet
  - Size in packets (with length l): b_i
  - Interdependency between DUs: *expressed by Directed Acyclic Graph (DAG) called dependency graph (DG)*

21 Single-packet transmission

- Single-packet transmission over wireless fading channel (modeled as finite state Markov chain)



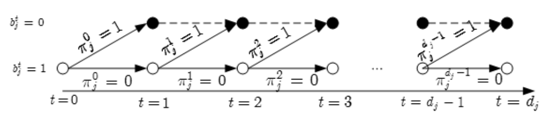
- Objective:

$$\max_{\pi_j^t} \sum_{t=t_j}^{d_j} \alpha^t \left(\underbrace{\boxed{q_j \pi_j^t - \lambda \rho_j^t \left(l\pi_j^t, h^t \right)}}_{\text{Reward at time } t} \right)$$

$$s.t.\ \pi_j^t \in \{0,1\}, \sum_{t=t_j}^{d_j} \pi_j^t \le 1 \quad \text{Scheduling constraint}$$

22 Optimal stopping problem formulation for single-packet transmission

- Optimal stopping problem formulation:



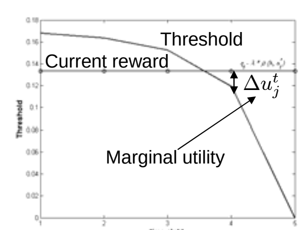
- Optimal solution

Dynamic programming:

$$U_j^{d_j+1}\left(s_j^{d_j+1}\right) = 0, \forall s_j^{d_j+1}$$

$$U_j^t\left(s_j^t\right) = \max_{\pi_j^t, a_j^t}$$

$$\left\{ q_j - \lambda \rho^t \left(l\pi_j^t, h^t \right) + \alpha \sum_{s_j^t} p\left(s_j^{t+1} \mid s_j^t, \pi_j^t\right) U_j^{t+1}\left(s_j^{t+1}\right) \right\}, \forall s_j^t, t$$



Threshold: $\overline{u}_j^t\left(h^t\right) = \alpha \sum_{h^{t+1}} p\left(h^{t+1} \mid h^t\right) U_j^{t+1}\left(\left(1, h^{t+1}\right)\right)$

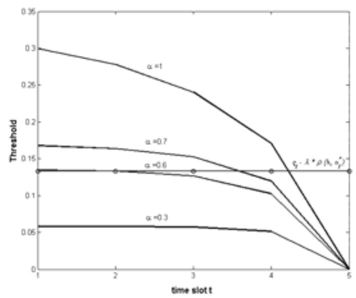
Marginal utility: $\Delta u_j^t = q_j - \lambda \rho^t \left(l\pi_j^t, h^t \right) - \overline{u}_j^t\left(h^t\right)$

$$\pi_j^{t,*}\left(\left(1, h^t\right)\right) = \begin{cases} 1 & if\ \Delta u_j^t > 0 \\ 0 & o.w. \end{cases}$$

23

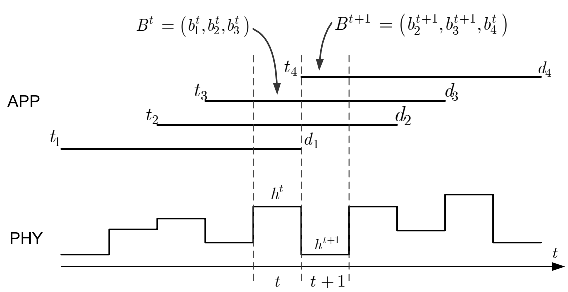
Insights about optimal single-packet transmission solutions

- Remarks on threshold
  - Threshold represents the average future net utility evaluating the future transmission opportunities.
  - It is decreased when approaching the delay deadline.
  - It answers the question: when is the packet transmitted?
- Discounted impact



24

Multi-packet transmission



State: $s^t = (B^t, h^t)$

Packet scheduling: $\pi^t = \{\pi_j^t\}_{j:t_j \le t \le d_j}$

Transmission cost: $\rho^t \left(l \sum_{j:t_j \le t \le d_j} \pi_j^t, h^t \right)$

25

Markov decision process formulation for independently decodable packets

Objective:

$$\max_{\pi^t, a^t} \sum_{t=0}^{d^{\max}} \alpha^t \left(\sum_{j:t_j \le t \le d_j} q_j \pi_j^t - \lambda \boxed{\rho^t \left(l \sum_{j,t_j \le t \le d_j} \pi_j^t, h^t \right)} \right)$$

convex transmission cost (cost of self-congestion)

$$s.t. \;\; \pi_j^t \in \{0,1\}, \sum_{t=t_j}^{d_j} \pi_j^t \le 1$$

⇦ Packet scheduling constraint

Solution: dynamic programming

$$U^{d_{\max}+1}\left(s^{d_{\max}+1}\right) = 0, \forall s^{d_j+1}$$

$$U^t\left(s^t\right) = \max_{\pi^t} \left\{ u^t\left(s^t, \left(\pi^t\right)\right) + \alpha \sum_{s^t} p\left(s^{t+1} \mid s^t, \pi^t\right) U^{t+1}\left(s^{t+1}\right) \right\}, \forall s^t, t$$

- Complexity is increased exponentially with the number of packets.
- Structure of this problem is not explored.

26

Transmission priority

Marginal utility:

$$\Delta u_j^t \boxed{(s^t, \pi_{-j}^t)} = q_j - \lambda \rho_j^t(s^t, \pi_{-j}^t) - \bar{u}_j^t(s^t, \pi_{-j}^t)$$

⇧

It depends on the current state & actions of other packets.

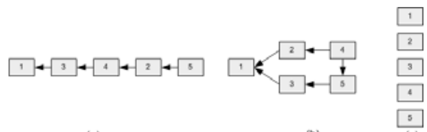
Definition (Transmission Priority): Packet j has a higher transmission priority than packet k (denoted by $j \lhd k$) at time slot t if $\Delta u_j^t(s^t, \pi_{-j}^t) \ge \Delta u_k^t(s^t, \pi_{-k}^t)$.

> Lemma : For independently decodable packets j and k, if $q_j \ge q_k$ and $d_j \ge d_k$, then $j \lhd k$ under any state.

27 Priority graph & traffic state

- Transmission priorities between packets can be expressed as a DAG called as *priority graph* (PG).
- At time slot t:
  - Packet to be transmitted: $J^t = \{j : b_j^t = 1, t_j \leq t \leq d_j\}$
  - PG expression: $PG =< V, E >$ with $V = J^t, E = \{(j,k) | j \lhd k\}$

e.g.

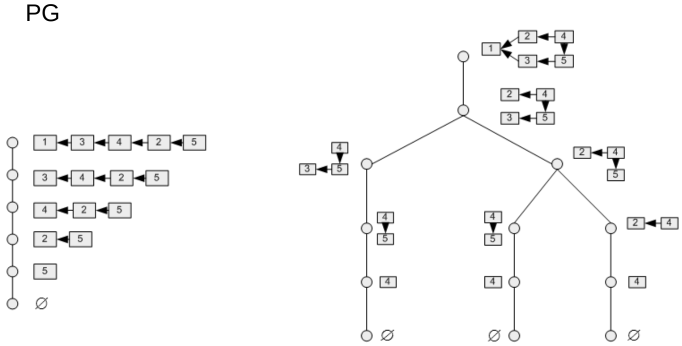


(a) (b) (c)

- Each traffic state can be uniquely represented by a PG

28 Constructing a state tree

- A state tree can be induced from the PG
  - Each node represents one PG
  - Each PG is obtained by removing one of the root packets in its parent PG

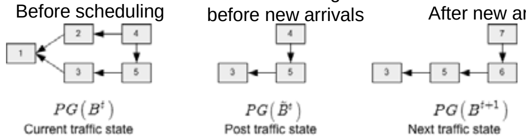


(a) (b)

29

Traffic state transition and post-decision state value function

Traffic state transition:



Remark: Post-decision traffic state and next traffic state is one-to-one mapping.

Post-state value function computation:

$$\overline{u}^t\left(\left(\tilde{B}^t, h^t\right)\right) = \alpha \sum_{h^{t+1}} p\left(h^{t+1} \mid h^t\right) U^{t+1}\left(s^{t+1}\right)$$

Dynamic programming:

$$U^t\left(s^t\right) = \max_{\pi^t} \left\{ \underbrace{\boxed{\sum_{j \in J^t} q_j b_j^t \pi_j^t - \lambda \rho^t \left(\sum_{j \in J^t} l_j b_j^t \pi_j^t, h^t \right)}}_{\text{Current utility}} + \underbrace{\boxed{u^t\left(\tilde{B}^t\left(\pi^t\right), h^t\right)}}_{\text{Future utility}} \right\}$$

30

Multi-packet transmission as travelling the state tree

Theorem (cross-layer optimization) At each state, the optimal packet scheduling is determined as follows.

$k = 1$

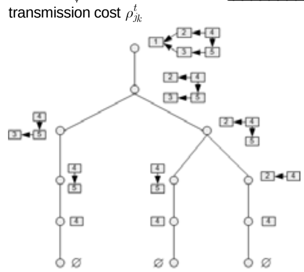
Repeat

Phase 1: $j_k = \arg\max_{j \in root(PG^k)} \{\Delta \bar{u}_j^t(PG^k, h^t)\}$

Phase 2: If $\Delta \bar{u}_{j_k}^t > 0$, then $\pi_{j_k}^t = 1; PG^{k+1} = PG^k / \{j_k\}; k \leftarrow k+1;$

Else *stop*.

$$\Delta \overline{u}_{j_k}^t\left(PG^k, h^t\right) = q_{j_k} - \lambda \underbrace{\left[\rho\left(kl, h^t\right) - \rho\left((k-1)l, h^t\right)\right]}_{\text{transmission cost } \rho_{j_k}^t} - \underbrace{\left[\overline{u}^t\left(\left(PG^k, h^t\right)\right) - \overline{u}^t\left(\left(PG^k / \{j_k\}, h^t\right)\right)\right]}_{\text{marginal utility } \overline{u}_j^t}$$



31

Properties of optimal solutions

- Property I

  If $j \lhd k$, then packet j is transmitted earlier than packet k.

- Property II

  If the optimal packet scheduling policy is $\{j_1, \cdots, j_N\}$, then
  $\Delta u_{j_1}^t \geq \Delta u_{j_2}^t \geq \cdots \geq \Delta u_{j_N}^t$

- Property III

  The state value function evaluated at state s^t is the summation of the marginal utility of all transmitted packets, i.e.

  $$U^t(s^t) = \sum_{j=1}^{N} \Delta u_j^t$$

- Property IV

  $U^t(s^t)$ is a convex function of transmission cost.

32

Multi-packet transmission with linear transmission cost

- For independently decodable packets, the linear transmission cost leads to separable utility function.

$$u^t(s^t, \pi^t) = \sum_{j:t_j \leq t \leq d_j} q_j \pi_j^t - \lambda \sum_{j:t_j \leq t \leq d_j} \pi_j^t \rho_j^t(l, h^t) = \sum_{j:t_j \leq t \leq d_j} \left(q_j - \rho_j^t(l, h^t)\right) \pi_j^t$$

Corollary : The cross-layer optimization for multiple independently decodable packets with linear transmission cost can be *decomposed* into multiple single-packet cross-layer optimization problems.

Linear complexity w.r.t. the number of packets

How about transmission with nonlinear costs?

33

Complexity of multi-packet transmission with nonlinear cost

- How many states can be visited at each time slot starting from the initial state $B^1 = (b_j^1 = 1 | \forall j : t_j = 1)$? *(computation complexity)*
- How many post-state value functions should be stored? *(memory overhead)*

Lemma : If $t_j \leq t_k$ and $j \lhd k$, then the traffic state must not have $b_j^t = 1, b_k^t = 0$.

- In other words, packet j cannot be transmitted after packet k during the time before time slot t.
  - Hence, the possible states visited at time slot t can be represented by a priority graph (called history graph (HG)).
  - Constructing HG: $HG^t = (V_{HG}, E_{HG})$

$$V_{HG} = \{j : t_j < t \leq d_j\} \qquad E_{HG} = \{(j,k) | t_j \leq t_k, j \lhd k\}$$

34

Complexity (cont'd)

Definition (Disconnection degree):
The disconnection degree $\phi(DAG)$ of a DAG is the number of packet pairs for each of which there exists no path to connect these two packets.

Note: the disconnection degree is determined by the characteristics of the media packets. The priority chain has *zero* disconnection degree.

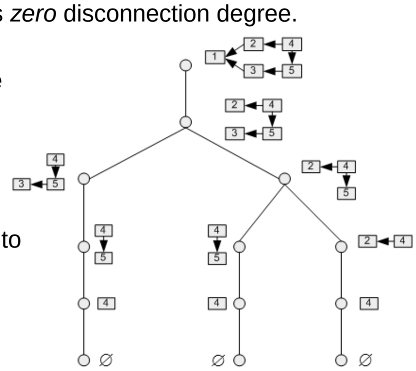
Number of states to be visited at time slot *t* is

$$\#V_{HG^t} + \phi(HG^t)$$
$$= \#\{\text{distinct nodes in the state tree}\}$$

e.g. 5 + 2 = 7.

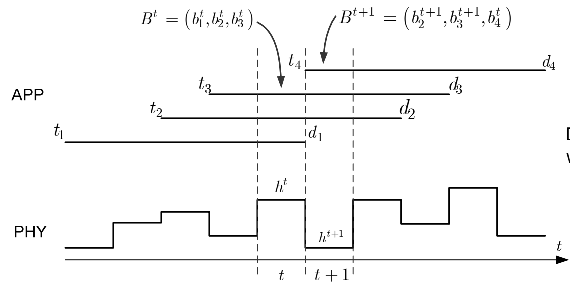
Number of post-state value functions to be stored at time slot t is

$$\#V_{HG^{t+1}} + \phi(HG^{t+1})$$
$$= \#\{\text{distinct nodes in the state tree}\}$$



35

Cross-layer optimization for multiple interdependent packets

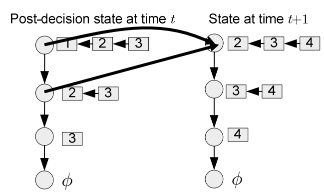


State:
$s^t = (D^t, B^t, h^t)$

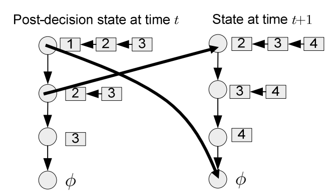
Dependency state representing whether the parant packets are transmitted or not.

Lemma : If $j \prec k$, then $j \lhd k$.

Post-decision state transition:



Independently decodable packets



Interdependent packets

36

Multi-packet transmission as travelling the state tree

Theorem (cross-layer optimization) At each state, the optimal packet scheduling is determined as follows.

$k = 1$

Repeat

Phase 1: $j_k = \arg\max_{j \in root(PG^k)} \{\Delta \bar{u}_j^t (PG^k, h^t)\}$

Phase 2: If $\Delta \bar{u}_{j_k}^t > 0$, then $\pi_{j_k}^t = 1; PG^{k+1} = PG^k / \{j_k\}; k \leftarrow k+1;$

Else *stop*.

$$\underbrace{\Delta \bar{u}_{j_k}^t \left(PG^k, h^t\right)}_{\text{marginal utility}} = q_{j_k} - \lambda \underbrace{\left[\rho\left(kl, h^t\right) - \rho\left((k-1)l, h^t\right)\right]}_{\text{transmission cost } \rho_{j_k}^t} - \underbrace{\left[\bar{u}^t\left(\left(PG^k, h^t\right)\right) - \bar{u}^t\left(\left(PG^k / \{j_k\}, h^t\right)\right)\right]}_{\text{threshold } \bar{u}_j^t}$$

37 **Complexity of cross-layer optimization**

Number of post-state value functions to be stored is

$$\leq 2^{|D^{t+1}|}\{\#V_{HG^{t+1}} + \phi(HG^{t+1})\}$$

Number of visited states is

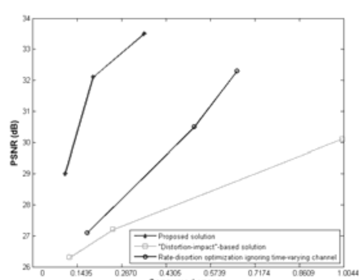
$$\leq 2^{|D^{t}|}\{\#V_{HG^{t}} + \phi(HG^{t})\}$$

Number of dependency states

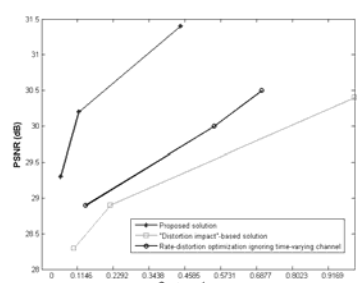
Remarks:
1. Interdependencies can significantly reduce the disconnection degree of the priority graph, and hence, reduce the complexity;
2. Long-term dependency may increase the dependency states which can increase the complexity.

38 **Simulation results**

- Comparison of various cross-layer optimization



Foreman



Coastguard

39 **Performance comparison**

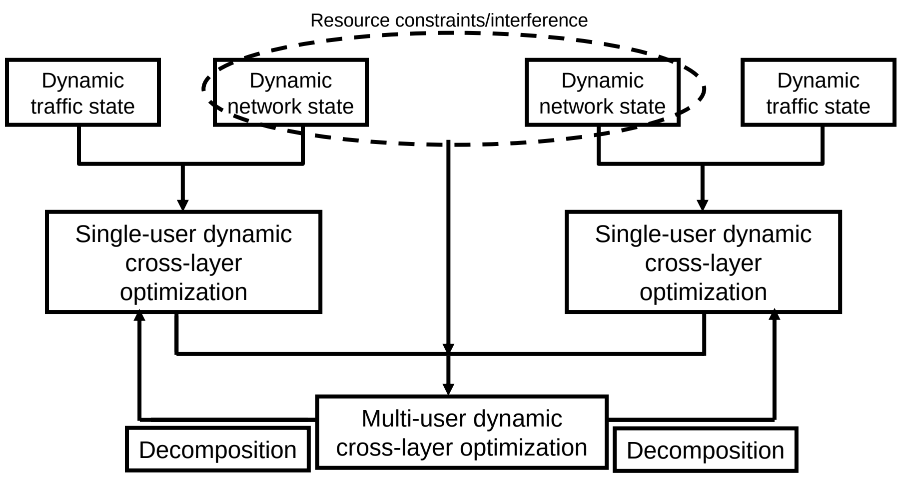
Table 1. Complexity comparison
(Number of post-states or states to be visited)

	Standard dynamic programming	Proposed solution
Independent Packets	329	26
Interdependent packets	4.4×10^{11}	304

Table 2. Comparison with RaDiO [P. Chou, 2005]

	Feedback of transmission outcome	Transmission cost	Adaptation	Optimization	Performance (time-varying channel)
RaDiO	Delayed	Linear	Single-layer (i.e. APP)	Iteration	Suboptimal
Proposed solution	Immediate feedback	Convex (e.g. self-congested)	Cross-layer	Dynamic programming	Optimal

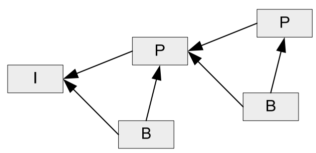
40 **A systematic framework for cross-layer optimization**



41

Dynamic traffic state (1)

Fixed GOP:



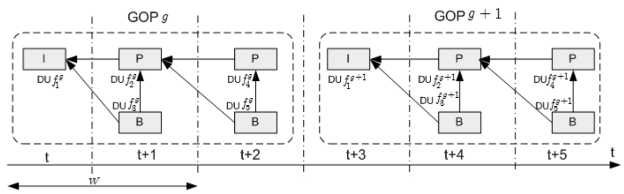
DU's attributes:

Size: l_j packets

Distortion impact: q_j per packet

Delay deadline: d_j

Dependency: DAG



$F_t = \{f_1^g, f_2^g, f_3^g\}$ $F_{t+1} = \{f_2^g, f_3^g, f_4^g, f_5^g\}$ $F_{t+2} = \{f_4^g, f_5^g, f_1^{g+1}\}$ $F_{t+3} = \{f_1^{g+1}, f_2^{g+1}, f_3^{g+1}\}$

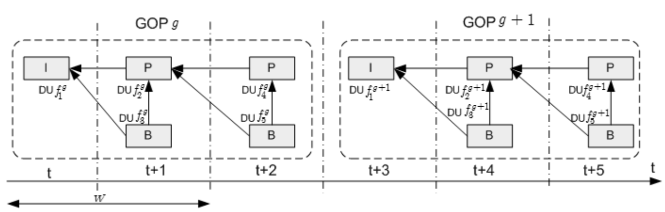
Traffic state: $\mathcal{T}_t = (F_t, B_t)$ $\quad F_t = \{f_j^g \mid d_j^g \in [t, t+W)\}$ $\quad B_t = \{b_j \mid j \in F_t\}$

DU type Buffer size

Note: Traffic state considered here is defined on the DU level (i.e. a coarse version of packet level state considered before).

42

Dynamic traffic state (2)



$F_t = \{f_1^g, f_2^g, f_3^g\}$ $F_{t+1} = \{f_2^g, f_3^g, f_4^g, f_5^g\}$ $F_{t+2} = \{f_4^g, f_5^g, f_1^{g+1}\}$ $F_{t+3} = \{f_1^{g+1}, f_2^{g+1}, f_3^{g+1}\}$

Traffic state transition:

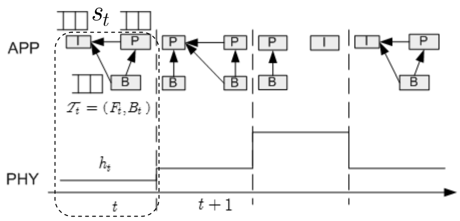
$F_t \to F_{t+1}$: Deterministic

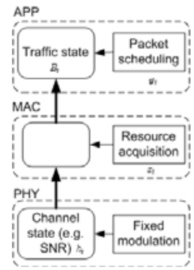
$$B_t \to B_{t+1}: \quad b_j \leftarrow \begin{cases} -1 & \textit{if undecodable} \\ b_j - y_j & \textit{if decodable \& previously arriving} \\ l_j & \textit{if decodable \& newly arriving} \end{cases}$$

Utility function: $u_t(\mathcal{T}_t, \boldsymbol{y}_t, r_t) = \sum_{j \in F_t} q_j \cdot y_j$

Traffic state transition is Markovian.

43 **Single-user cross-layer optimization**





Objective:

$$\max_{\substack{\boldsymbol{y}_t \in \mathcal{P}(s_t, x_t) \\ x_t \geq 0, t \geq 0}} E\left\{\sum_{t=0}^{\infty} \alpha^t \left(u_t\left(s_t, \boldsymbol{y}_t, x_t\right) - \lambda_{s_t} x_t\right)\right\}$$

State: s_t Action: $(x_t, \boldsymbol{y}_t)$

44 **Optimal solution property**

- Bellman's equation

$$U(s, \boldsymbol{\lambda}) = \max_{\substack{\boldsymbol{y} \in \mathcal{P}(s,x) \\ x \geq 0}} \left[u(s, \boldsymbol{y}, x) - \lambda_s x + \alpha p(s' \mid s, \boldsymbol{y}, x) U(s', \boldsymbol{\lambda})\right]$$

- Define the utility function of allocated resource:

$$H(s, \boldsymbol{\lambda}, x) = \max_{\boldsymbol{y} \in \mathcal{P}(s,x)} \left[u(s, \boldsymbol{y}, x) - \lambda_s x + \alpha p(s' \mid s, \boldsymbol{y}, x) U(s', \boldsymbol{\lambda})\right]$$

Lemma: $H(s, \boldsymbol{\lambda}, x)$ is a concave function of x.

Because the optimal packet scheduling policy always transmits the packets with highest marginal utility.

45

Multi-user dynamic cross-layer optimization formulation

Multi-user Markov decision process (MUMDP)

$$U^* = \max_{\boldsymbol{y}_t^i, x_t^i, i=1,\cdots,M} E\left[\sum_{t=0}^{\infty}\sum_{i=1}^{M} (\alpha)^t \, u_t^i\left(s_t^i, \boldsymbol{y}_t^i, x_t^i\right)\right]$$

$$s.t.\ \boxed{\boldsymbol{y}_t^i} \in \mathcal{P}^i\left(s_t^i, \boxed{x_t^i}\right),\ \boxed{\sum_{i=1}^{M} x_t^i \le 1}, \forall t \ge 0$$

Packet scheduling

Transmission time

Stage resource constraint

Convex optimization on resource allocation.

Network utility maximization (NUM)

$$\max \sum_{i=1}^{M} u^i\left(s^i, \boldsymbol{y}^i, x^i\right)$$

$$s.t.\ \boldsymbol{y}^i \in \mathcal{P}^i\left(s^i, x^i\right), \quad \sum_{i=1}^{M} x^i \le 1$$

46

Primary solution and its difficulties

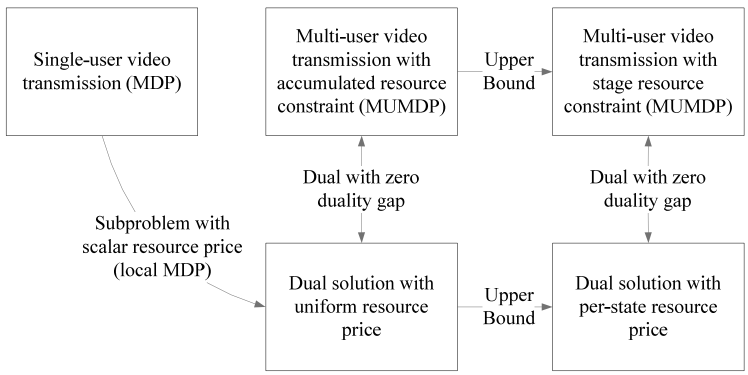
- Primary solution

$$U(\boldsymbol{s}) = \max_{\substack{\boldsymbol{y}^i \in \mathcal{P}^i\left(s^i, x^i\right), i=1,\cdots,M \\ \sum_{i=1}^{M} x^i \le 1}} \left[\sum_{i=1}^{M} u_i\left(s^i, \boldsymbol{y}^i, x^i\right) + \alpha \sum_{\boldsymbol{s}'} \prod_{k=1}^{M} p\left(s^{k'} \mid s^k, \boldsymbol{y}^k, x^k\right) U(\boldsymbol{s}')\right], \forall \boldsymbol{s}$$

  - Centralized solution is required: *value iteration, policy iteration, etc*
  - State value function is not decomposable, ☹.

Coupling is due to the stage-resource constraint.

Relationship of different solutions



Dual solutions

Introducing Lagrangian multiplier (resource price) at each multi-user system state (per-state resource price):

$$U(\boldsymbol{\lambda}) = \max_{\substack{\boldsymbol{y}_t^i, x_t^i \geq 0,\\ i=1,\cdots,M,\\ t\geq 0}} E\left[\sum_{t=0}^{\infty}(\alpha)^t \sum_{i=1}^{M}\left(u_t^i\left(s_t^i, \boldsymbol{y}_t^i, x_t^i\right) - \lambda_{\boldsymbol{s}_t} x_t^i + \frac{\lambda_{\boldsymbol{s}_t}}{M}\right) \mid \boldsymbol{s}_0\right]$$

Zero duality gap!

$$U^{\boldsymbol{\lambda},*} = \min_{\boldsymbol{\lambda}\geq 0} U(\boldsymbol{\lambda})$$

Non-decomposable! ☹

Introducing uniform resource price

$$U(\lambda) = \max_{\substack{\boldsymbol{y}_t^i, x_t^i \geq 0,\\ i=1,\cdots,M, t\geq 0}} E\left[\sum_{t=0}^{\infty}(\alpha)^t \sum_{i=1}^{M}\left(u_t^i\left(s_t^i, \boldsymbol{y}_t^i, x_t^i\right) - \lambda x_t^i + \frac{\lambda}{M}\right) \mid \boldsymbol{s}_0\right]$$

$$U^{\lambda,*} = \min_{\lambda\geq 0}\sum_{i=1}^{M} U^i\left(s_0^i, \lambda\right)$$

Decomposable, ☺

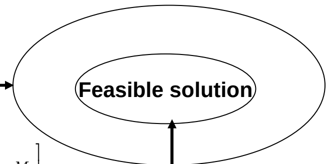
$$U^i\left(s^i, \lambda\right) = \max_{y^i, x^i}\left[u^i\left(s^i, y^i, x^i\right) - \lambda x^i + \frac{1}{M}\lambda + \alpha\sum_{s^{i\prime}} p\left(s^{i\prime} \mid s^i, \boldsymbol{y}^i, x^i\right) U^i\left(s^{i\prime}, \lambda\right)\right]$$

49 However, the duality gap is not zero…

- The dual solution with uniform resource price is dual to the following problem:

$$\hat{U}^* = \max_{\boldsymbol{y}_t^i, x_t^i, i=1,\cdots,M} E\left[\sum_{t=0}^{\infty}\sum_{i=1}^{M}(\alpha)^t\, u_t^i\left(s_t^i, \boldsymbol{y}_t^i, x_t^i\right) \mid s_0^1, \cdots, s_0^M\right]$$

$$s.t.\ \boldsymbol{y}_t^i \in \mathcal{P}^i\left(s_t^i, x_t^i\right),\ \sum_{t=0}^{\infty}\sum_{i=1}^{M}(\alpha)^t\left(x_t^i\right) - \frac{1}{1-\alpha} \le 0$$



- Original primary problem:

$$U^* = \max_{\boldsymbol{y}_t^i, x_t^i, i=1,\cdots,M} E\left[\sum_{t=0}^{\infty}\sum_{i=1}^{M}(\alpha)^t\, u_t^i\left(s_t^i, \boldsymbol{y}_t^i, x_t^i\right) \mid s_0^1, \cdots, s_0^M\right]$$

$$s.t.\ \boldsymbol{y}_t^i \in \mathcal{P}^i\left(s_t^i, x_t^i\right),\ \sum_{i=1}^{M} x_t^i \le 1, \forall t \ge 0$$

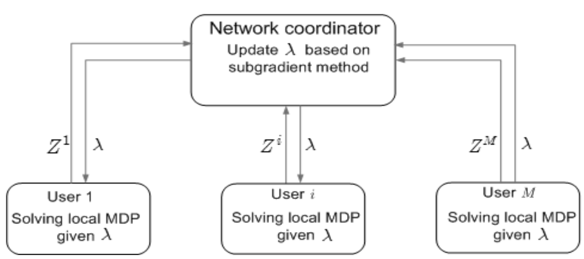
50 Resource price update

- Subgradient method to update resource price

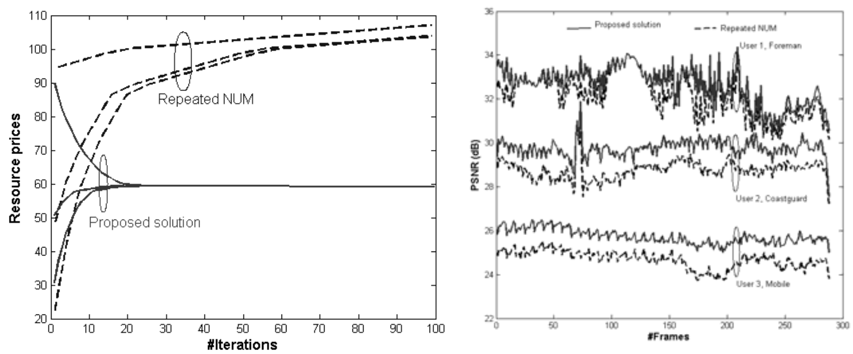
Theorem 3. The resource price is updated by

$$\lambda^{k+1} = \left[\lambda^k + \beta^k\left(\boxed{\sum_{i=1}^{M} Z^i - \frac{1}{1-\alpha}}\right)\right]^+,$$ subgradient

where $Z^i = \boldsymbol{e}_{s_0^i}^T\left(I - \alpha P^i\right)^{-1}\boldsymbol{x}^i(\lambda)$



51 Multi-user dual solution-convergence



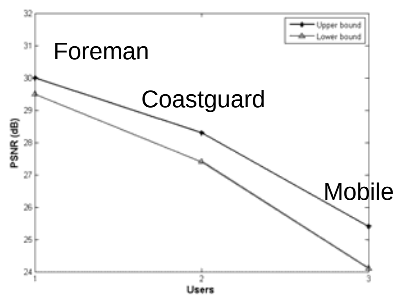
52 Resource allocation

- Using uniform resource price, each user computes its own resource requirement: $x^{i,\lambda^*}\left(s^i\right) = \arg\max_{x^i \geq 0} H^i\left(s^i, \lambda, x^i\right)$
- However, it may happen that $\sum_{i=1}^{M} x^{i,\lambda^*}\left(s^i\right) > 1$
- Gradient-based scaling:

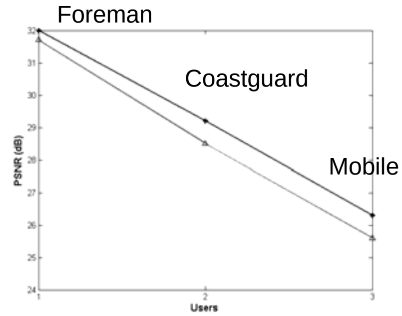
$$\hat{x}^{i,\lambda^*}\left(s^i\right) = \frac{\nabla H^i \cdot x^{i,\lambda^*}\left(s^i\right)}{\sum_{j=1}^{M} \nabla H^j \cdot x^{j,\lambda^*}\left(s^j\right)}$$

- This feasible solution provides a lower bound for the optimal solution

53 Upper & lower bounds



Users experienced with average channel conditions 22dB

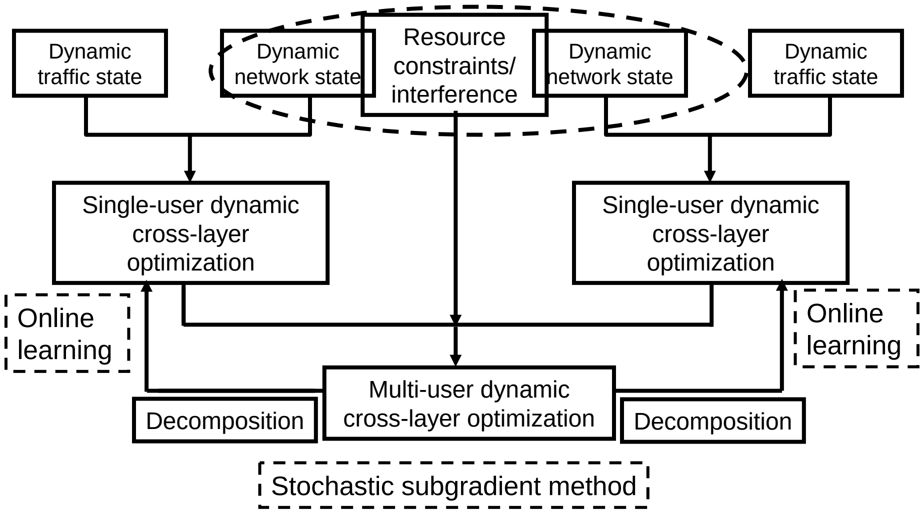


Users experienced with average channel conditions 28dB

54 What problems are remaining?

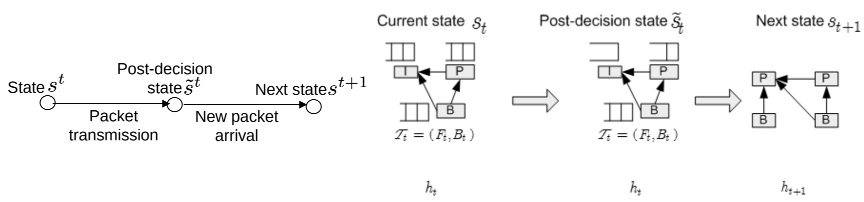
- High computation complexity still exists in each user to solve its own local MDP problem
- Channel states and incoming DUs dynamics are often difficult to characterize a priori
- Without being able to optimally solve the local MDP, the subgradient cannot be computed.

55 Summary



56 Post-decision state and online learning

- Post-decision state



- Bellman's equation on post-decision state

Post-decision state value function

New DUs' dynamics   DU type transition   Channel state transition

$$U^i\left(\tilde{s}_t^i, \lambda\right) = \boxed{\prod_{f_j^g \in F_{t+1}^d \cap (F_{t+1} \setminus F_t)} PMF_j\left(b_j^g\right) \cdot \delta\left(F_{t+1}^i - \text{next}(\tilde{F}_t^i)\right) p\left(h_{t+1}^i \mid \tilde{h}_t^i\right)} \cdot$$

Expectation

$$\boxed{\max_{x_{t+1}^i \geq 0, \boldsymbol{y}_{t+1}^i \in \mathcal{P}^i\left(s_{t+1}^i, x_{t+1}^i\right)}} \left[u^i\left(s_{t+1}^i, \boldsymbol{y}_{t+1}^i, x_{t+1}^i\right) - \lambda x_{t+1}^i + \frac{1}{M}\lambda + \alpha U^i\left(\left(\left(F_{t+1}^i, B_{t+1}^i - \boldsymbol{y}_{t+1}^i\right), h_{t+1}^i\right), \lambda\right) \right]$$

Maximization

Expectation is independent of buffer size!

57 **Batch update for post-decision state value function**

- Specifically, the channel state transition probability and packet arrival probability are the same for all the states in the set of

$$\tilde{S}_t^i = \left\{\left(\left(F_t^i, \tilde{B}_t^i\right), h_t^i\right), \forall \tilde{B}_t^i\right\}$$

- Batch update

$$U^i\left(\tilde{s}^i, \lambda\right) = \left(1 - \gamma_t^i\right) U^i\left(\tilde{s}^i, \lambda\right) + \gamma_t^i \cdot$$
$$\max_{x^i \geq 0, \boldsymbol{y}^i \in \mathcal{P}^i\left(s'^i, x^i\right)} \left[u^i\left(s'^i, \boldsymbol{y}^i, x^i\right) - \lambda x^i + \frac{1}{M}\lambda + \alpha U^i\left(\tilde{s}'^i, \lambda\right)\right], \forall \tilde{s}^i \in \tilde{S}_t^i$$

Theorem: the subgradient-based resource price update and batch update converge to the optimal resource price and corresponding post-decision state value function if

$$\sum_{k=1}^{\infty} \beta^k = \infty, \sum_{k=1}^{\infty} \left(\beta^k\right)^2 < \infty \quad \sum_{k=1}^{\infty} \gamma_k^i = \infty, \sum_{k=1}^{\infty} \left(\gamma_k^i\right)^2 < \infty \quad \lim_{k \to \infty} \beta^k / \gamma_k^i = 0$$

Compared to conventional reinforcement learning, batch update significantly improves the learning rate!

58 **Stochastic subgradient method**

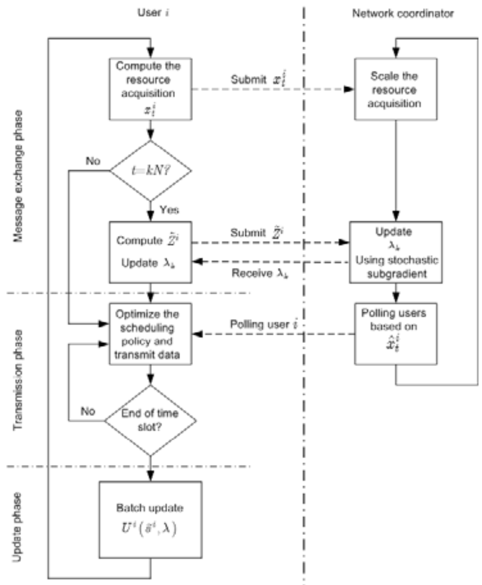
- Subgradient for each user

$$Z^i = \boldsymbol{e}_{s_0^i}^T \left(I - \alpha P^i\right)^{-1} \boldsymbol{x}^i(\lambda)$$

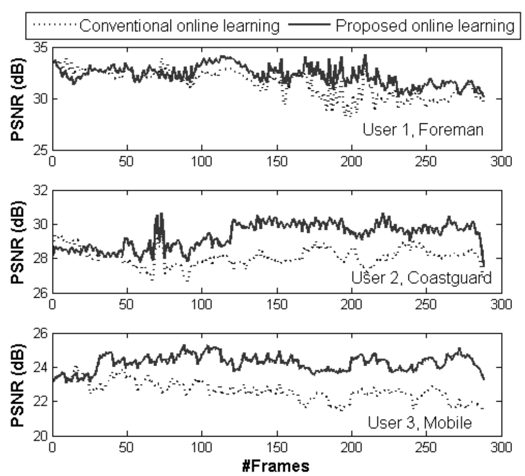
- Stochastic subgradient

$$Z^i = \sum_{t=0}^{\infty} (\alpha)^t \left(x_t^i\right)$$
$$\approx \tilde{Z}^i = \sum_{t=kK}^{(k+1)K-1} (\alpha)^{t-kK} \left(x_t^i\right)$$

59 **How to implement the online learning**



60 **Results on online learning**



1.Received PSNR of each user with different online algorithms

61

Conclusions

- Layered solution for cross-layer optimization
  - Optimal QoS frontier
  - Layered DP operator
  - Optimal message exchange across layers
- Structural results of the optimal solution to cross-layer optimization
  - DAG expression for transmission priority
  - Marginal utility-driven packet scheduling
  - Convexity of the utility of allocated resource
- Decomposition for dynamic multi-user cross-layer optimization
  - Each user solves its own dynamic cross-layer optimization (local MDP)
  - Easy implementation of network coordinator
  - Message exchange across users
- Other applications
  - Multi-core media encoding and decoding (optimal frame scheduling)
  - Cross-layer optimization for multi-hop networks
  - Media-oriented TCP

62

References

- Systematic cross-layer optimization
  - F. Fu and M. van der Schaar, "Structural solutions for cross-layer optimization of wireless multimedia transmission", Tech-report. [http://medianetlab.ee.ucla.edu/papers/UCLATechReport_CLO.pdf]
  - F. Fu and M. van der Schaar, "A Systematic Framework for Dynamically Optimizing Multi-User Video Transmission", Tech-report [http://arxiv.org/ftp/arxiv/papers/0903/0903.0207.pdf]
  - F. Fu and M. van der Schaar, "Decomposition Principles and Online Learning in Cross-Layer Optimization for Delay-Sensitive Applications", *IEEE Trans. Signal Process.*, to appear.
  - F. Fu and M. van der Schaar, "A New Systematic Framework for Autonomous Cross-Layer Optimization," *IEEE Trans. Veh. Tech.*, vol. 58, no. 4, pp. 1887-1903, May, 2009.
  - Y. Zhang, F. Fu, and M. van der Schaar, "Online Learning for Wireless Video Transmission with Limited Information," in *Proc. Int. Packet Video Workshop 2009 (PV 2009).*
- Myopic cross-layer adaption
  - M. van der Schaar, S. Krishnamachari, S. Choi, and X. Xu, "Adaptive cross-layer protection strategies for robust scalable video transmission over 802.11 WLANs," *IEEE J. Sel. Areas Commun.*, vol. 21, no. 10, pp. 1752-1763, Dec. 2003.
  - Q. Li and M. van der Schaar, "Providing adaptive QoS to layered video over wireless local area networks through real-time retry limit adaptation," *IEEE Trans. Multimedia*, vol. 6, no. 2, pp. 278-290, Apr. 2004.
  - M. van der Schaar and S. Shankar, "Cross-layer wireless multimedia transmission: challenges, principles, and new paradigms," *IEEE Wireless Commun. Mag.*, vol. 12, no. 4, pp. 50-58, Aug. 2005.
  - M. van der Schaar, Y. Andreopoulos, and Z. Hu, "Optimized scalable video streaming over IEEE 802.11 a/e HCCA wireless networks under delay constraints," *IEEE Trans. Mobile Comput.*, vol. 5, no. 6, pp. 755-768, June 2006.
  - M. van der Schaar and D. Turaga, "Cross-layer Packetization and Retransmission Strategies for Delay-Sensitive Wireless Multimedia Transmission," *IEEE Trans. Multimedia*, vol. 9, no. 1, pp. 185-197, Jan. 2007.
  - M. van der Schaar, D. Turaga, and R. Wong, "Classification-Based System For Cross-Layer Optimized Wireless Video Transmission," *IEEE Trans. Multimedia*, vol. 8, no. 5, pp. 1082-1095, Oct. 2006.
- Other applications
  - N. Mastronarde and M. van der Schaar, "Online Reinforcement Learning for Dynamic Multimedia Systems," *IEEE Trans. on Image Processing*, to appear.
  - N. Mastronarde and M. van der Schaar, "Towards a general framework for cross-layer decision making in multimedia systems," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 19, no. 5, pp. 719-732, May 2009.
  - H. P. Shiang and M. van der Schaar, " Online Learning in Autonomic Multi-Hop Wireless Networks for Transmitting Mission-Critical Applications", *IEEE J. Sel. Areas Commun.*, to appear
  - H. P. Shiang and M. van der Schaar, "Distributed Resource Management in Multi-hop Cognitive Radio Networks for Delay Sensitive Transmission," *IEEE Trans. Veh. Tech.*, vol. 58, no. 2, pp. 941-953, Feb. 2009.
  - H.P. Shiang, and M. van der Schaar, "Media-TCP", Tech-report, 2009.

Thank you!

MCSL website: http://medianetlab.ee.ucla.edu