Lecture 16
Internet Congestion Control
Adaptive Control in Action?

work with Andrew Liu
Mechanical & Aerospace Engineering
University of California, San Diego
A heads-up

There are lots of control problems in communications at the many different system layers; power control, admission control, congestion control

We wish to look at network congestion control in a stochastic environment

A natural state estimation problem arises

We now know what we mean by stochastic observability?

This is our story ...
TCP as feedback

From the perspective of the source node:
- Packets are sent into the network.
- A packet arriving at the destination produces an ACK packet response.
- The send rate is controlled into the network based on ACK sequence.
- The aim is to avoid traffic congestion.
- The competing traffic is stochastic in nature.
- Normally modeled as dominated by a single bottleneck node.

This is a stochastic feedback control problem.
Destination node behavior

Upon receipt of a packet from the source, the destination node sends an acknowledgement packet in reply.

Generally ACKs are simple few-bit packets arriving packet identifier.

There are proposals for ACKs to contain more and more useful data:

- arrival time
- data inserted by intervening nodes
- buffer state and/or statistics
- traffic state and/or statistics

The ACKs have to travel back through the same network. They too can be lost.
Source node behavior

Data made up of packets with rate or window size $r_k$.

Packet rate into network is the mechanism of congestion control.

Rate $r_k$ is adjusted in response to arrival or non-arrival of ACKs:
- non-arrival: time-out or ACK out of sequence

Common congestion control law AIMD:
- Additive increase / multiplicative decrease
- ACK arrives: increase rate $r_k$ by one packet
- ACK missing: decrease $r_k$ by a factor of two
  - out-of-order receipt also treated as missing
Additive increase / multiplicative decrease

ACK arrives: increase rate \( r_k \) by one packet

ACK missing: decrease \( r_k \) by a factor of two
Additive increase / multiplicative decrease

ACK arrives: increase rate by one packet
ACK missing: decrease by a factor of two

What wacky computer scientist cooked up a wild and crazy control law like that?!!

MAE283B, Lecture 16, Slide 7
Bottleneck node behavior

(source) send rate, \( r_k \)

\( b_k \)

\( b_{\text{max}} \)

\( c_k \)

\( q_k \)

\( y_k \)

(queue) for each source

(for other sources)

(link)

(other destinations)

(Acknowledgment (ACK), \( y_k \))

(bottle-neck) router using RED

(queues for each source) from other sources

MAE283B, Lecture 16, Slide 8
Bottleneck node behavior

Nodes have finite buffers
- buffer occupancy $b_k$
- competing traffic (random) $c_k$
- arrival rate from source $r_k$
- packets deliberately dropped $q_k$

From other sources
- calculate drop probability $p_k$
- drop $q_k$
- calculate $b_k$ up to $b_{\text{max}}$

Source
- send rate $r_k$

Destination
- acknowledgment (ACK) $y_k$
Bottleneck node behavior

Nodes have finite buffers

buffer occupancy $b_k$

competing traffic (random) $c_k$

arrival rate from source $r_k$

packets deliberately dropped $q_k$

Packets are dropped with a certain probability

Random Early Detection (RED) algorithm depends on buffer state

Droptail algorithm drops all packets when buffer full

There are other algorithms
Bottleneck node behavior

Nodes have finite buffers

- Buffer occupancy: $b_k$
- Competing traffic (random): $c_k$
- Arrival rate from source: $r_k$
- Packets deliberately dropped: $q_k$

Packets are dropped with a certain probability $p_k$

Random Early Detection (RED) algorithm

- $p_k$ depends on buffer state $b_k$
  
  $b_k \rightarrow \text{full} \implies p_k \rightarrow 1$

  $b_k \rightarrow \text{empty} \implies p_k \rightarrow 0$

Droptail algorithm drops all packets when buffer full

There are other algorithms
Hidden Markov Model of Bottleneck Node

State of the bottleneck node

\[ x_k = \begin{pmatrix} b_k \\ c_k \end{pmatrix} \]

Drop probability \[ p_k = f(b_k) \]

Capacity (traffic) model - Markov chain

\[ P(c_{k+1} = c) = \sum_d P(c_{k+1} = c | c_k = d) P(c_k = d) \]

Controlled Hidden Markov Model for bottleneck node state

\[ \Pi_x(k+1) = A(r_k) \Pi_x(k) \]
\[ \Pi_y(k) = C \Pi_x(k) \]

Known model - input sequence \( r_k \) and output (ACK) sequence \( y_k \)
A control theorist arrives on the scene
A control theorist arrives on the scene
Can the source computer reliably estimate the state of the bottleneck node?
A control theorist arrives on the scene

Can the source computer reliably estimate the state of the bottleneck node?

Buffer length and capacity value
A control theorist arrives on the scene

Can the source computer reliably estimate the state of the bottleneck node?

Buffer length and capacity value

Available data are input rate history and ACK sequence history
A control theorist arrives on the scene

Can the source computer reliably estimate the state of the bottleneck node?

**Buffer length and capacity value**

**Available data are input rate history and ACK sequence history**

This is an observability question about an HMM
A control theorist arrives on the scene

Can the source computer reliably estimate the state of the bottleneck node?

Buffer length and capacity value

Available data are input rate history and ACK sequence history

This is an observability question about an HMM

A theory of HMM filtering and smoothing exists
A control theorist arrives on the scene

Can the source computer reliably estimate the state of the bottleneck node?

Buffer length and capacity value

Available data are input rate history and ACK sequence history

This is an observability question about an HMM

A theory of HMM filtering and smoothing exists

But what about the observability questions?
A control theorist arrives on the scene

Can the source computer reliably estimate the state of the bottleneck node?

Buffer length and capacity value

Available data are input rate history and ACK sequence history

This is an observability question about an HMM

A theory of HMM filtering and smoothing exists

But what about the observability questions?

We just had a lecture on that. We know everything
HMM Filtering and Smoothing

\[
\Pi_x(k + 1|k) = A(r_k)\Pi_x(k|k)
\]

\[
\Pi_x(k + 1|k + 1) = \frac{1}{[1 \ldots 1]C_{y_{k+1}}\Pi_{k+1|k}C_{y_{k+1}}\Pi_x(k + 1|k)}
\]

where \( C_{y_{k+1}} = \text{diag}(C_{p,1}, C_{p,2}, \ldots, C_{p,n}) \) when \( y_{k+1} = p \)
HMM Filtering and Smoothing

\[
\begin{align*}
\Pi_x(k+1|k) &= A(r_k)\Pi_x(k|k) \\
\Pi_x(k+1|k+1) &= \frac{1}{[1\ldots1]C_{y_{k+1}}\Pi_{k+1|k}} C_{y_{k+1}} \Pi_x(k+1|k)
\end{align*}
\]

where \( C_{y_{k+1}} = \text{diag}(C_{p,1}, C_{p,2}, \ldots, C_{p,n}) \) when \( y_{k+1} = p \)

Reconstruct an estimate of the distribution of the HMM state

Akin to the Kalman filter recursions

Smoothing variants exist with corresponding performance properties

But nobody talks about the observability question

Nor about the dependence of observability on the control law
Performance of the HMM smoother

Comparison of True to Estimated Queue Length

Buffer length estimate

True Queue Length
State Estimate
Performance of the HMM smoother

Comparison of True to Estimated State

Capacity estimate

State

Time

Capacity estimate

State

Time
Back to the internet

HMM observability

\[ \Pi_x(k+1) = A(r_k)\Pi_x(k) \]
\[ \Pi_y(k) = C\Pi_x(k) \]
We need to test the conditional entropy of every function of the HMM state.
Back to the internet

**HMM observability**

\[ \Pi_x(k + 1) = A(r_k)\Pi_x(k) \]
\[ \Pi_y(k) = C\Pi_x(k) \]

We need to test the conditional entropy of every function of the HMM state

Since the output (ACK) sequence is denumerable and the states are denumerable, the number of tested functions is finite
Back to the internet

HMM observability

\[
\Pi_x(k + 1) = A(r_k)\Pi_x(k) \\
\Pi_y(k) = C\Pi_x(k)
\]

We need to test the conditional entropy of every function of the HMM state

Since the output (ACK) sequence is denumerable and the states are denumerable, the number of tested functions is finite

The tests can be exhaustively evaluated
Back to the internet

HMM observability

\[ \Pi_x(k+1) = A(r_k) \Pi_x(k) \]
\[ \Pi_y(k) = C \Pi_x(k) \]

We need to test the conditional entropy of every function of the HMM state.

Since the output (ACK) sequence is denumerable and the states are denumerable, the number of tested functions is finite.

The tests can be exhaustively evaluated.

The test protocol depends on the specific control (source rate) sequence in operation.
Back to the internet

**HMM observability**

\[
\Pi_x(k+1) = A(r_k)\Pi_x(k) \\
\Pi_y(k) = C\Pi_x(k)
\]

We need to test the conditional entropy of every function of the HMM state.

Since the output (ACK) sequence is denumerable and the states are denumerable, the number of tested functions is finite.

The tests can be exhaustively evaluated.

The test protocol depends on the specific control (source rate) sequence in operation.
We need to test the conditional entropy of every function of the HMM state.

Since the output (ACK) sequence is denumerable and the states are denumerable, the number of tested functions is finite.

The tests can be exhaustively evaluated.

The test protocol depends on the specific control (source rate) sequence in operation.

The bottleneck node state is observable/reconstructible from the source when operating with the TCP/IP control.
Performance and reconstrucibility

- True queue length $b_k$
- Simulated – input information only
- Smoothed estimate – input and output information
Dependence of reconstructibility on control

Constant source rate  AIMD source rate

Capacity values of one are not estimated with constant source rate

Observability quantified by mutual information

\[ I_{\text{constant}}(c_0; \{y_k\}) = 0.52143 \]
\[ I_{\text{AIMD}}(c_0; \{y_k\}) = 0.96951 \]
So what’s the point?

Network congestion control led to questions of observability and reconstructibility

Consideration of AIMD versus constant rate source control led to quantification of reconstructibility via $I(x;y)(r)$

This AIMD control law might not be too crazy after all

It makes the capacity variation visible in the $r_k$ and ACK signals

Are the communications people really the first to implement an meaningful and effective dual adaptive control law?

I hope so.
Estimator Performance

Kullback-Leibler Divergence (averaged over 20 simulations)

- Blue line: State Propogation
- Red line: Estimator

Divergence vs. Time (0 to 50)

Tuesday, April 3, 2012
Some traps for young players

Observability/reconstructibility depend in part on the state distribution.

There is a need to consider the possibility that the state covariance might be singular. No improvement is possible.

For stochastic linear systems, observability does not imply reconstructibility: \[ \Sigma_0 = 0, C' = 0, \Sigma_{n-1|n-1} = \Sigma_{n-1} \]

This dependence on distribution focuses attention on the information content.
Convergence of Conditional Entropy

Time Evolution of Entropy

Conditional Entropy $H(x(0))$ Given Outputs
Where to from here?

TCP has thrown up the question of the effect on observability of feedback control laws for network state estimation

Does this idea have more legs?

Dual adaptive control - Fel’dbaum early 1960s

Poorly known system parameter

State/regressor dependent on control

Noisy output measurements

Given prior distribution on parameter and state

Find feedback control $u_k$ to minimize

Very hard problem

Control must probe and regulate the system

Control $u_k$ needs to affect observability of the system
Bonus slide

At no extra charge!
Bonus slide
The time-invariant (uncontrolled) HMM defined by

is completely stochastically observable for any initial state distribution provided the rank of , the dimension of the state.
The time-invariant (uncontrolled) HMM defined by

\[
\begin{align*}
\Pi_x(k+1) &= A \Pi_x(k) \\
\Pi_y(k) &= C \Pi_x(k)
\end{align*}
\]

is completely stochastically observable for any initial state distribution \( \Pi_x(0) \) provided the rank of \( \mathcal{O}_n = n \), the dimension of the state

\[
\mathcal{O}_n = \begin{pmatrix}
C \\
CA \\
CA^2 \\
\vdots \\
CA^{n-1}
\end{pmatrix}
\]
Summary of everything

The course has considered linkages between identification and control design

First six lectures on modeling from data and methods for influencing model fit

Next two lecture on the benefits and pitfalls of using closed-loop experimental data and their connection to the needs of control design

Iterative identification and control methods were introduced via a number of approaches including adding caution

The final three lectures were more focused on adaptive control and the observability/identifiability/duality issues

Practical and industrial examples were sprinkled about; combustion instability control, helicopter vibration control, sugar mill control, network congestion control