UNIVERSITY of VIRGINIA

DENALI

A Quantitative Assured Forwarding Service

Nicolas Christin, Jörg Liebeherr, and Tarek F. Abdelzaher

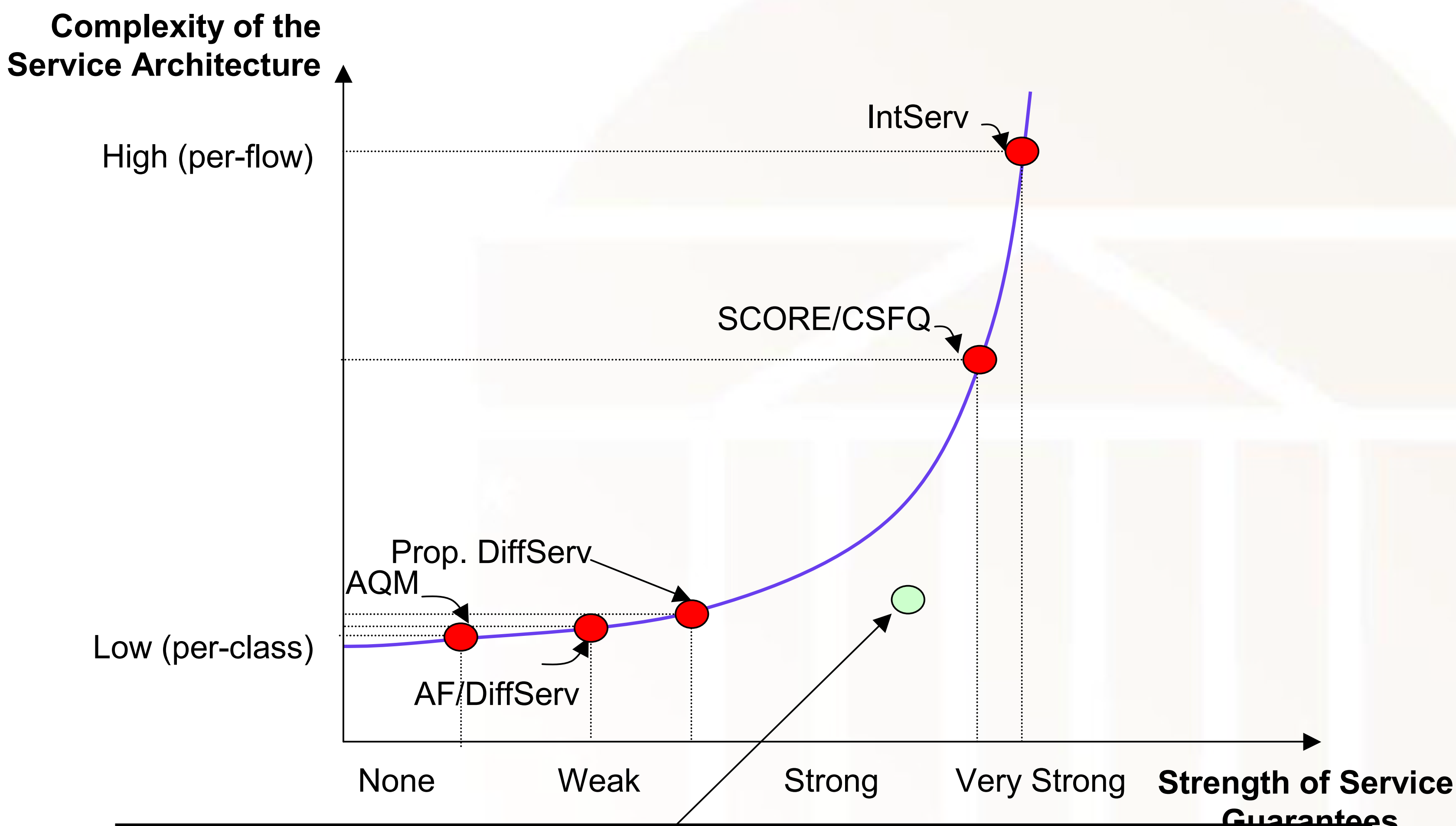
University of Virginia
Department of Computer Science
P.O. Box 400740
Charlottesville VA, 22904-4740

[nicolas|jorg|zaher]@cs.virginia.edu

Outline

- Problem and Context
- Related Work
- The Quantitative Assured Forwarding Service
- Mechanisms for QAF
  - Feedback-control based algorithms
  - Implementation
- Evaluation
  - Service guarantees
  - Overhead
- Conclusions

Problem and Context



Challenge: Can we provide strong service guarantees with low computational complexity?

Related Work

- Flow-Based Service Architectures
  - FRED [Lin and Morris, 1997]
  - SCORE/CSFQ [Stoica and Zhang, 1998, 1999]

- Class-Based Service Architectures
  - Proportional Differentiated Services [Dovrolis et al., 1999, 2000, 2001]
  - Enhancements on the Prop. DiffServ model [Nandagopal et al., 2000][Bodamer, 2000][Bodin et al., 2001]
  - Alternative Best-Effort [Hurley et al., 1999, 2000]
  - JoBS [Liebeherr and Christin, 2001]
  - C-DBP [Striegel and Manimaran, 2002]

Quantitative Assured Forwarding

- Guarantees provided on a per-hop, per-class basis
- No admission control, no signaling, no traffic conditioning
  - No per-flow operations
- **Proportional and absolute** per-class guarantees for both loss and delay and **lower bound on throughput**

$$\frac{\text{Class-2 loss rate}}{\text{Class-1 loss rate}} \approx 2$$

$$\text{Class-2 delay} \leq 5 \text{ ms}$$

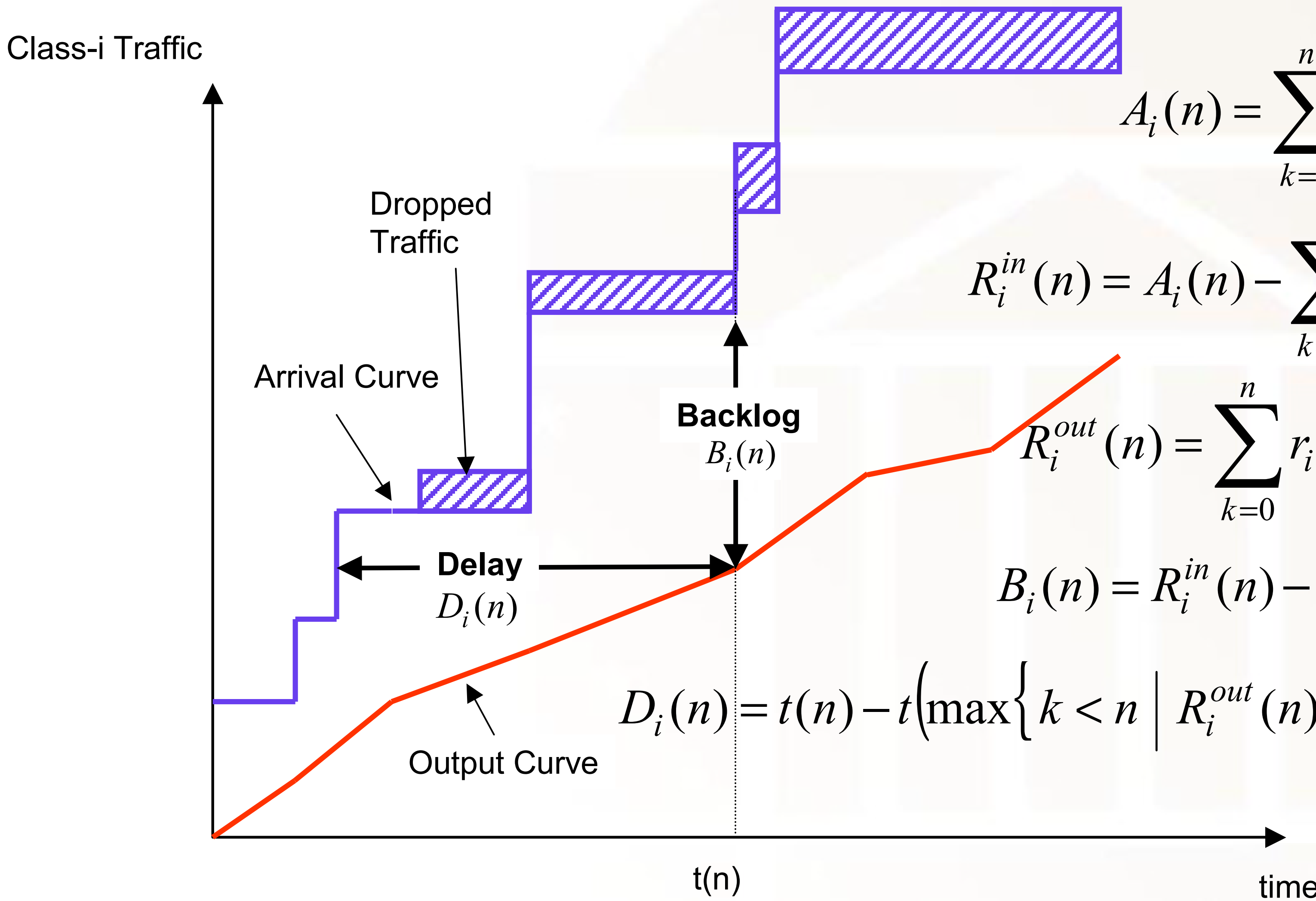
- Concession: service guarantees may need to be temporarily relaxed

None of the existing mechanisms can realize this service

Mechanisms for QAF: Overview

- **Key idea: Manage the head and the tail of the transmission queue in a single algorithm**
  - Combine buffer management and rate allocation
- **How can QAF be implemented?**
  - Service rate allocation to traffic classes, periodically adjusted
  - If no feasible rate allocation exists, drop traffic
  - Rate allocation and packet drop decisions use feedback control
  - If set of guarantees infeasible (no admission control), temporarily relax some guarantees

Arrivals, Departures, Losses at a Node



$$A_i(n) = \sum_{k=0}^{n} a_i(k)$$

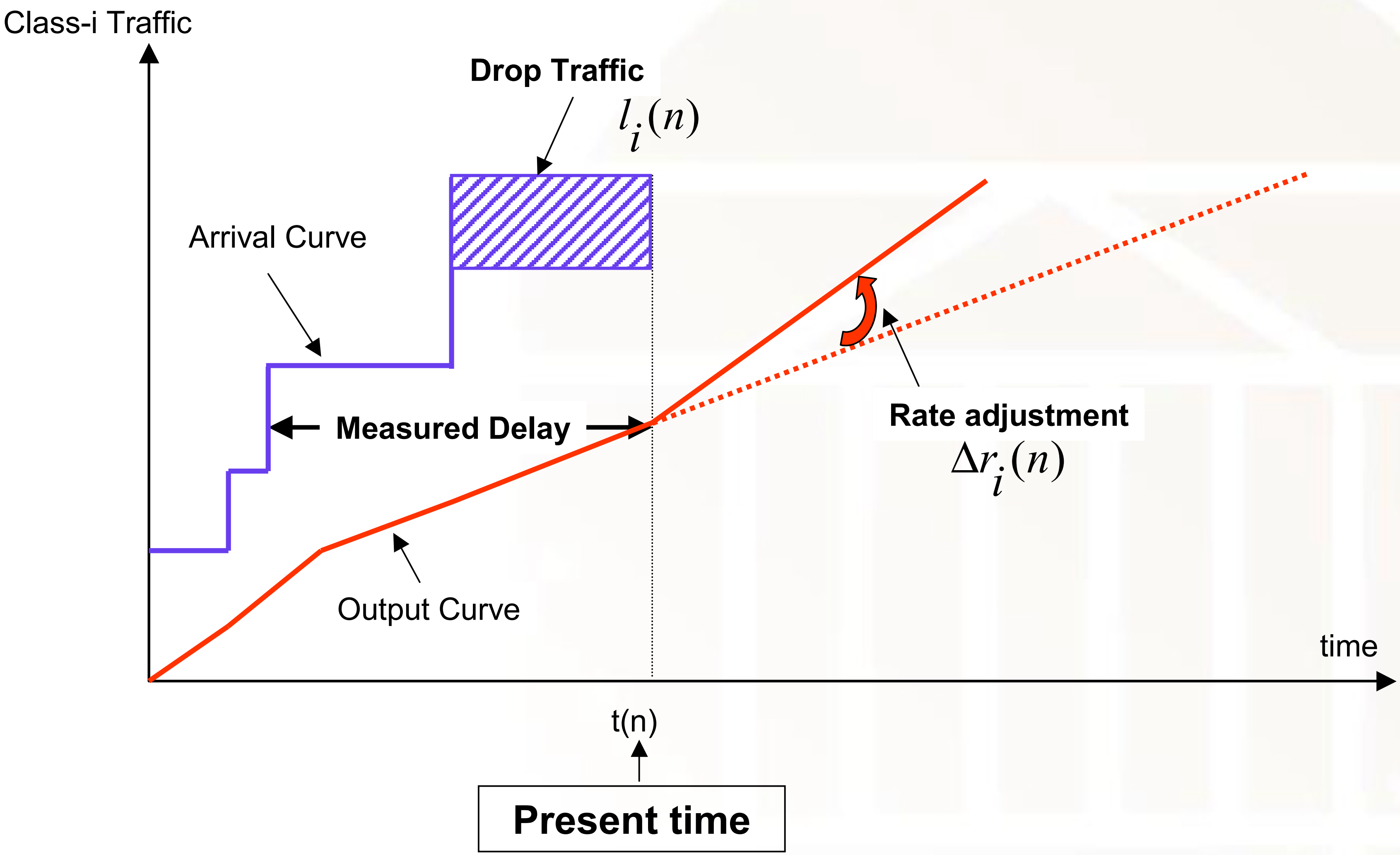
$$R_i^{in}(n) = A_i(n) - \sum_{k=0}^{n} l_i(k)$$

$$R_i^{out}(n) = \sum_{k=0}^{n} r_i(k) \cdot \Delta t(k)$$

$$B_i(n) = R_i^{in}(n) - R_i^{out}(n)$$

$$D_i(n) = t(n) - t\left(\max\left\{ k < n \;\middle|\; R_i^{out}(n) \geq R_i^{in}(k) \right\}\right)$$

Combined Rate Allocation and Buffer Management



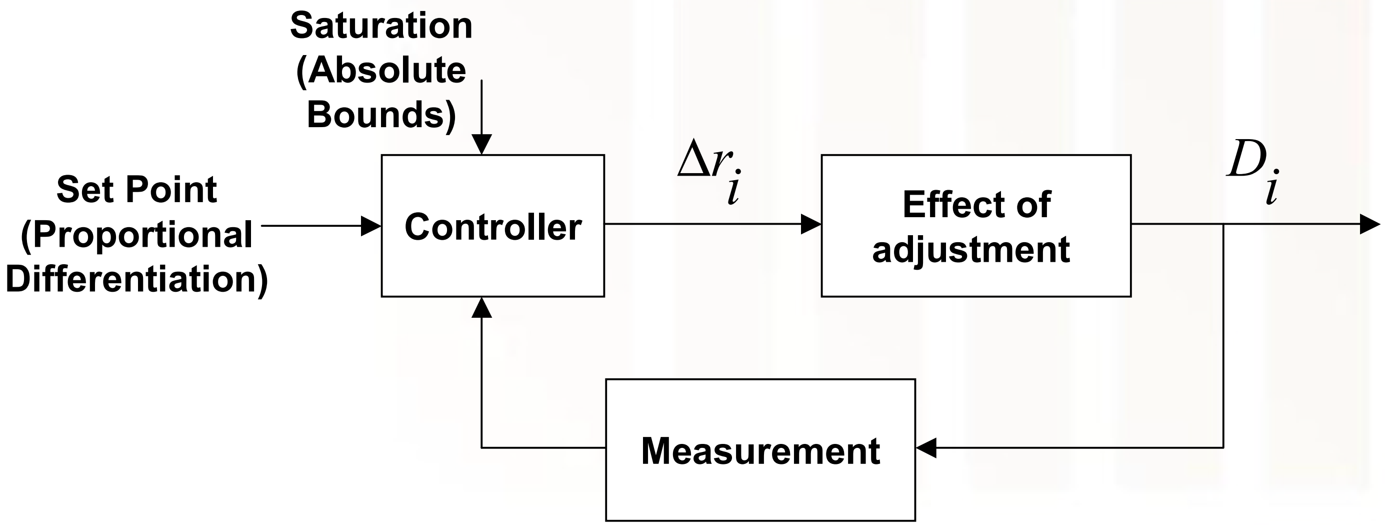
Feedback Loops

- Service rate allocation and loss rates can be viewed in terms of a recursion:

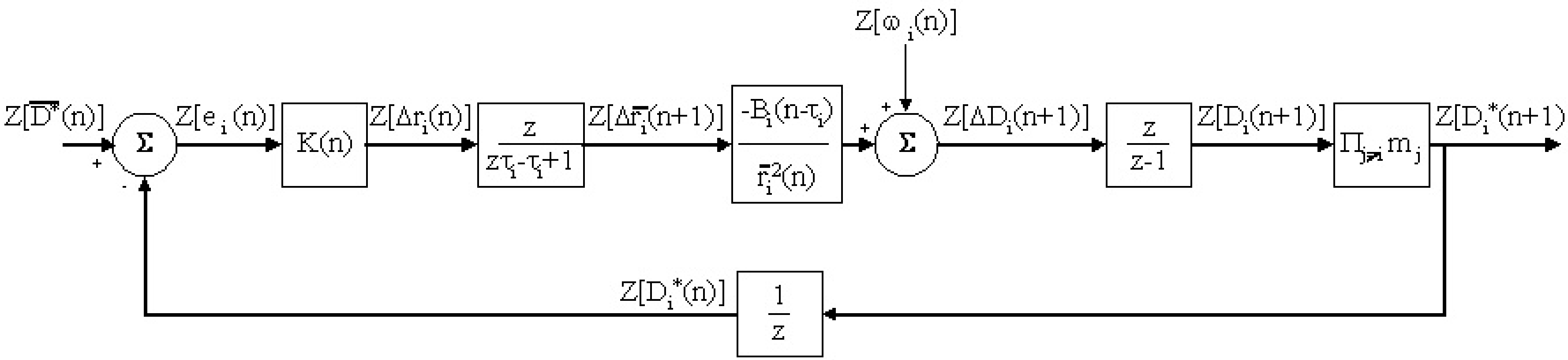
$$r_i(n) = r_i(n-1) + \Delta r_i(n)$$

$$p_i(n) = p_i(n-1)\frac{A_i(n-1)}{A_i(n)} + \frac{l_i(n)}{A_i(n)}$$

- Feedback loops



Delay Feedback Loops



- **One loop per class**
- **Propor**

$$\overline{D}^*(n) = \frac{1}{N}\sum_i D_i^*(n)$$

$$D_i^*(n) = \left(\prod_{k=1,k\neq i}^{N} m_k\right) D_i(n) \quad \text{with} \quad m_i = \prod_{j=1}^{i-1} k_j,\ m_1 = 1$$

  - Eac

$$e_i(n) = \overline{D}^*(n) - D_i^*(n)$$

  - Rate is adjusted by controller: $\Delta r_i(n) = K(n)\cdot e_i(n)$

- **Absolute delay and rate guarantees:** $r_i(n) \geq \mu_i \qquad D_i(n) \leq d_i$
  - Bounds on *K(n)*

Absolute Delay and Rate Guarantees

- Limit the rate adjustment permitted:

$$r_i(n) \geq r_{i,\min}(n)$$

with

$$r_{i,\min}(n) = \max\left\{ \frac{B_i(n)}{d_i - D_i(n)}, \mu_i \cdot \chi_{B_i(n) \geq 0} \right\}$$

- Bound on *K(n)*

$$K(n) \geq \max_i \left(\frac{r_{i,\min}(n) - r_i(n-1)}{e_i(n)} \right)$$

Linearization and Stability

- System is intrinsically non-linear (delay = inverse of the rate)
- Can be linearized with following assumptions
  - Backlog does not change significantly during the time a particular arrival is backlogged
  - $\Delta \overline{r_i}(n) << \overline{r_i}(n)$
- Allows to derive bounds on *K(n)* for stability (I.e., convergence to proportional differentiation):

$$0 \geq K(n) \geq -2 \cdot \min_i \left(\frac{B_i(n)}{\prod_{j \neq i} m_j \cdot D_j^2(n)} \right)$$

Properties of the Controller

- Adjustment is simple:
  - Compute bounds on $K(n)$
  - Compute errors $e_i(n)$
  - Multiply (for each class)
- Work-conserving scheduler!
  - Since $K(n)$ common to all classes:

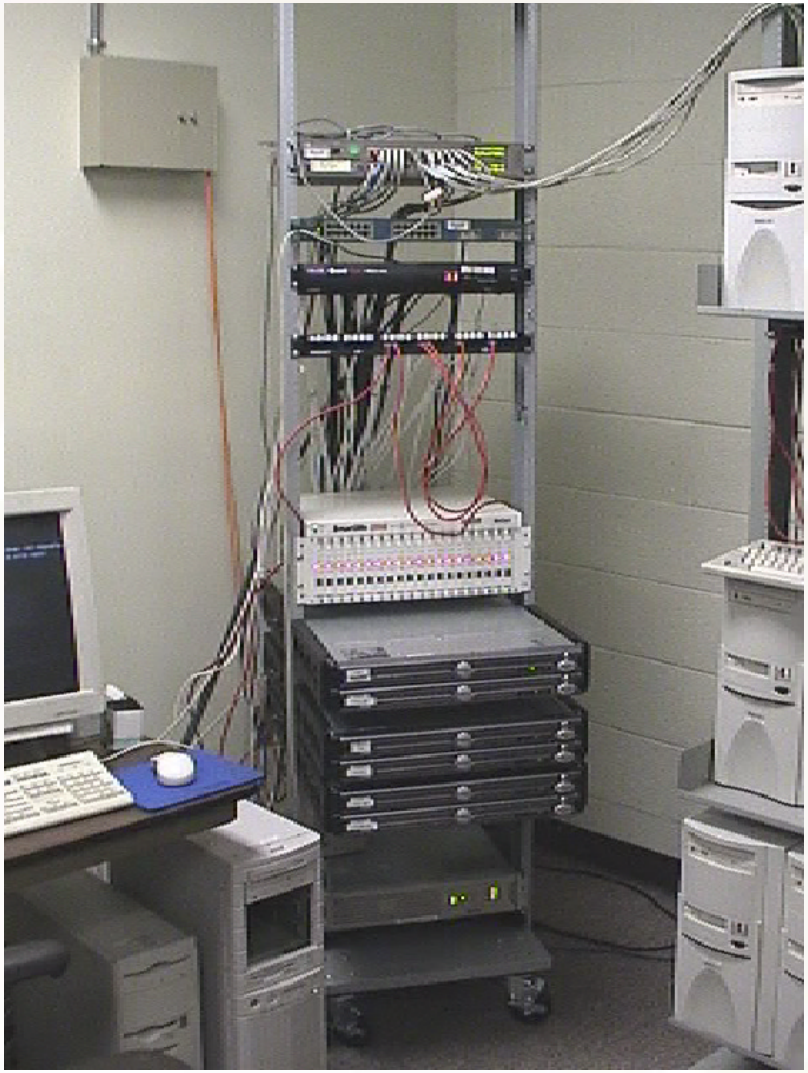
$$\sum e_i(n) = 0 \Rightarrow \sum \Delta r_i(n) = 0$$

Loss Feedback Loops

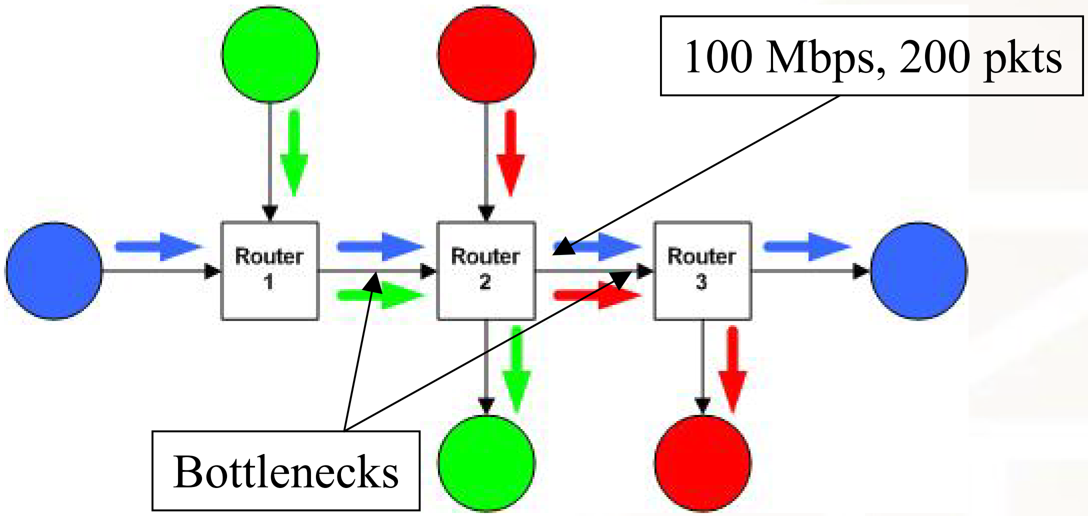
- **No adjustment here, but three decisions to make**
- When to drop
  - Buffer is full
  - Minimum capacity needed for service guarantees exceeds output link capacity
- Which class to drop from
  - Measure distance $e'_i(n)$ between target loss rate (for proportional loss differentiation) and loss rate of class i
  - Drop in increasing order of $e'_i(n)$
- How much traffic to drop
  - Drop as long as buffer is full or minimum capacity needed exceeds output link capacity
  - Stop dropping from a given class when absolute loss rate bound is reached

Implementation

- Implementation in FreeBSD kernel
  - Testbed of 6 Pentium IIIs 1Ghz with multiple interfaces
  - Allows testing at 100 Mbps (FastEthernet)
  - Developed for ALTQ 3.0 (package allowing modifications to the network stack), now part of ALTQ 3.1



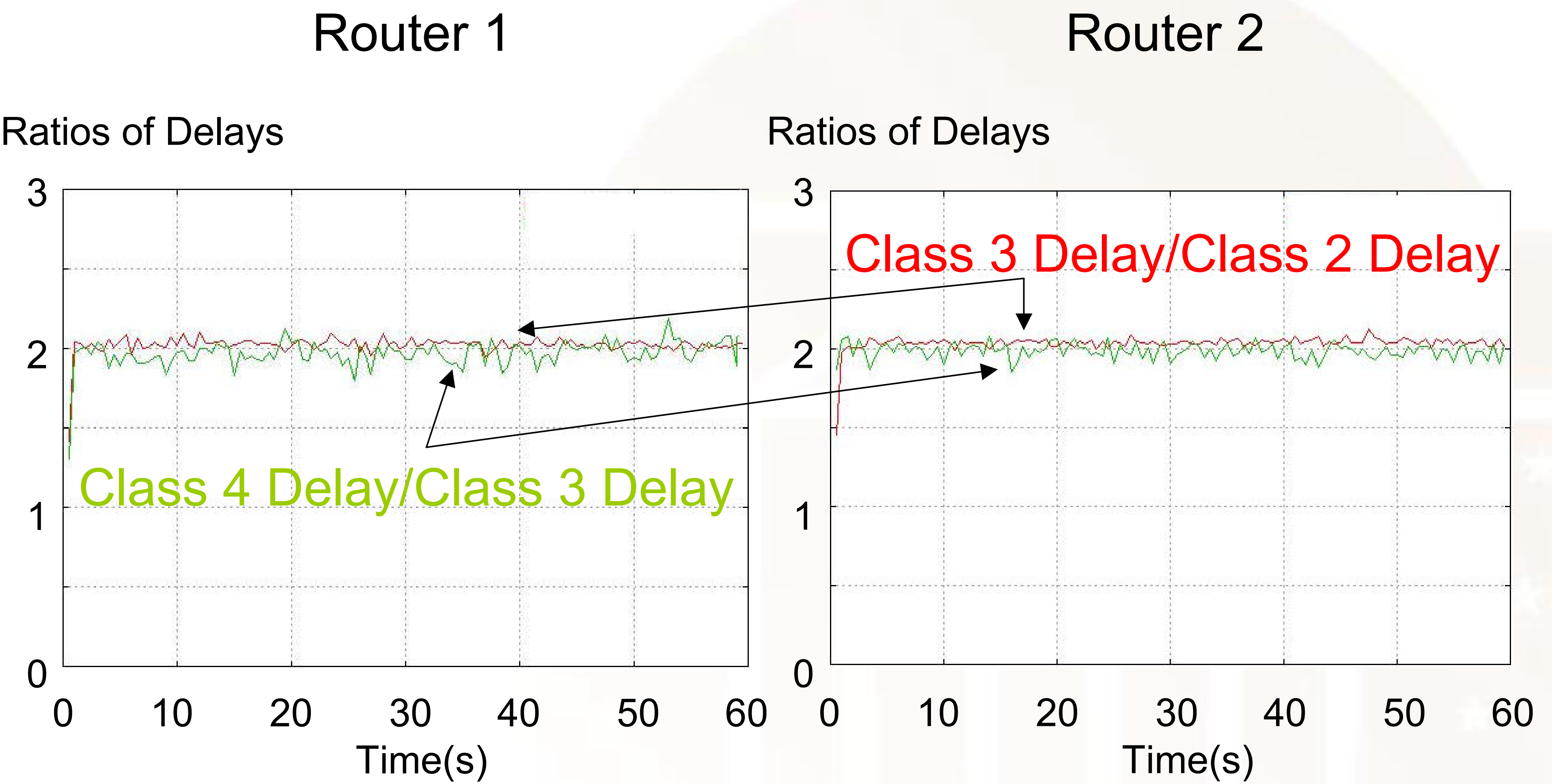
Experimental Setup



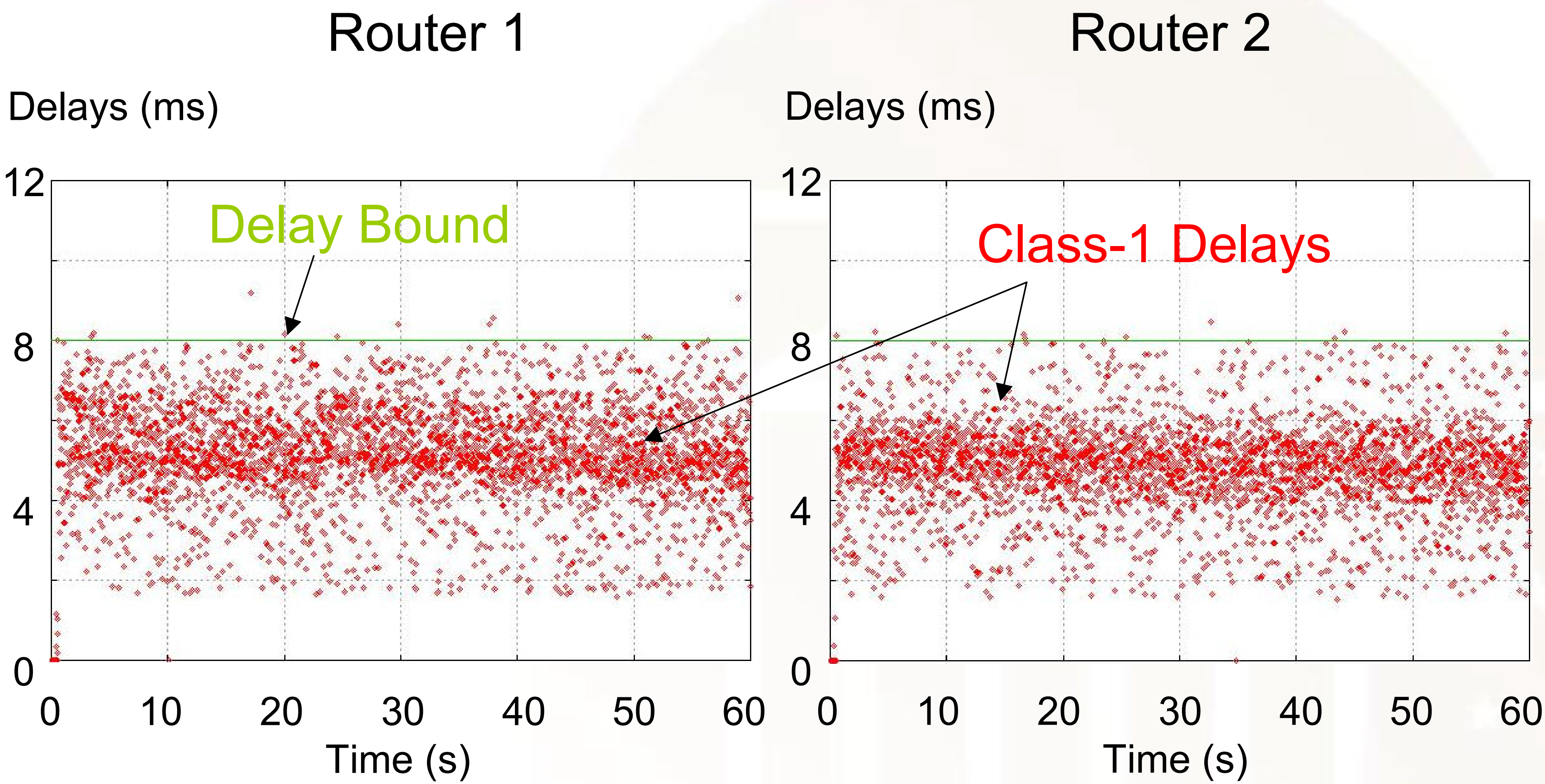
Class	No. of Flows	Proto.	Traffic
1	6	UDP	On-off
2	6	TCP	Greedy
3	6	TCP	Greedy
4	6	TCP	Greedy

Class	d_i	L_i	μ_i	k_i	k'_i
1	8 ms	1 %	-	-	-
2	-	-	35 Mbps	2	2
3	-	-	-	2	2
4	-	-	-	N/A	N/A

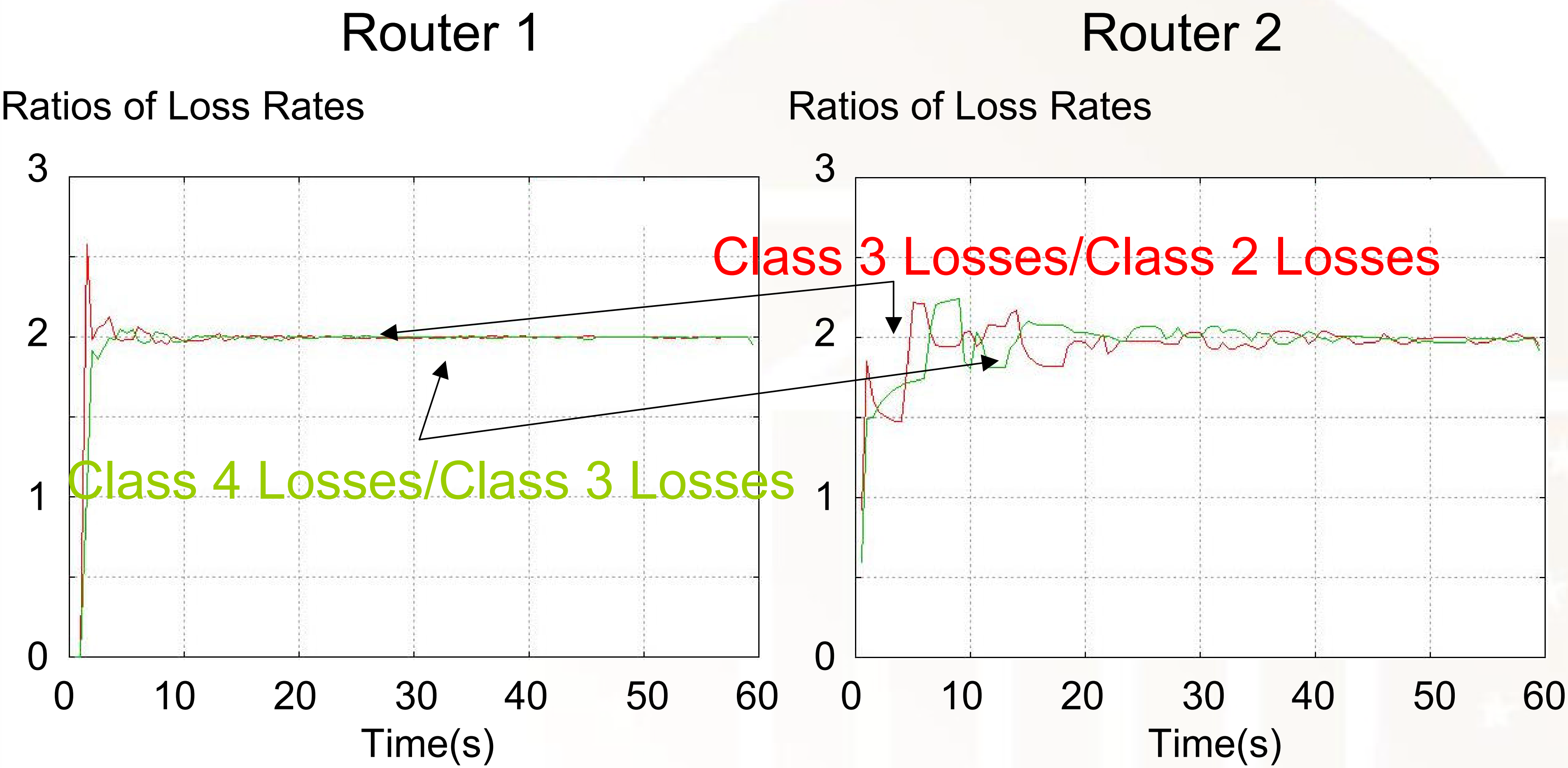
Proportional Delay Differentiation



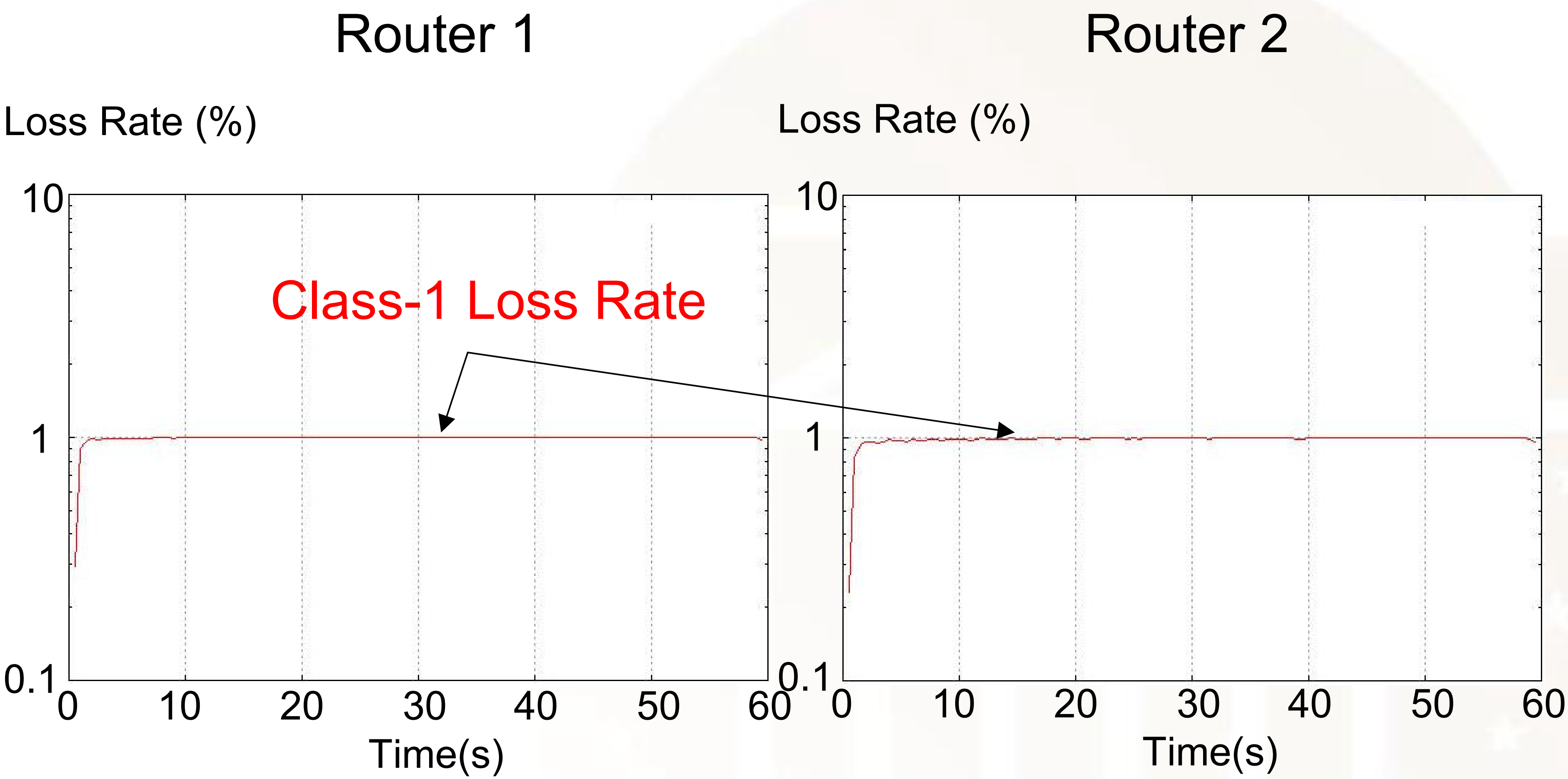
Absolute Delay Bounds



Proportional Loss Differentiation

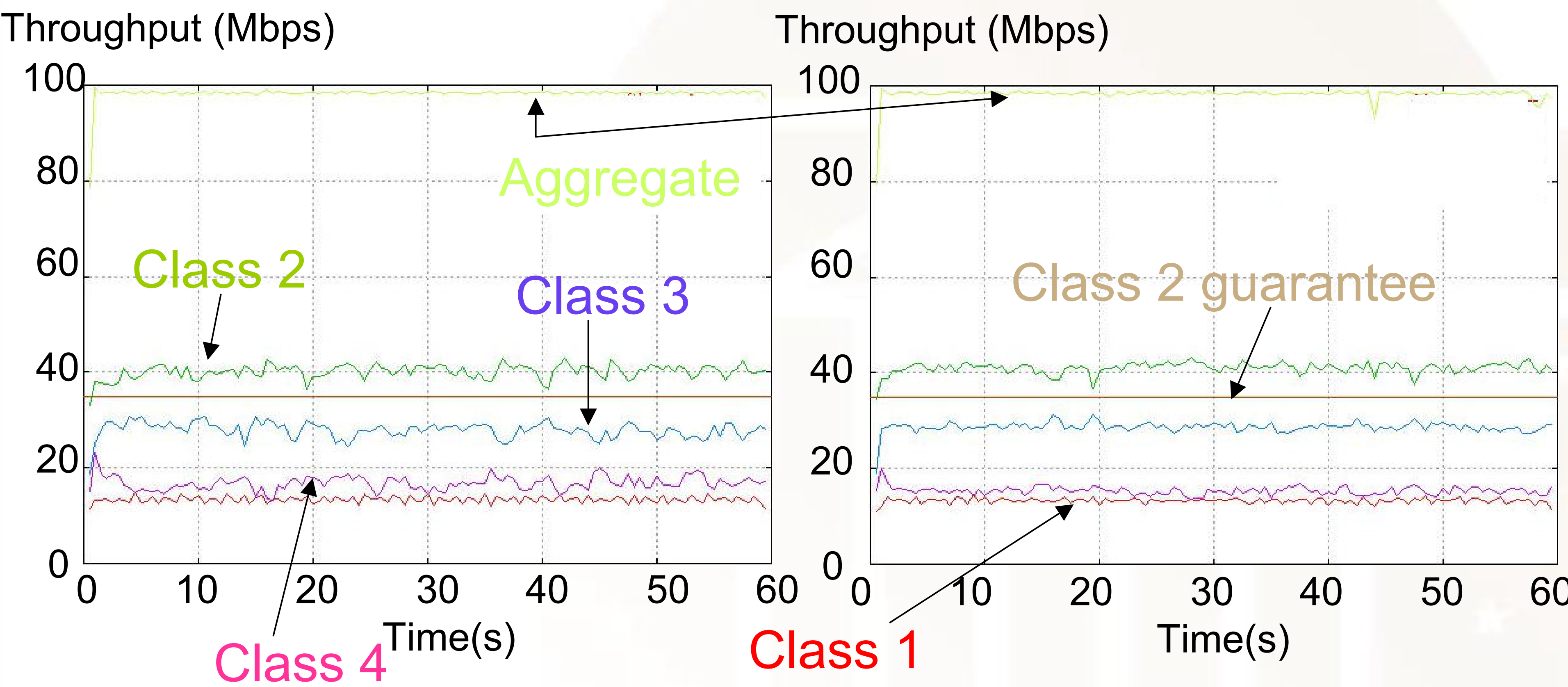


Absolute Loss Rate Bounds



Throughput Differentiation

Router 1 Router 2



Computational Overhead

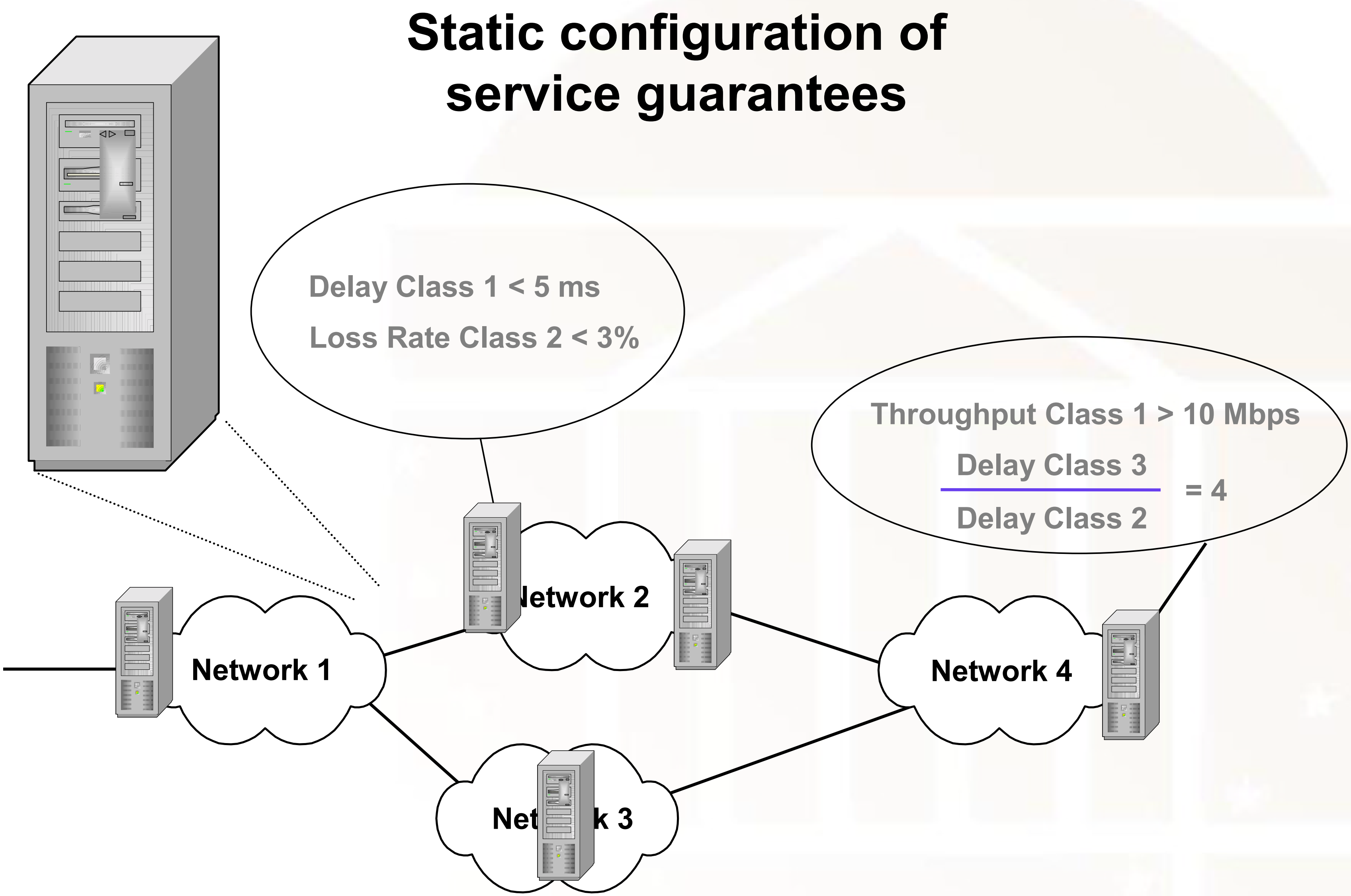
- Two functions:
  - enqueue (feedback loops)
  - dequeue (translation of service rates into packet scheduling decisions)

Number of cycles (1 cycle ≈ 1 ns here)

Guarantees	enqueue		dequeue	
	Avg.	**Std. Dev.**	**Avg.**	**Std. Dev.**
with	15347	2603	4053	912
without	2415	837	3810	858

- A Pentium III-1GHz can process over 50,000 packets/sec.

Deployment


Static configuration of
service guarantees
Delay Class 1 < 5 ms
Loss Rate Class 2 < 3%
Throughput Class 1 > 10 Mbps
Delay Class 3
Delay Class 2
= 4
Network 1
Network 2
Network 4

Conclusions

- Quantitative Assured Forwarding service: subsume per-class service architectures
- Low complexity/Strong guarantees
- Can be implemented at high-speeds
- Current work:
  - Avoid infeasible set of service guarantees by regulating traffic using TCP congestion control algorithms
  - Implementation at Gbps speeds (Network processor)
- Software and more information is available at:

  http://qosbox.cs.virginia.edu