Incentive-Compatible Differentiated Scheduling

HotNets IV - November 2005

Martin Karsten, Yunfeng Lin, Kate Larson

School of Computer Science, University of Waterloo
200 University Ave W
Waterloo, ON N2L 3G1
Canada

Background

Topic: Network Quality of Service

Rate Control...

- **simple (edge) with *rate-neutral* FIFO scheduling → FIFO Principle**

...vs. Delay Control

- **priority scheduling → preferred service class**
- **allocation-based scheduling**

⇒ Multi-class Admission Control → Complicated!

ICDS: Reconciliation of Delay Control and FIFO Principle

- **rate control oblivious to delay control**

Alternative Motivation: Queueing Delay

- ...produced by buffering
- ...required for bursty traffic

→ **Fate-sharing between bursty and smooth traffic?**

Typical "Internet Applications"

- varying flexibility of handling different rates
- some network loss tolerance
- limited number of delay targets
  - e.g. interactive human users
  - for different media types

Scheduler Model

0. Basics

- ICDS provides *n* service classes with fixed delay targets

1. FIFO Principle

- relative service rate = relative arrival rate
- at time *t*: arrival rates *a*, link capacity *C* → compute service rate *r*

$$r_i(t) = C\frac{a_i(t)}{\sum a_j(t)}$$

2. Delay "Guarantee" → Packet Discard

- discard packets that cannot be forwarded in due time
- non-trivial for varying rate allocation...

Scheduler Model - Game-theoretic Properties

Game

- each player (traffic source) has fixed delay target
- each player selfishly chooses service class

Assumptions

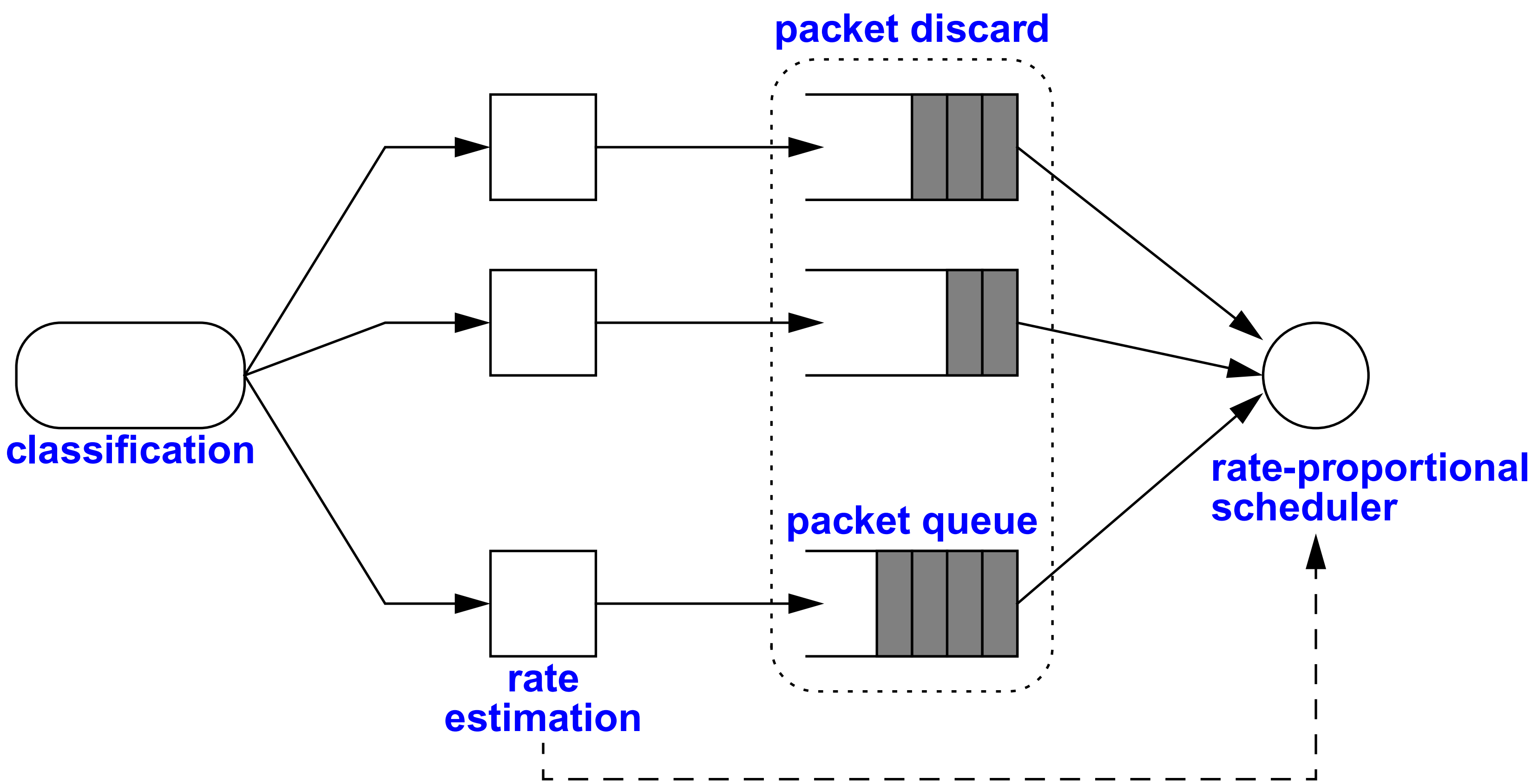
1. lower delay $\Rightarrow$ higher drop rate
2. delay exceeds target $\Rightarrow$ zero utility
3. any delay lower than target $\Rightarrow$ same utility
4. lower drop rate $\Rightarrow$ higher utility
5. service rate (throughput) unaffected by choice of service class

Result: ICDS is strategy-proof

- best strategy is to always choose true delay target
  (that is: highest delay lower than target)

Implementation

Overview



Implementation Details

Rate Estimation

- **avoid arbitrary division → modify Time Sliding Window (TSW)**
- **direct relative estimation: operate on arrived bytes rather than time**

Packet Scheduling

- **limited number of classes: scheduler no big concern?**
- **prototype uses WF^2Q+**

Packet Discard

- **drop on departure? may not be efficient**

Rate Allocation and Delay

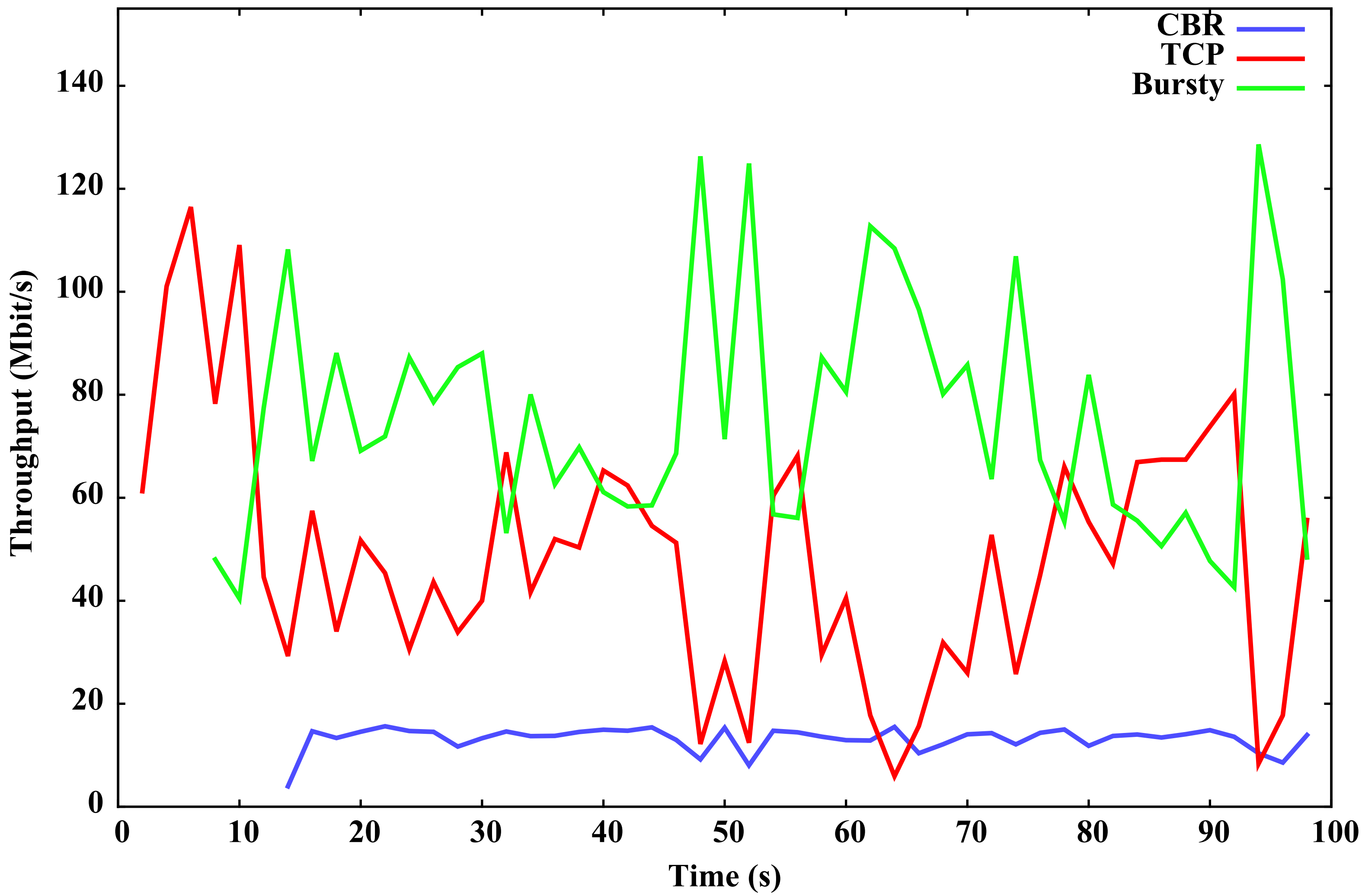
- **loose delay mode: ignore estimation errors and rate variation**
  - introduces errors
- **strict delay mode: account for rate variation**
  - check sum of rates against budget
  - implement rate increase immediately
  - implement rate reduction only after previous packets are served
  - conservative scheme → reduced resource (buffer) utilization

Evaluation

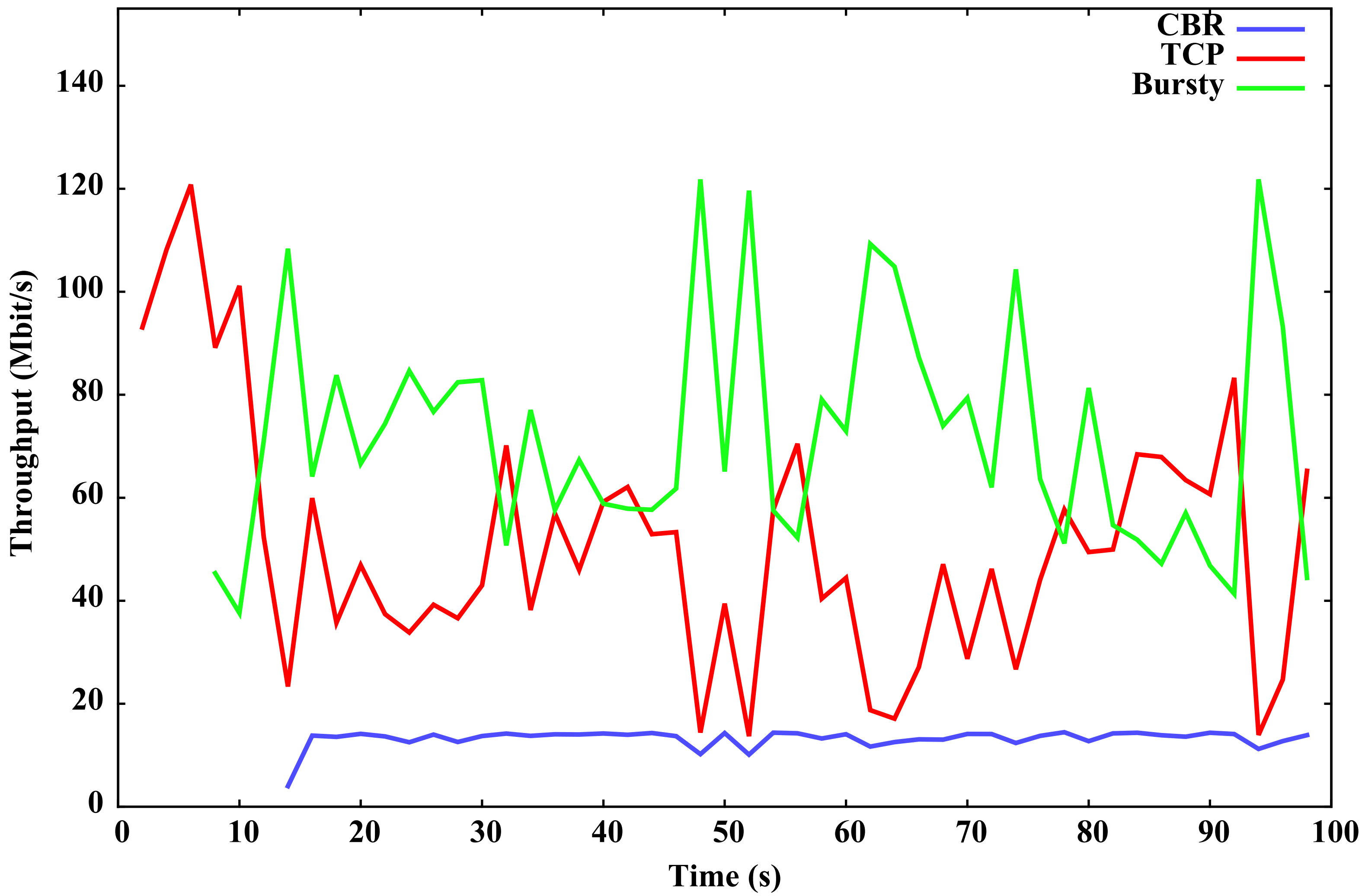
Simulation Experiment

- **dumbbell topology with 155 Mbit/sec at bottleneck**
- **end-to-end latency: 30 msec → 60 msec round-trip latency**
- **3 traffic sources**
  - CBR - 1 flow UDP/CBR with 15.5 Mbit/sec (10%)
  - TCP - 100 flows TCP/Greedy
  - Bursty - 32 flows UDP/Pareto with 93 Mbit/sec average rate (60%)
- **FIFO: 60 msec buffer**
- **ICDS: 3 delay classes**
  - 10 msec
  - 30 msec
  - 60 msec
- **ICDS loose-delay mode ⇒ occasional delay violations**

FIFO with 60 msec buffer



ICDS with CBR in 10, TCP in 30, and Bursty in 60 msec class



Evaluation (cont'd)

Average Throughput in Mbit/sec

Scenario (CBR/TCP/Bursty)	CBR	TCP	Bursty
FIFO (60/60/60)	13.6	44.5	75.9
ICDS (10/30/60)	13.5	45.2	72.4
ICDS (10/10/60)	14.1	34.5	75.1
ICDS (10/30/30)	13.7	33.0	72.0
ICDS (10/60/60)	13.6	42.6	75.3
ICDS (10/30/10)	12.4	50.5	59.2

- **ICDS (10/30/60) provides "best" performance**
- **"cheating" does not help**
- **TCP can be affected by competing traffic - see ICDS (10/30/30)**
  - no gain for Bursty $\rightarrow$ denial-of-service only
- **TCP target not obvious - compare ICDS (10/30/30) with ICDS (10/60/60)**

Discussion

Essence of ICDS

- **proper incentives for burst control and/or traffic shaping**
- **policy-free delay differentiation**
- **no more fate-sharing for smooth and bursty traffic**

Deployment Scenarios

- **isolated deployment: delay differentiation without control regime**
  - overloaded nodes without sophisticated traffic management
  - e.g. peering exchanges?
  - end-to-end rate control
- **domain deployment: admission control at edge gateways**
  - no static resource partitioning
  - no signalling with internal nodes
  - multiple bottlenecks: no pay-bursts-once principle

Traffic Aggregation

- **“misbehaving” flows: strong enough incentives?**
- **...or traffic shaping at input ports needed?**

Wrap Up

FIFO Principle vs. Delay Control

- ICDS reconciles both
- incentives for traffic shaping, if low delay wanted
- low-complexity QoS solution: single-class admission control

Strong Game-theoretic Properties

- with certain assumptions

Implementation Details

- partially solved

Simulation Results

- limited but encouraging

Open Issues

Validity of Game-theoretic Model

- **realistic assumptions?**

Implementation Details

- **non-trivial feedback loop**
  - arrival rate $\rightarrow$ service rate
  - loss $\rightarrow$ sending rate
- **feasible general configuration?**
- **cf. Validity of Game-theoretic Model**
- **implementation efficiency**
  - especially strict delay mode

Multiplexing and Traffic Aggregation

- **robustness?**