Dynamics of End-to-End Bandwidth Allocations in QoS-adaptive Data Connections

Mohammad Rabby & Kaliappa Ravindran

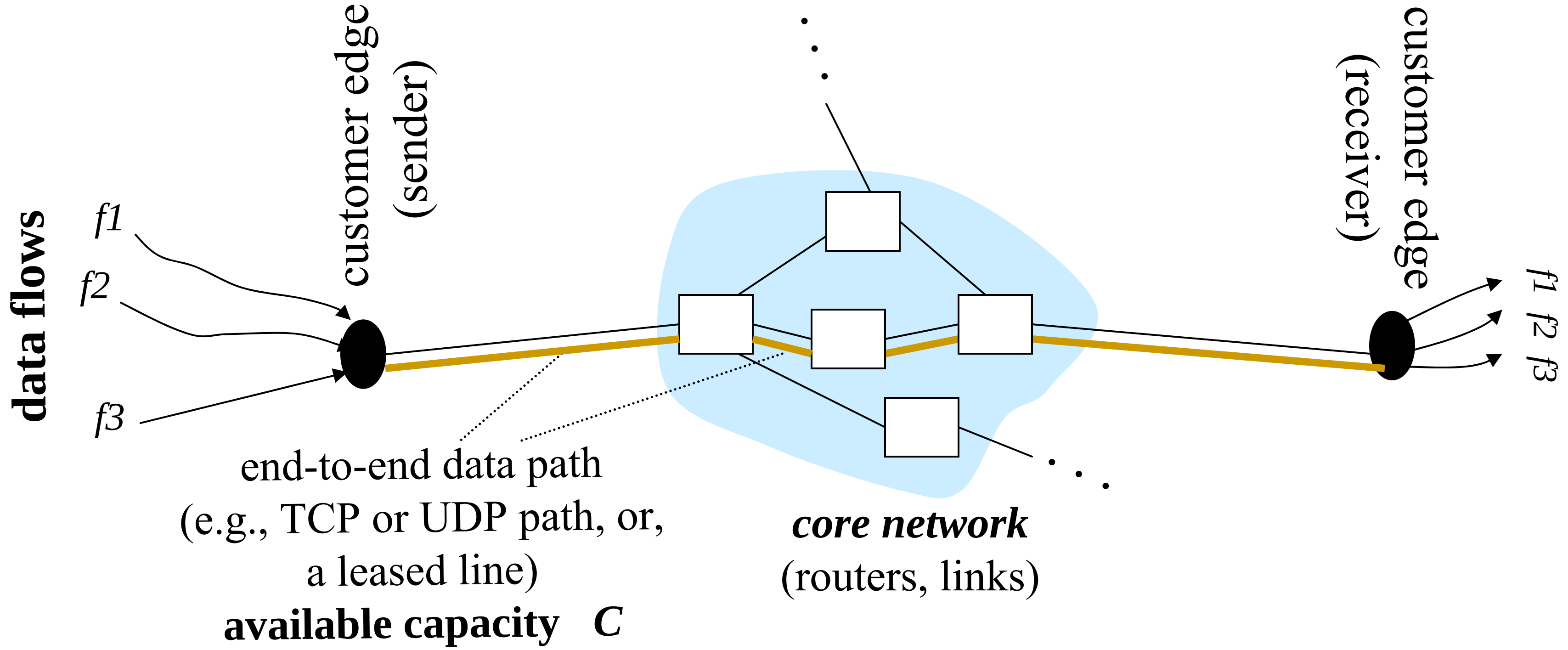
Department of Computer Science
City University of New York

Organization of talk

- ***Application-controlled*** `diffserv'-style flow classification
- 'data connections' *vs* 'data flows'

  [Hose model, class-based queuing, etc]
- Dynamic adaptation control at connection end-points
- Our study of ***end-to-end bandwidth control dynamics***
- Experimental results of simulation
- Summary and future works

Problem statement



data flows

f1

f2

f3

customer edge (sender)

customer edge (receiver)

f1 f2 f3

end-to-end data path
(e.g., TCP or UDP path, or,
a leased line)
available capacity *C*

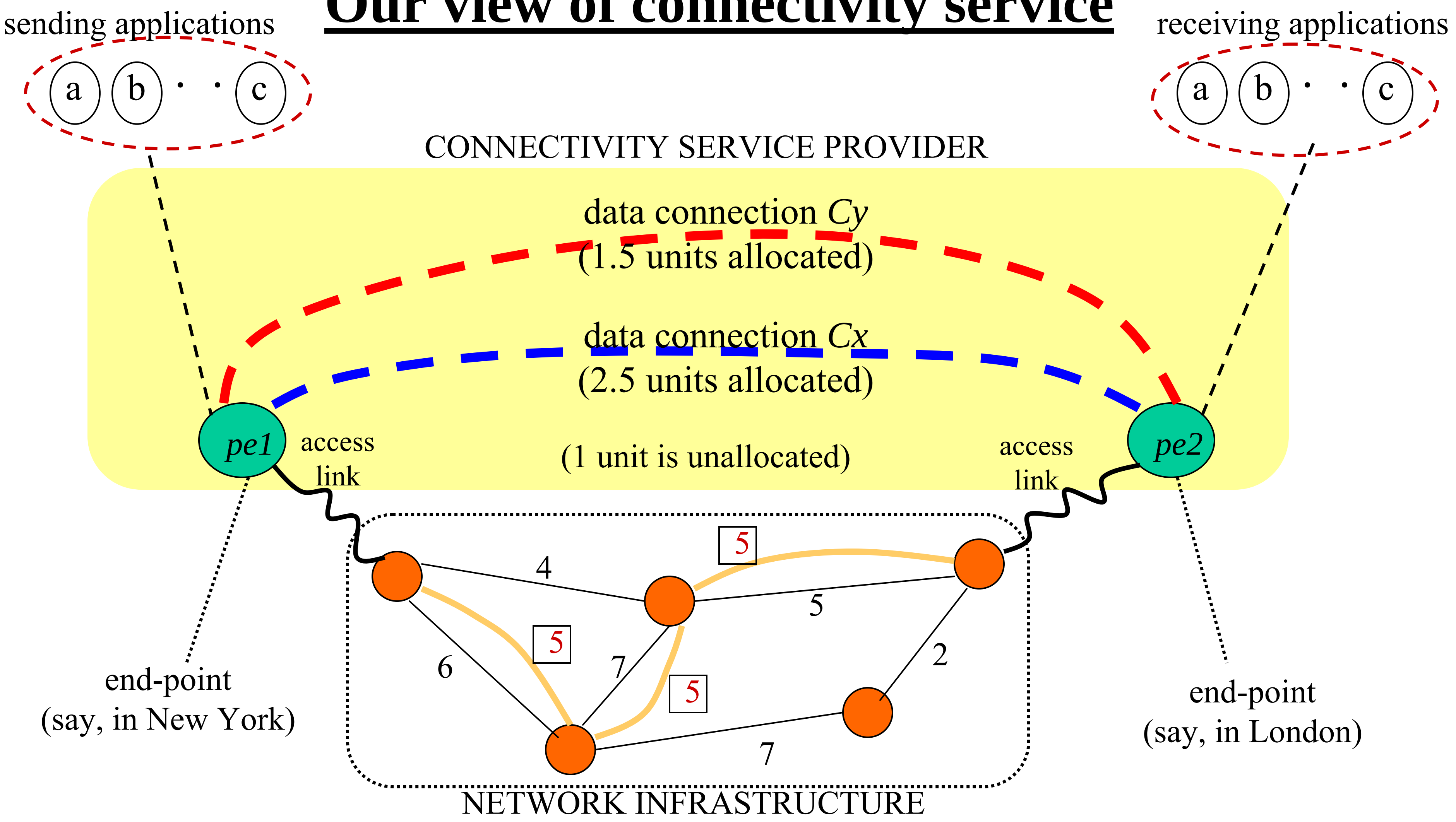
core network
(routers, links)

GOAL: To determine the bandwidth allocation *B* that suffices to carry the data flows *f1*, *f2*, *f3* --- where $B < C$

^^ Path capacity *C* is shared with other data traffic

^^ Connectivity service provider wishes to maximize revenues (by minimizing *B*)

Our view of connectivity service



$\boxed{x}$ path segment with *x* units of bandwidth availability

● *y* Network element in the infrastructure with *y* units of bandwidth capacity

Logical 'data connections' are objects visible to end-systems, to which bandwidth allocations are assigned (each connection carries multiple data flows)

Layered view of bandwidth allocation

Two main characteristics:

1. Fuzziness in estimating the bandwidth needs to service a data flow

   (arises due to the inability to precisely characterize the flow parameters)

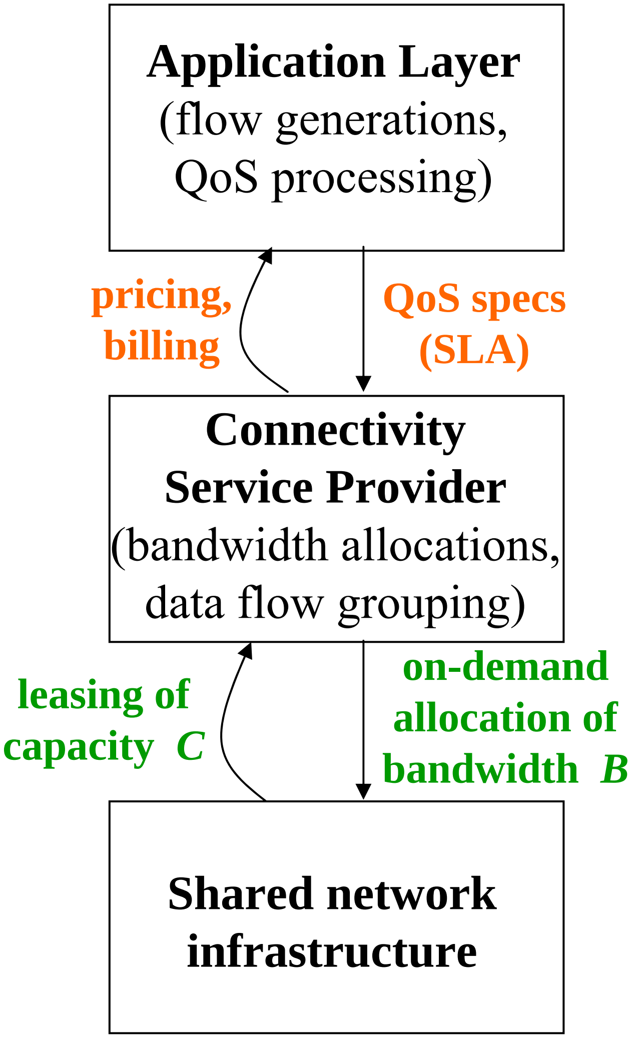
2. Statistical sharing of bandwidth among data flows

   (arises due to the bursty and random nature of bandwidth demands)

bandwidth allocation $B < C$

Service provider's goals:

Maximize revenues by minimizing B



Application Layer
(flow generations,
QoS processing)

pricing, billing

QoS specs (SLA)

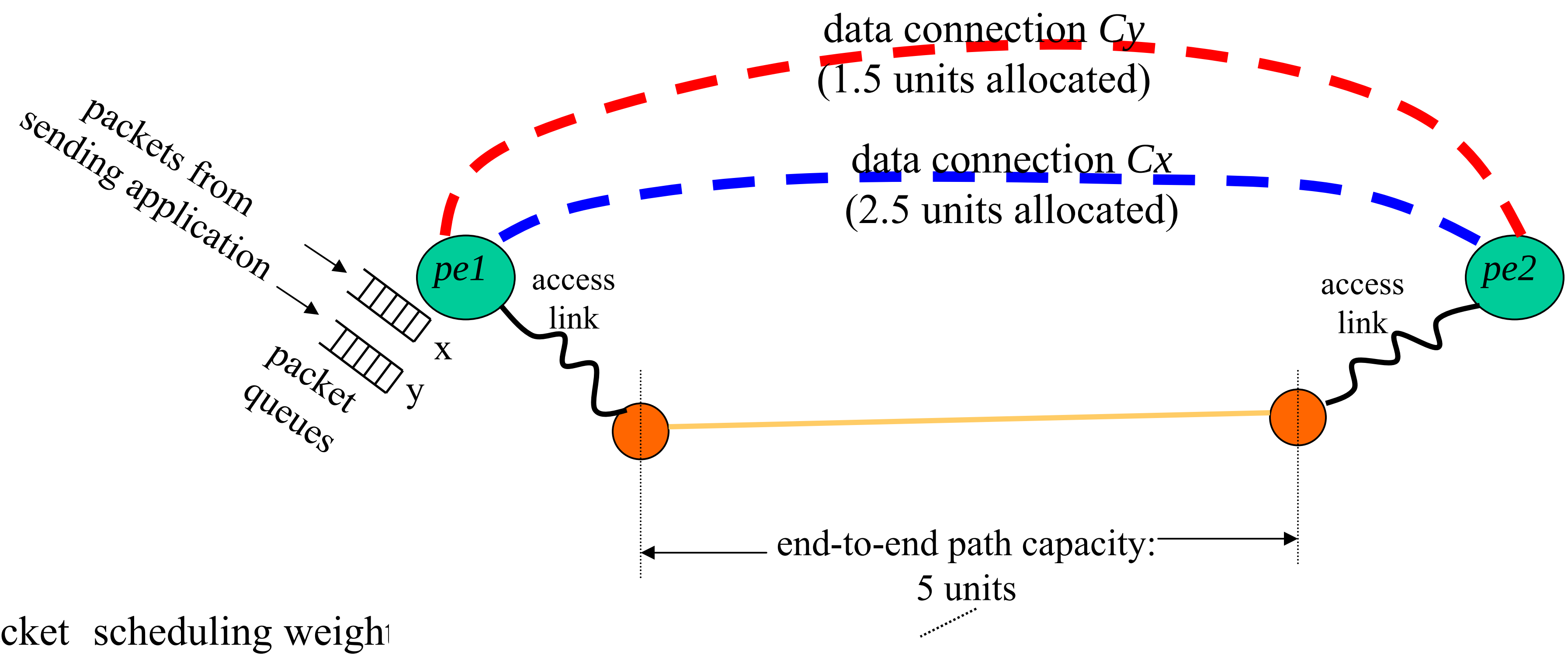
Connectivity Service Provider
(bandwidth allocations,
data flow grouping)

leasing of capacity C

on-demand allocation of bandwidth B

Shared network infrastructure

What exactly "bandwidth allocation" means ?



$$\text{packet scheduling weight} = \frac{\text{bandwidth allocated}}{\text{capacity}}$$

[obtained through auxiliary means (such as 'packet-pair probing in Internet; known readily if it is a leased path)]

bandwidth allocation on *Cx*: 2.5 units ➔ [scheduling weight of packet queue(*x*)=0.5]

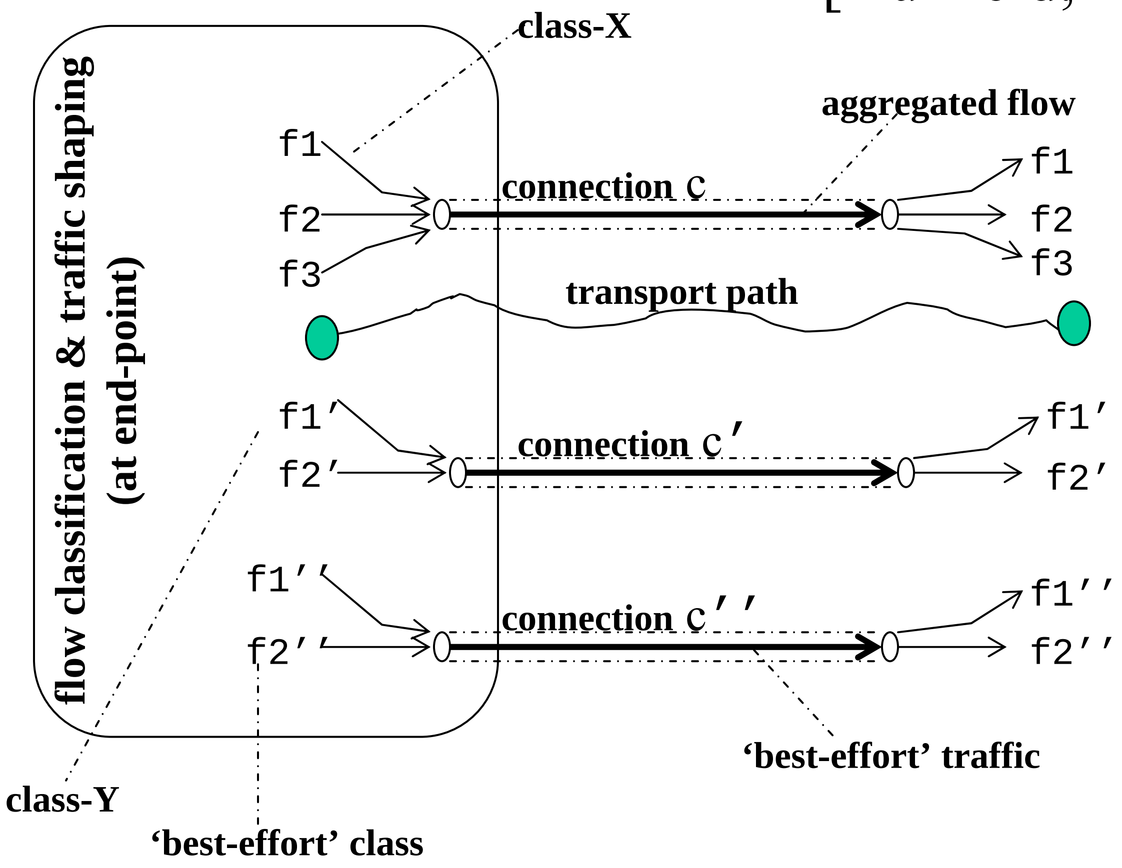
bandwidth allocation on *Cy*: 1.5 units ➔ [scheduling weight of packet queue(*x*)=0.3]

WFQ scheduling of packets from end-point queues

‘data connections’ *versus* ‘data flows’

‘data flow’ → end user-level object for QoS specs

‘data connection’ → system-level object for bandwidth management
(similar to a “hose” in the hose model of VPNs
[Duffield, K.K. Ramakrishnan, et al])

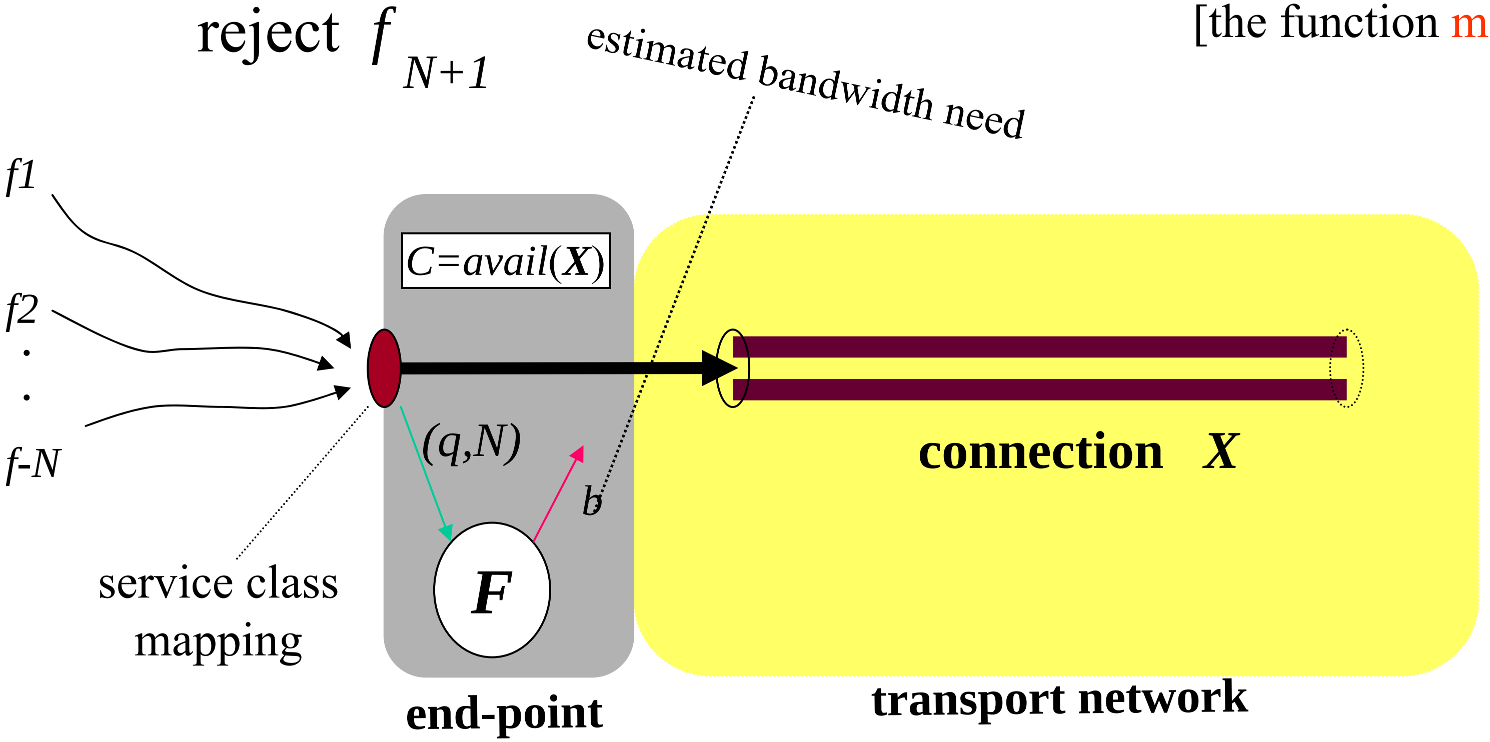


`f1,f2,f3,f1’,f2’ : data flow`

Admission control

```
ADMIT_FLOW ($f_{N+1}$ , $q$, $X$)
    additional bandwidth $\mathbf{Z} := \mathbf{F}(q,N+1) - \mathbf{F}(q,N)$;
    if [$\mathbf{Z} < avail(\mathbf{X})$]
        admit new flow $f_{N+1}$
        $avail(X) := avail(\mathbf{X}) - \mathbf{Z}$;
    else
        reject $f_{N+1}$
```

q: flow descriptor
$\mathbf{F}$ (..): bandwidth estimator for
an aggregated set of flows
[the function may not be known]



Admission control (. . . contd.)

Edge node of an end-to-end path ***P*** may:

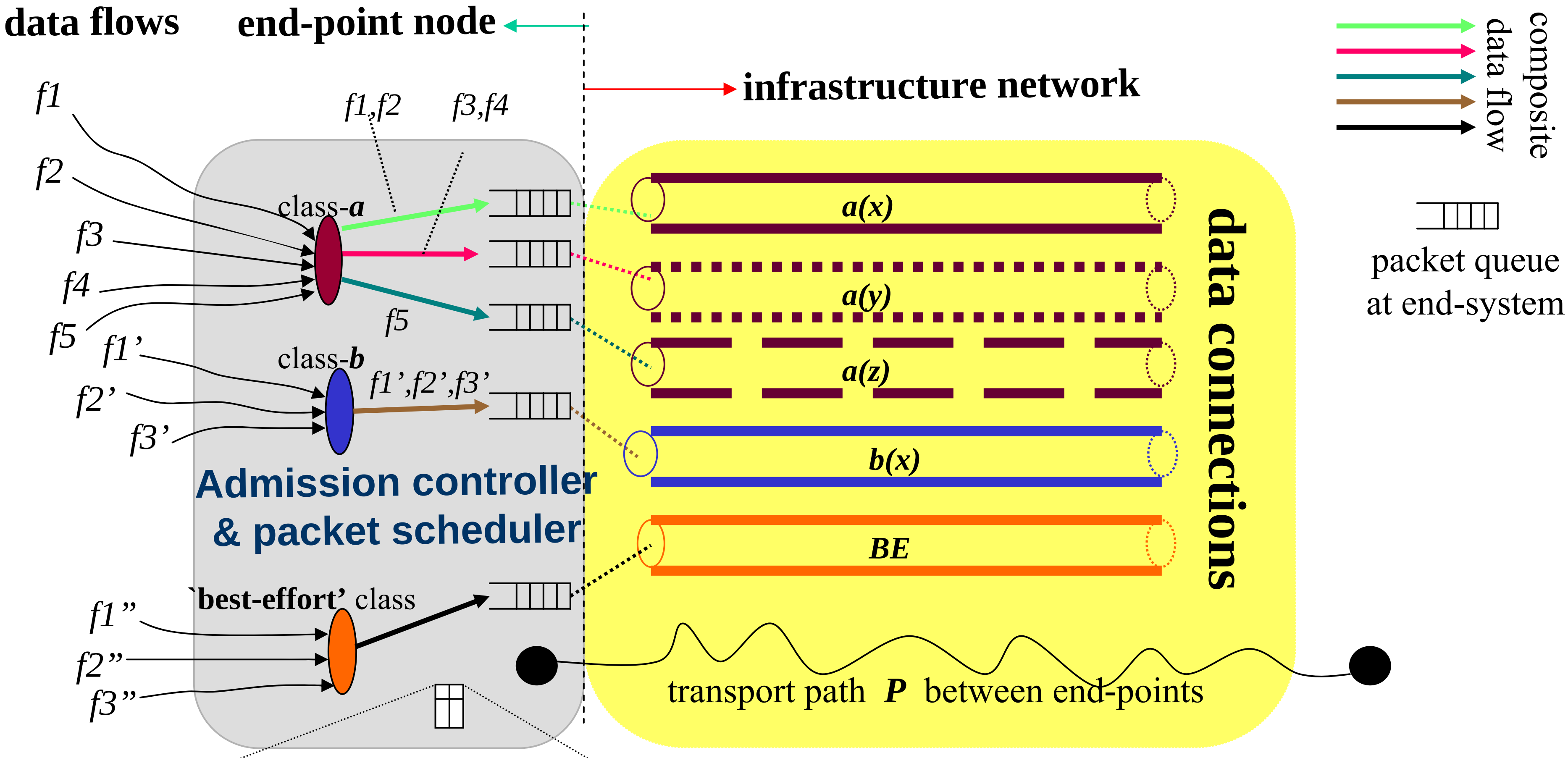
** Aggregate *closely-similar* flows for ease of estimation of combined bandwidth needs [using policy functions ***F(..)***]

** Reject new flows if the estimated (combined) bandwidth needs exceed the available capacity of ***P***

1. Two flows are closely-similar ➔ same data rate, delay, loss spec (or associated with a single application-specified label)

2. closely-similar flows fall under the same service class.
(applications determines the # of classes
➔ somewhat differs from CBQ model [S. Floyd, V. Jacobsen]**)**

Admission control (. . . contd.)



available bandwidth along path ***P***	class-***a***	🟩	2	*avail*(***a***,*x*)
		🟥	2	*avail*(***a***,*y*)
		🟦	1	*avail*(***a***,*z*)
	class-***b***	🟫	3	*avail*(***b***,*x*)

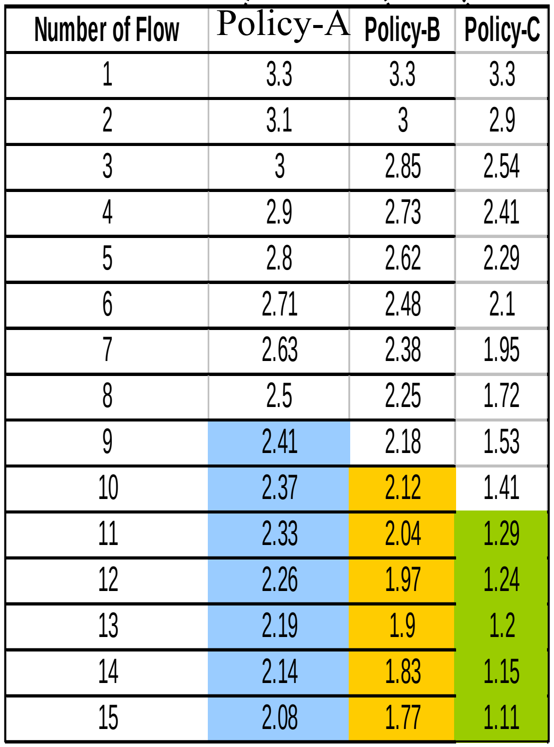
state tuple in table

(path capacity, {(service class id, flow descriptor),
{(connection id, # of flows aggregated,
available capacity in connection)}})

Is there a quantifiable benchmark of statistical multiplexing "bandwidth gains" ??

each flow is a
MPEG video
stream of
"Jurassic Park"

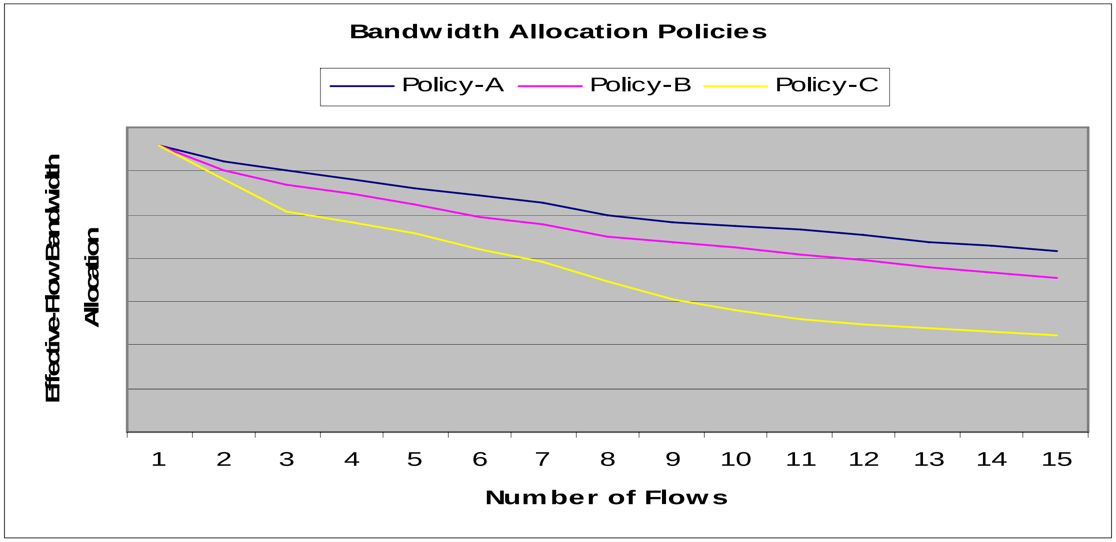
per-flow bandwidth
allocation in *mbps*
(pre-computed for study purposes)



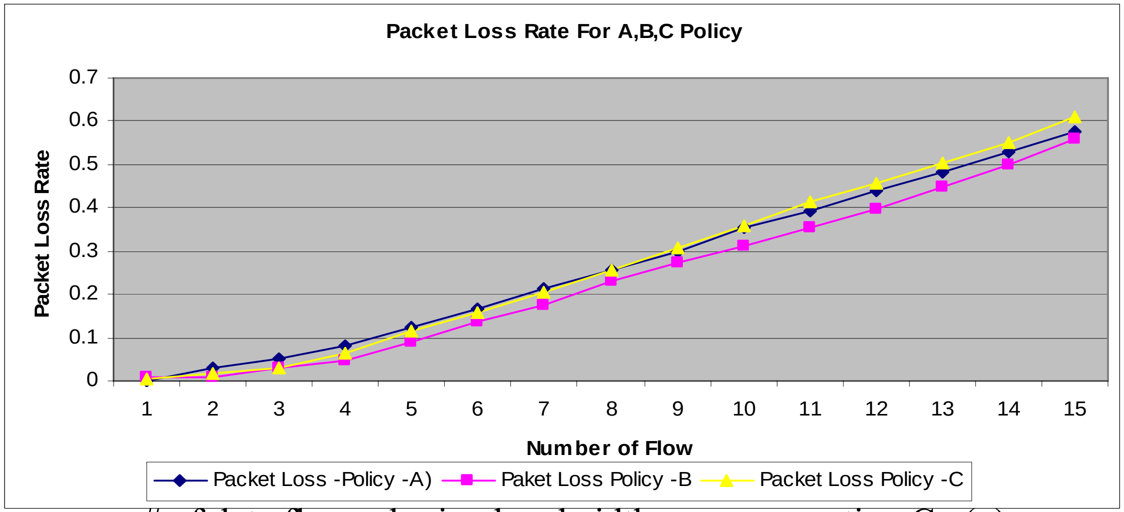
Number of Flow	Policy-A	Policy-B	Policy-C
1	3.3	3.3	3.3
2	3.1	3	2.9
3	3	2.85	2.54
4	2.9	2.73	2.41
5	2.8	2.62	2.29
6	2.71	2.48	2.1
7	2.63	2.38	1.95
8	2.5	2.25	1.72
9	2.41	2.18	1.53
10	2.37	2.12	1.41
11	2.33	2.04	1.29
12	2.26	1.97	1.24
13	2.19	1.9	1.2
14	2.14	1.83	1.15
15	2.08	1.77	1.11

tables to represent
allocation policies ***F-A, F-B, F-C***
are not known before-hand

Per-flow allocation =
total bandwidth / # of



Packet loss/delay
over connection C



# of data flows sharing bandwidth over connection C (*n*)

policy C is more aggressive than policy B;
policy B is more aggressive than policy A

Difficulties

Unfortunately, the estimator function ***F(..)*** that is needed for optimal control of bandwidth allocation ***is not known before-hand***

Reasons:

1. Fuzziness in estimating the bandwidth needs to service a data flow

   Arises due to the inability to precisely characterize the flow parameters

2. Statistical sharing of bandwidth among data flows

   Arises due to the bursty and random nature of bandwidth demands that are too complex to quantify

Issues on hand:

➔ How much bandwidth to allocate, because ***F(…)*** is not known ??

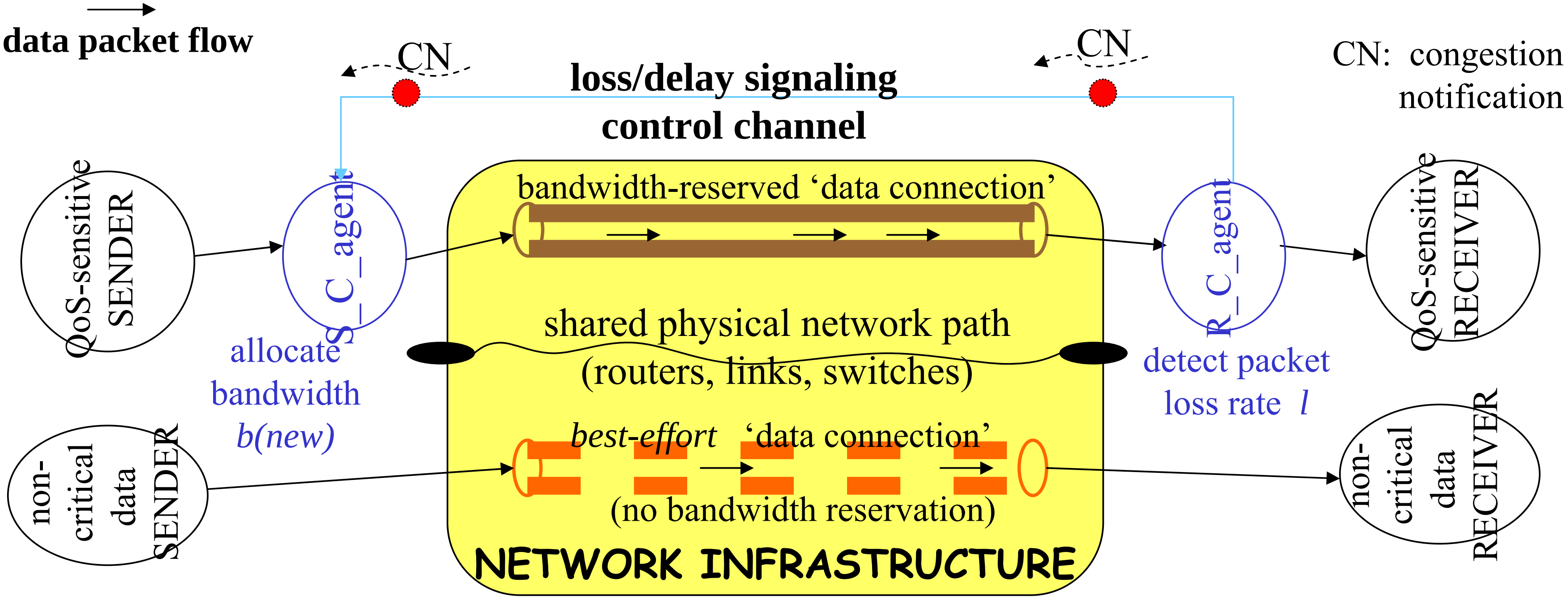
➔Resort to a "**allocate-and-see-what-happens**" approach using an iterative procedure

[on-line identification of *F(..)* is needed]

(estimator function for allocation requirements to reap statistical multiplexing gains)

Our approach to bandwidth allocations:

on-line monitoring and control



data packet flow

CN

loss/delay signaling control channel

CN

CN: congestion notification

QoS-sensitive SENDER

S_C_agent

allocate bandwidth *b(new)*

bandwidth-reserved ‘data connection’

shared physical network path (routers, links, switches)

best-effort ‘data connection’

(no bandwidth reservation)

NETWORK INFRASTRUCTURE

R_C_agent

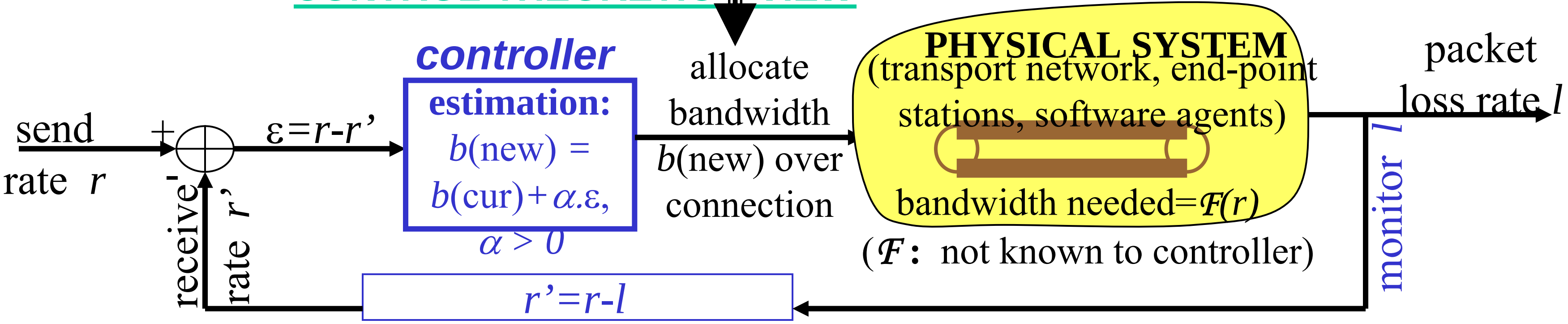
detect packet loss rate *l*

QoS-sensitive RECEIVER

non-critical data SENDER

non-critical data RECEIVER

CONTROL-THEORETIC VIEW



send rate *r*

+

-

$\varepsilon = r - r'$

controller

estimation:

$b(\text{new}) = b(\text{cur}) + \alpha.\varepsilon,$

$\alpha > 0$

allocate bandwidth *b*(new) over connection

PHYSICAL SYSTEM

(transport network, end-point stations, software agents)

bandwidth needed=$\mathcal{F}(r)$

($\mathcal{F}$: not known to controller)

packet loss rate *l*

monitor *l*

$r' = r - l$

receive rate *r’*

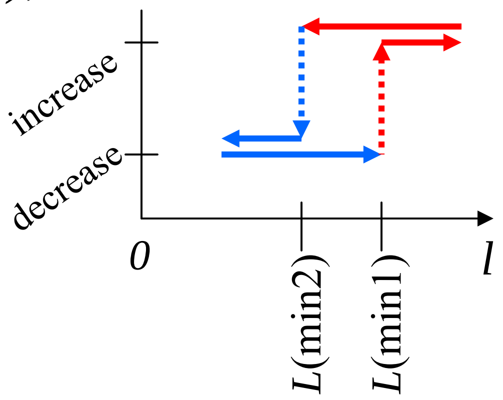
Iterative procedure for bandwidth adjustment

Multiplicative Increase additive decrease (MIAD)

In each ‘loss reporting’ interval

$$b(new) = b(cur) + \alpha . l \quad \text{when} \quad l > L(min1),$$
$$\text{where } \alpha > 0$$

$$b(new) = b(cur) - \beta \quad \text{when} \quad l < L(min2),$$
$$\text{where } \beta > 0$$



l: observed packet loss rate

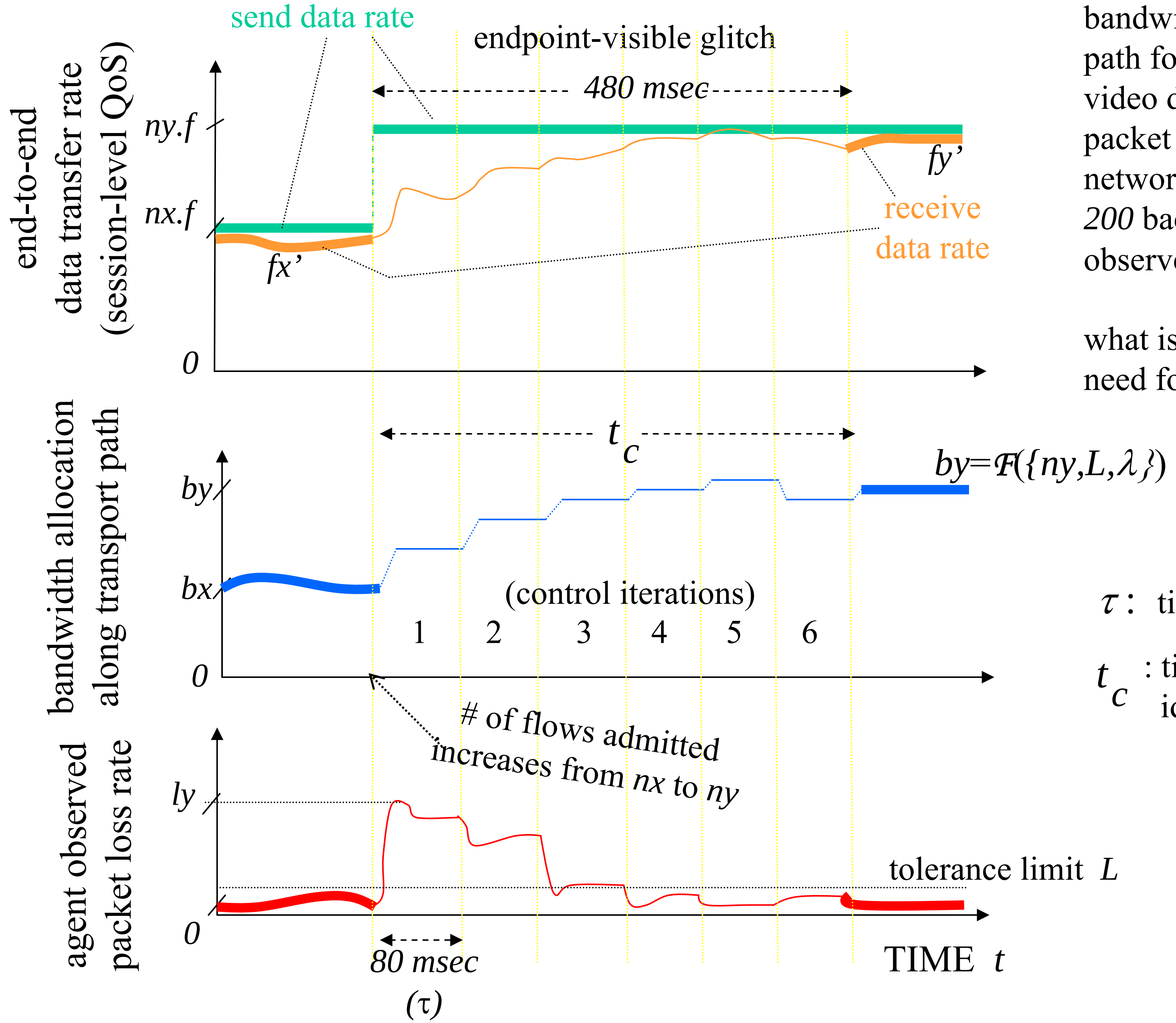
L(min1), *L*(min2): Acceptable loss thresholds

[$L(min2) < L(min1)$, to avoid ping-pong effect]

^^ Each execution of this procedure constitutes a “**control iteration**”

^^ A sequence iterations that lead to a steady-state in bandwidth allocation (when the flow specs changes or a set of new flows are admitted) constitutes a “**control round**”

A sample scenario of iterative allocation of bandwidth



bandwidth allocated along path for *nx =10* is *25mbps;*
video data rate *f = 3 mbps;*
packet size=*10000 bits;*
network capacity=*100mbps;*
200 back-to-back packets observed → which impacts *τ*

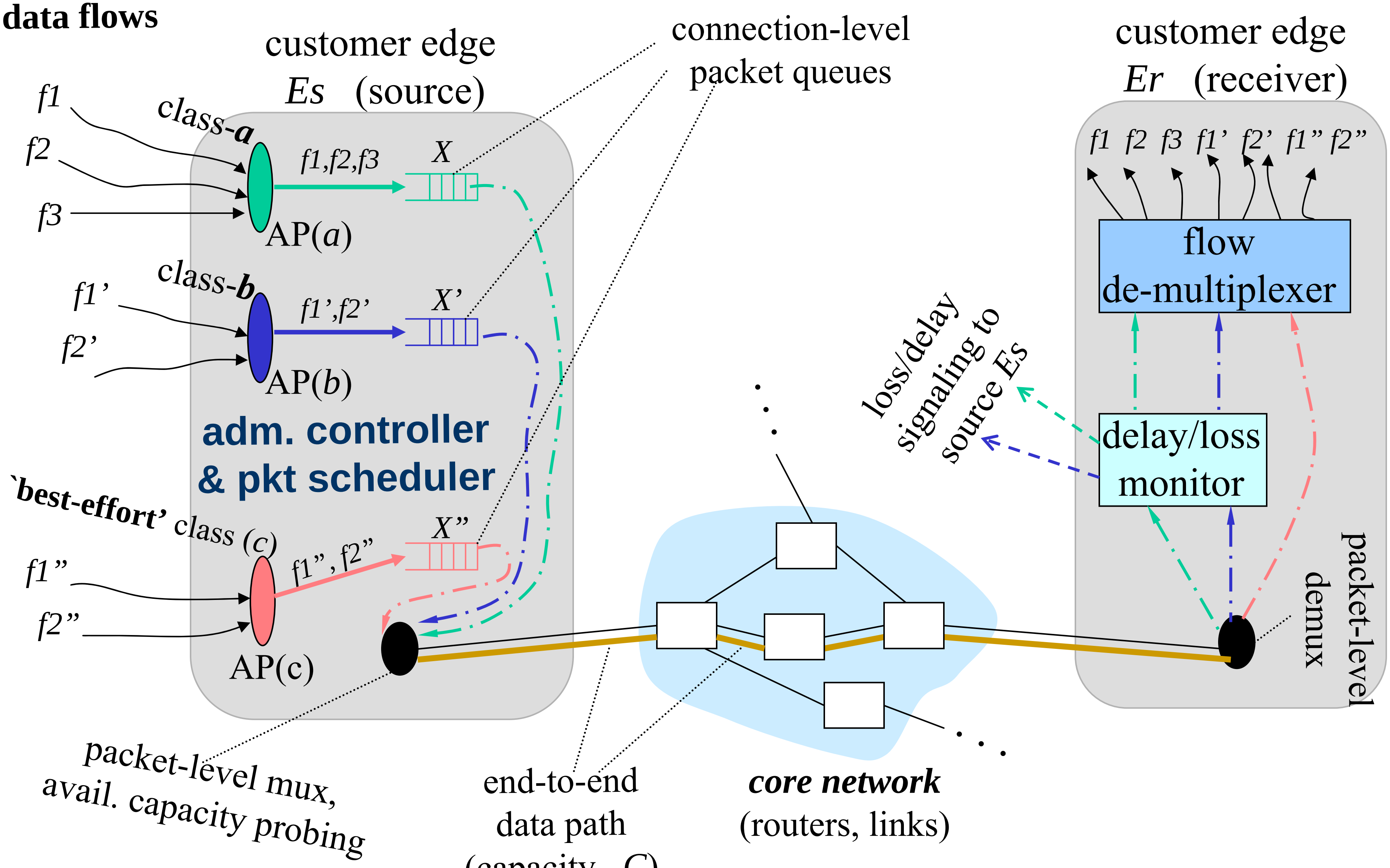
what is the bandwidth need for, say, *ny = 16 ??*

λ: data rate

τ : time constant of system

t_c : time taken for on-line identification of ***F(..)***

Architecture employed for “monitor-and-control”



$$[\lambda_X + \lambda_{X'}] < C$$

AP(*y*): aggregation point
for connection *y*
(assigns MPLS label)

λ: connection-level traffic flow rate

packet departures/arrivals
of respective connections

Experimental studies by simulation

^^ Employed discrete-event simulation (at packet-level)

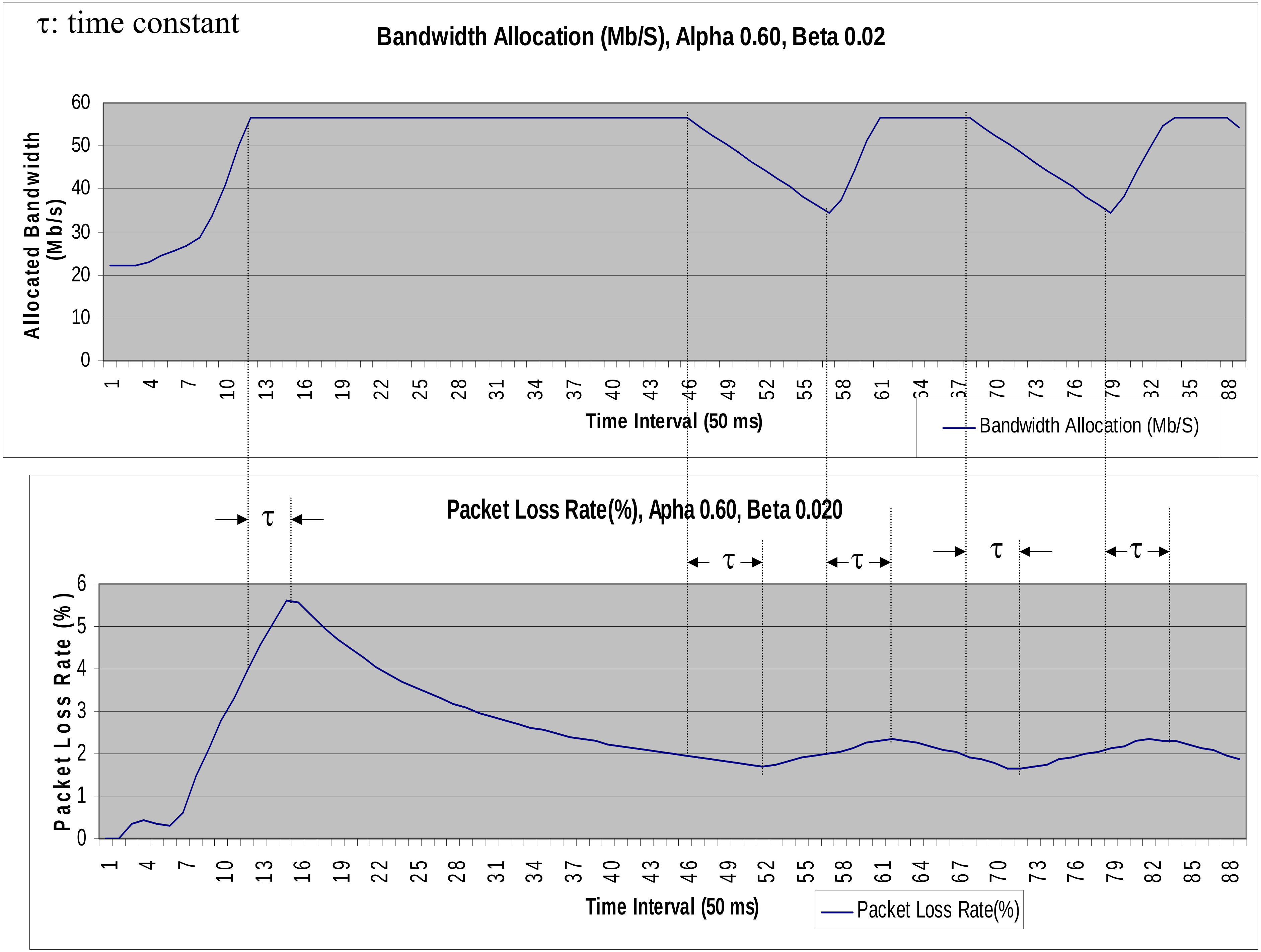
^^ Generated ‘packet arrival’ events based on video traces [Jurassic Park]

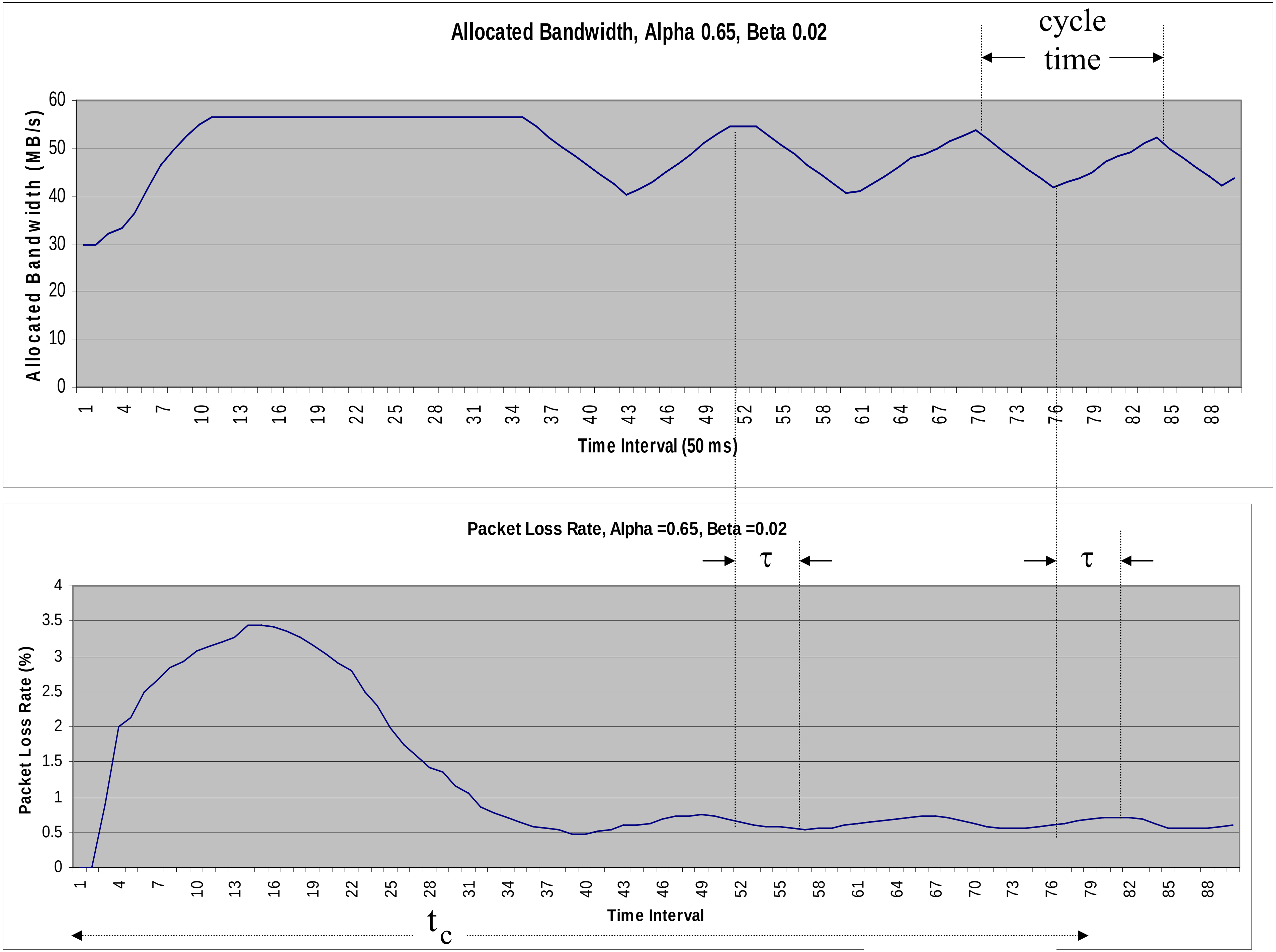
^^ 3 ‘data connections’ plus a ‘best-effort connection’ are simulated

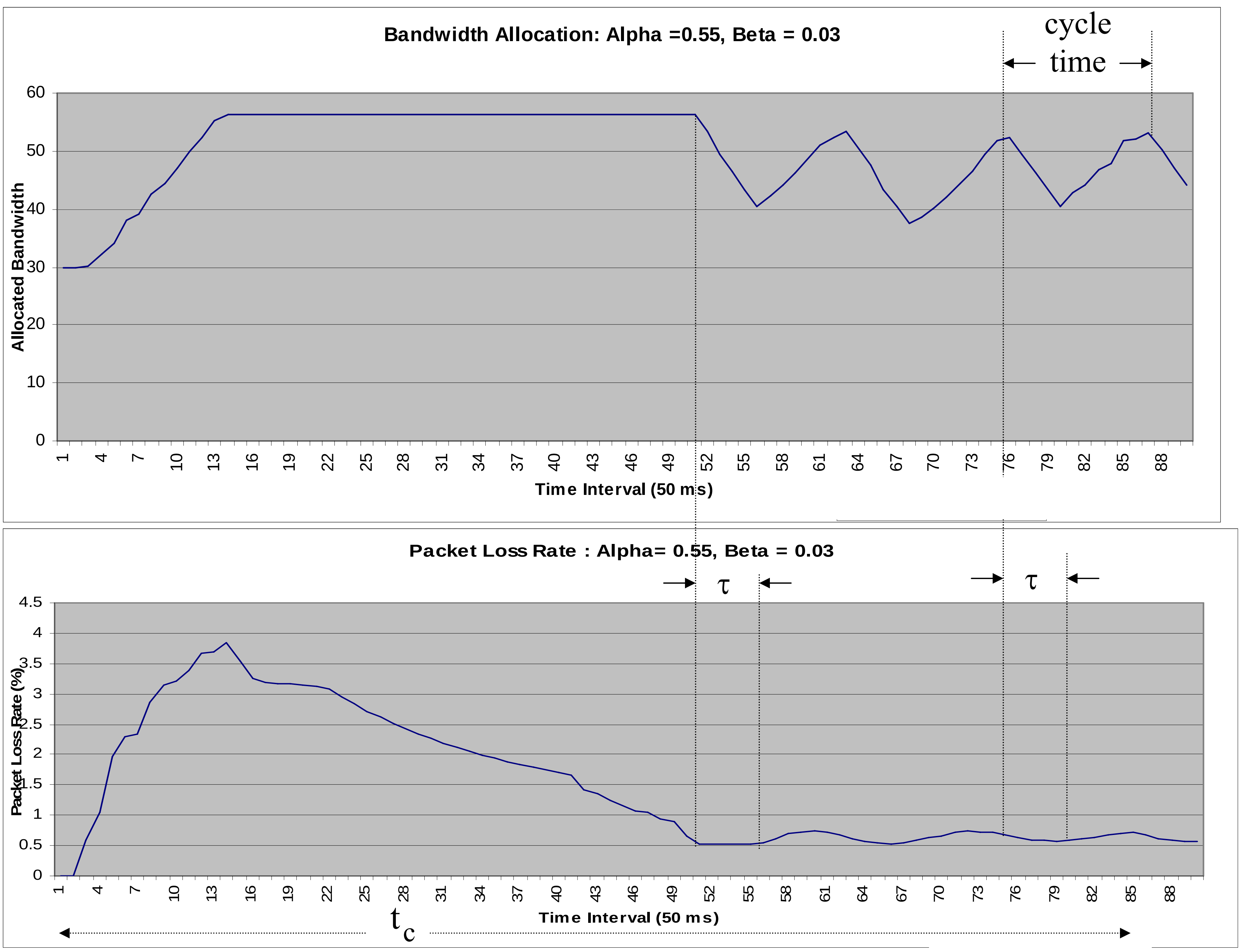
(each connection has 10 flows, each flow being a video stream)

^^ 100 *mbps* single-hop physical link in core network (packets sent through an access router)

^^ Packet-loss monitoring by sequence numbers (assigned along the ‘management’ plane by monitor agents)


τ: time constant
Bandwidth Allocation (Mb/S), Alpha 0.60, Beta 0.02
Allocated Bandwidth (Mb/s)
60
50
40
30
20
10
0
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55 58 61 64 67 70 73 76 79 82 85 88
Time Interval (50 ms)
Bandwidth Allocation (Mb/S)
Packet Loss Rate(%), Apha 0.60, Beta 0.020
τ
τ
τ
τ
τ
Packet Loss Rate (%)
6
5
4
3
2
1
0
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55 58 61 64 67 70 73 76 79 82 85 88
Time Interval (50 ms)
Packet Loss Rate(%)


Allocated Bandwidth, Alpha 0.65, Beta 0.02
cycle time
Allocated Bandwidth (MB/s)
0
10
20
30
40
50
60
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55 58 61 64 67 70 73 76 79 82 85 88
Time Interval (50 ms)
Packet Loss Rate, Alpha =0.65, Beta =0.02
τ
τ
Packet Loss Rate (%)
0
0.5
1
1.5
2
2.5
3
3.5
4
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55 58 61 64 67 70 73 76 79 82 85 88
Time Interval
t_c


Bandwidth Allocation: Alpha =0.55, Beta = 0.03
cycle time
Allocated Bandwidth
60
50
40
30
20
10
0
1
4
7
10
13
16
19
22
25
28
31
34
37
40
43
46
49
52
55
58
61
64
67
70
73
76
79
82
85
88
Time Interval (50 ms)
Packet Loss Rate : Alpha= 0.55, Beta = 0.03
τ
τ
Packet Loss Rate (%)
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
1
4
7
10
13
16
19
22
25
28
31
34
37
40
43
46
49
52
55
58
61
64
67
70
73
76
79
82
85
88
Time Interval (50 ms)
t_c

Observations on experimental results

^^ In our simulation we multiplexed 3 connections: #1 has 10 flows multiplexed, #2 has 8 flows, and #3 has 6 flows; (combined) bandwidth allocation is: 30 mbps, 25 mbps and 20 mbps respectively.

^^ To observe the multiplexing effects, we then increased the # flows in connection 1 from 10 to 16 (by injecting 6 new flows) .

^^ The result shows it takes about 1.2 second for the bandwidth adaptation procedure to complete successfully.

^^ $\tau \approx 225$ *msec* (time it takes to loss rate to decrease / increase after an additional bandwidth has been allocated /de-allocated).

^^ After injecting 6 new flows, we achieve optimal bandwidth of 45 mbps (which is 2.8 mbps per flow allocation

➔ savings of 3.0 mbps across 16 flows, i.e., about 0.2 mbps savings per flow) .

Summary

Bandwidth allocation problem is intractable because it is difficult to quantify as to what the right level of statistical multiplexing gains should be while not violating the end-user's QoS

(service provider's revenue goals conflict with end-user's QoS utility)

Solution:

"***online monitor-and-control***" based iterative approach for optimal bandwidth allocations in end-to-end connections to maximize service provider's revenues

➔ Existing works on "hose" model of VPNs and "class-based queuing" disciplines provide architectural reference models for our approach

Future Plans

1. Employ “optimal control theory” principles to study the stability and convergence properties of on-line bandwidth allocations

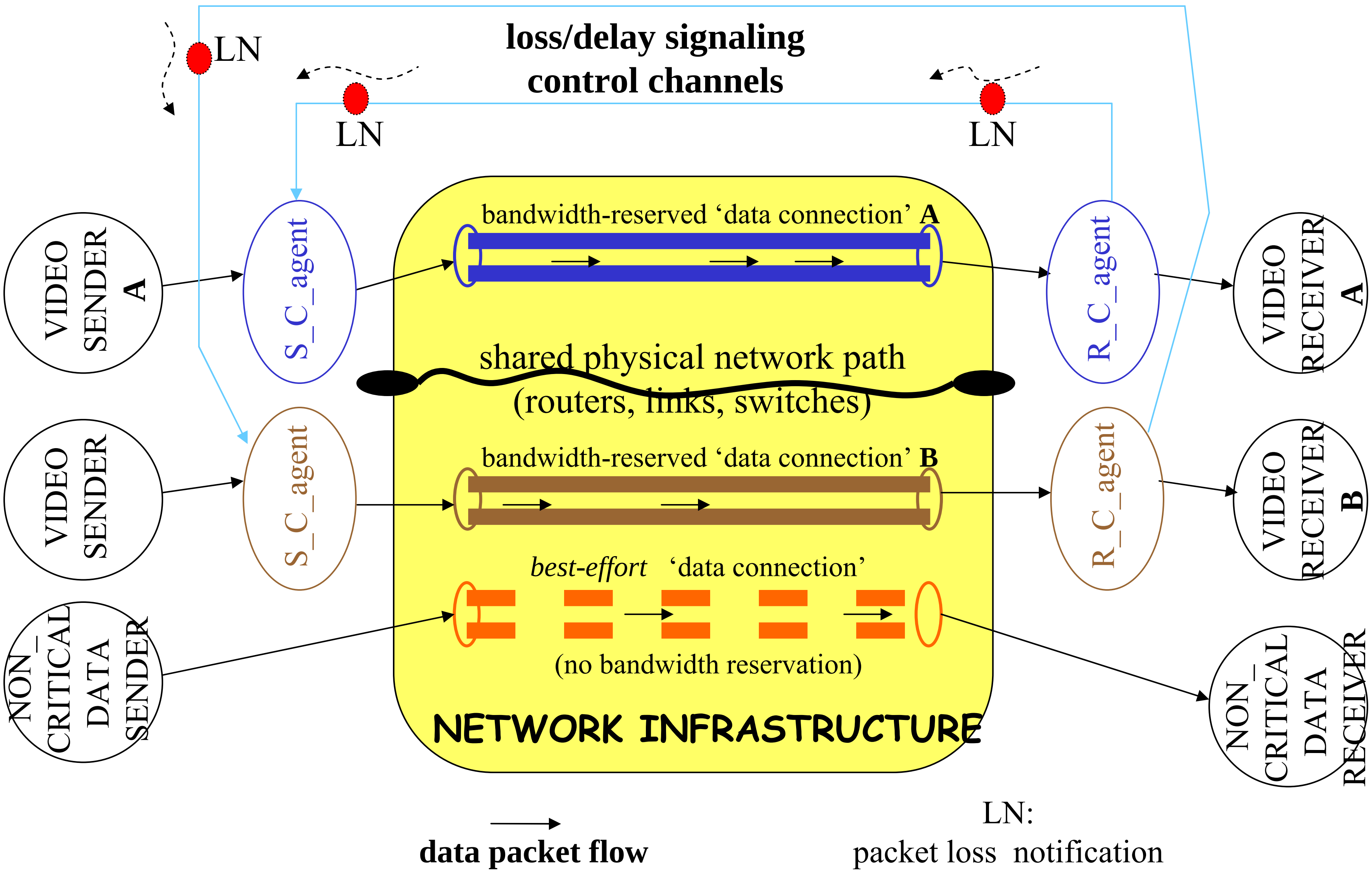
   (treat QoS regulation and revenue maximization as instances of “constrained optimization” problems

   ➔ heuristics-based search procedures)

2. Study cross-coupling effects across multiple connections because of the sharing of physical infrastructure between connections

   (QoS isolation needs to be achieved between connections)

Cross-coupling between data connections



Question: What is the effect of bandwidth change in connection A, on connection B (due to the sharing physical infrastructure) ?

Cross-coupling (… contd.)