Evaluation of End-node Based Rate Allocation Schemes for Lambda Networks

Xinran (Ryan) Wu* and Andrew A. Chien* **

Dept of Computer Science and Engineering, Univ. of California, San Diego*

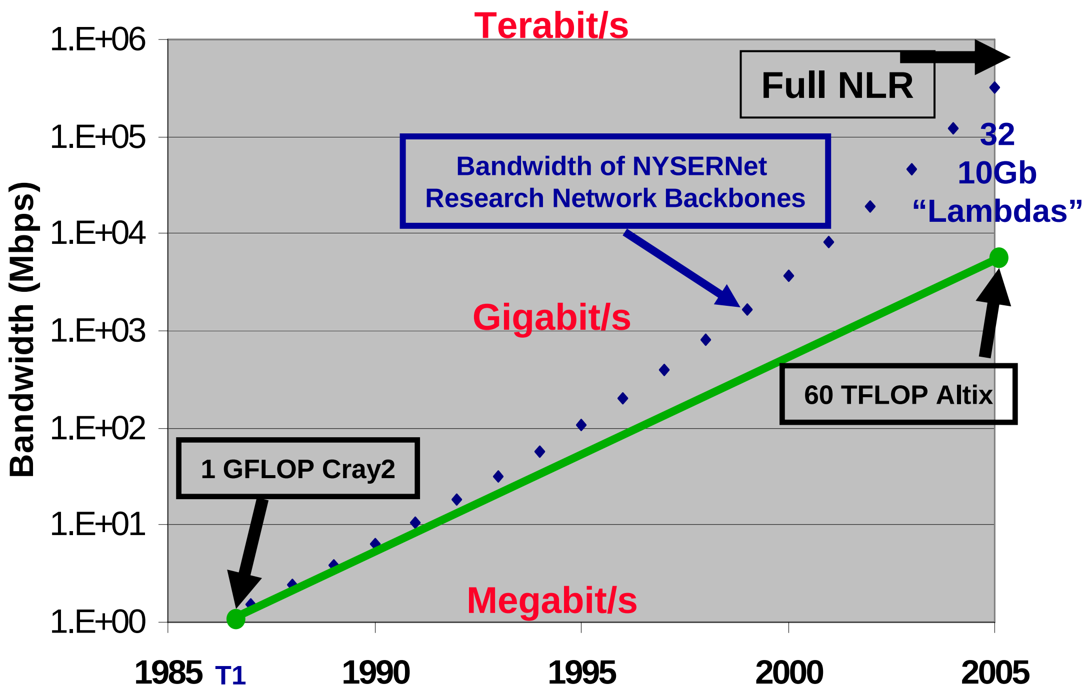
Intel Research**

PFLDnet 2006

Feb 3, 2006

Optical WAN Research Bandwidth Has Grown Much Faster than Supercomputer Speed!

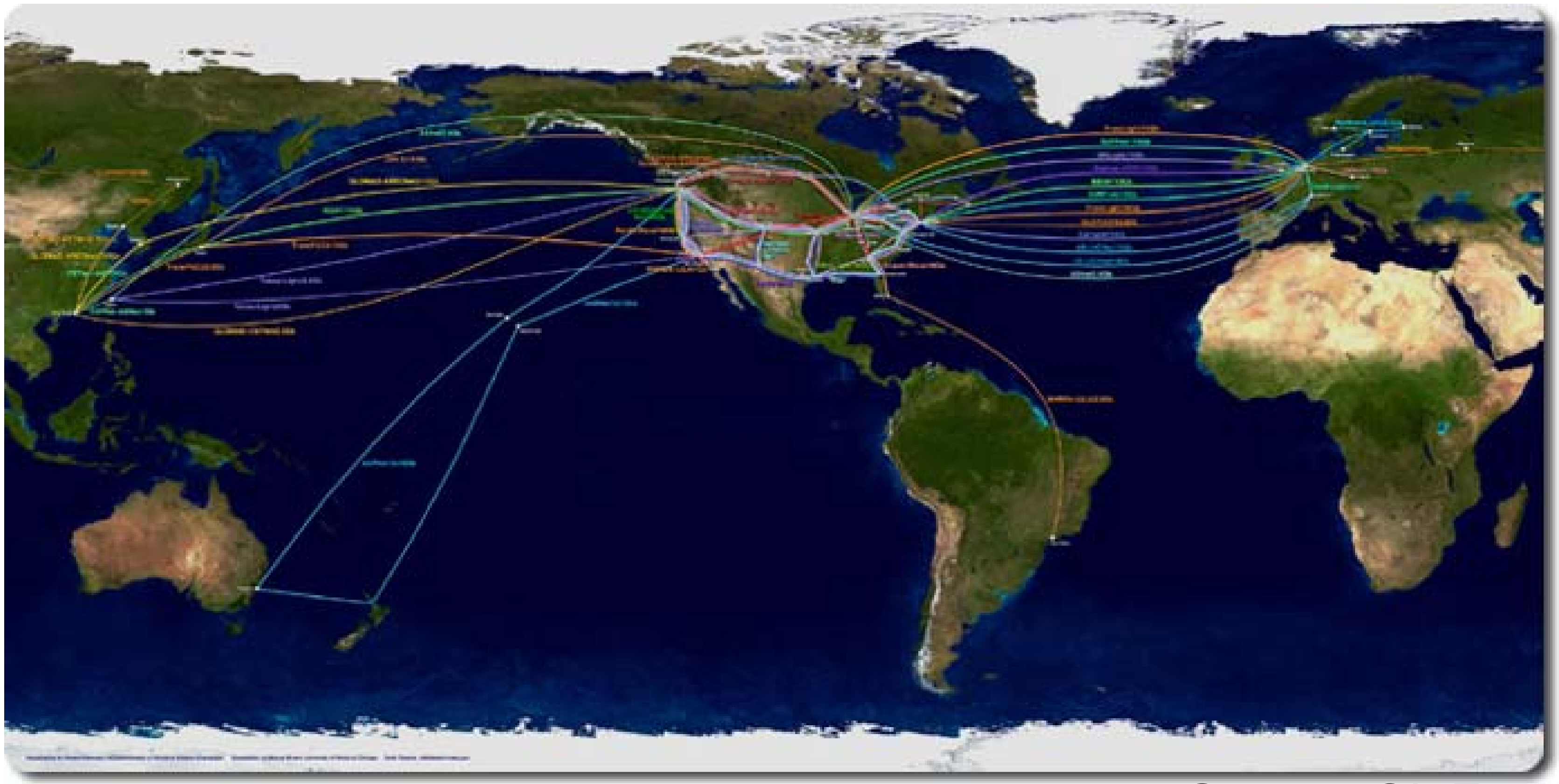
- **DWDM enables a single fiber to carry 100's of lambdas (10 or 40 Gbps each)**
- **Plentiful network bandwidth**
  - **Network speed >> Computing & I/O speed**
  - **Inversed World: Not all applications have infinite demands**



Source: Timothy Lance, President, NYSERNet

Lambda Networks Are Widely Deployed!

- The OptIPuter
- Global Lambda Integrated Facility
- National Lambda Rail
- Netherlight
- Ultralight
- CANARIE, Canada
- DataTAG
- Teragrid
- UKLight
- … …

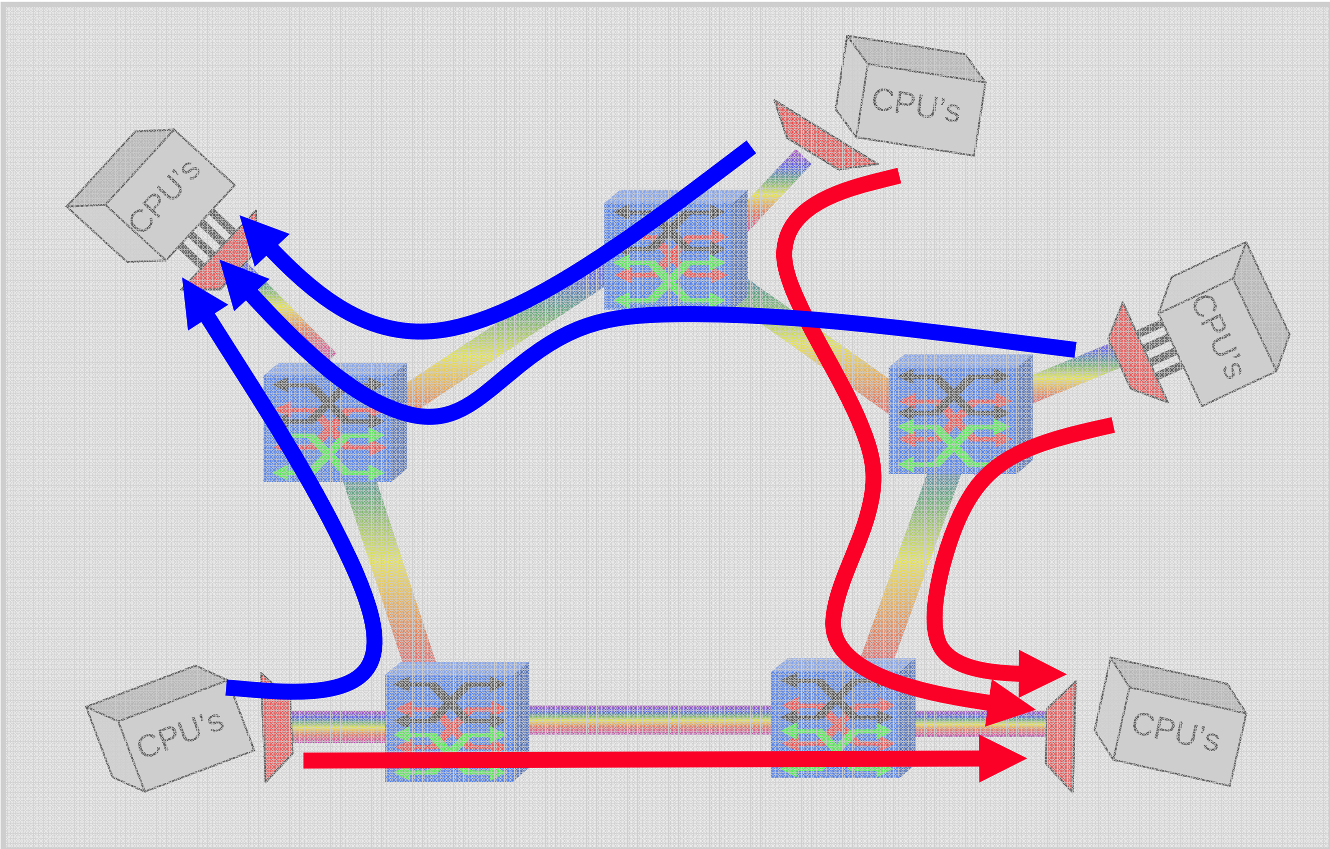


Source: GLIF

OptIPuter

Optical Network Cores Shift Contention to Network Edge

- **Lambda Networks: dedicated optical connections providing plentiful core bandwidth**
- **Driving applications access many high data rate sources**
  - **Multiple multipoint-to-point communication**
  - **Sharing bottleneck moves to the endpoints**

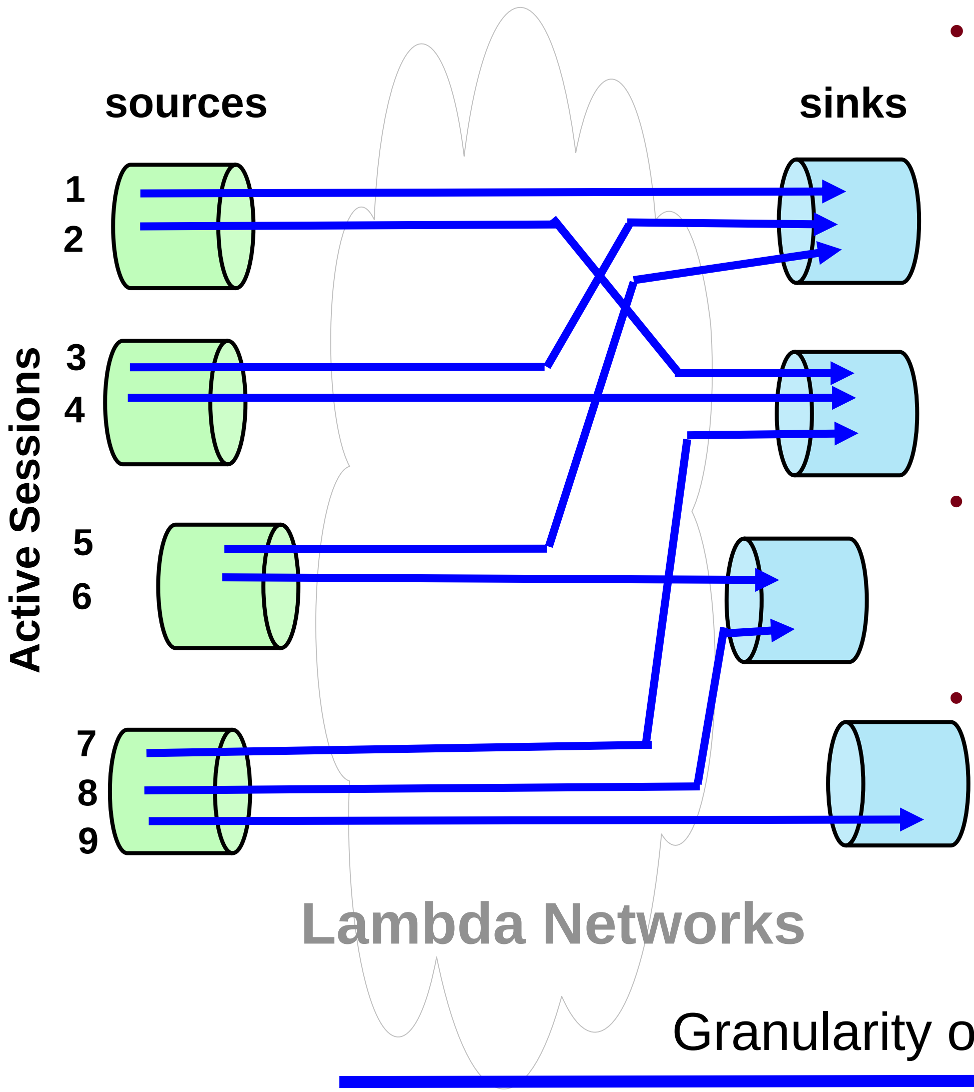


OptIPuter

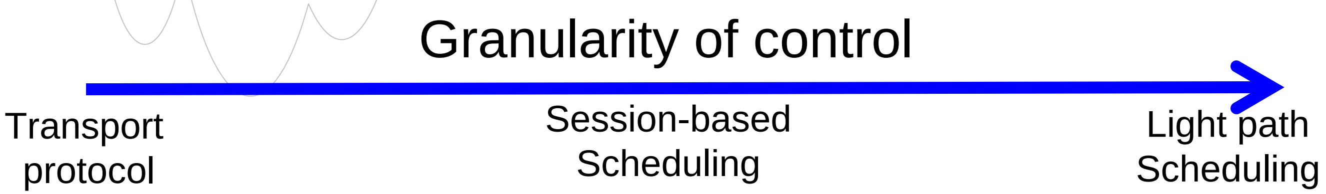
Outline

- **Problem Formulation**
- **Current Approaches**
  - **Extending switch-based schemes to end points?**
  - **Extending router-based schemes to end points?**
  - **New rate allocation schemes?**
- **Evaluation Results**
- **Discussion and Conclusion**

The Rate Allocation Problem in Lambda Networks

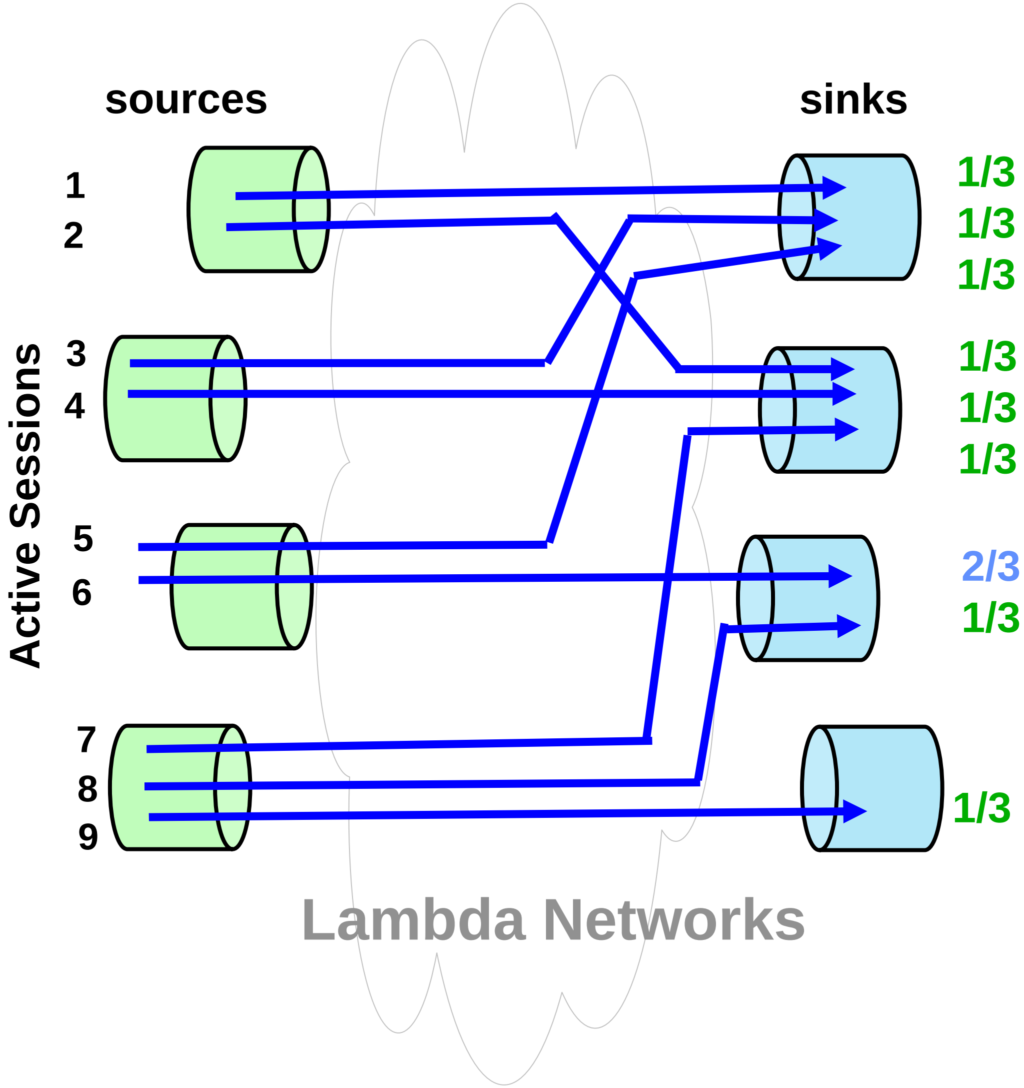


- **Assumptions**
  - **No need to model network internals**
  - **Each node has explicit knowledge regarding its capacity and associated sessions**
  - **Explicit rate feedback between sources and sinks is feasible**
  - **Each session has a desired rate, unknown to its sources and sinks**
- **The rate allocation problem:**
  - **How to *efficiently* and *fairly* share the capacity of each source and sink among active sessions?**
- **The challenges**
  - **Congestion at end nodes due to high bandwidth and long delay**
  - **Fair to sessions with various RTT, demands, etc.**



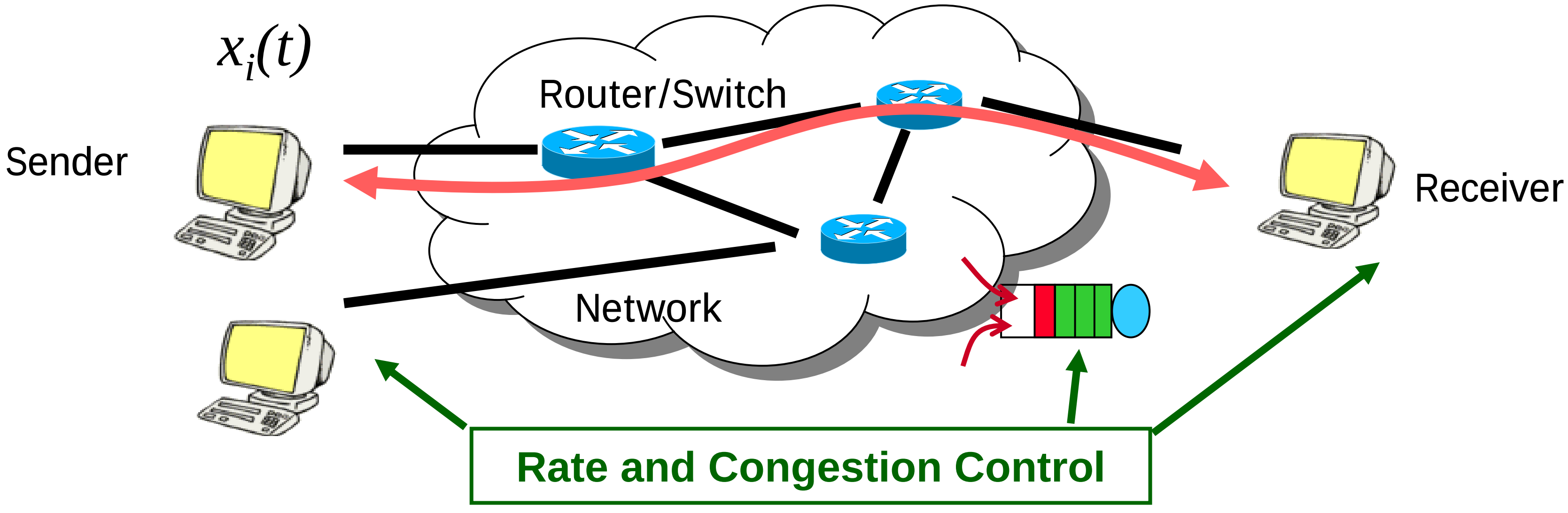
OptIPuter

Solution Criteria and Metrics



- Feasibility
- Efficiency
  - High link utilization
  - Avoidance of severe congestion
  - Quick reaction to flow dynamics
- Fairness
  - Max-min fair among sessions
- Stability and Convergence

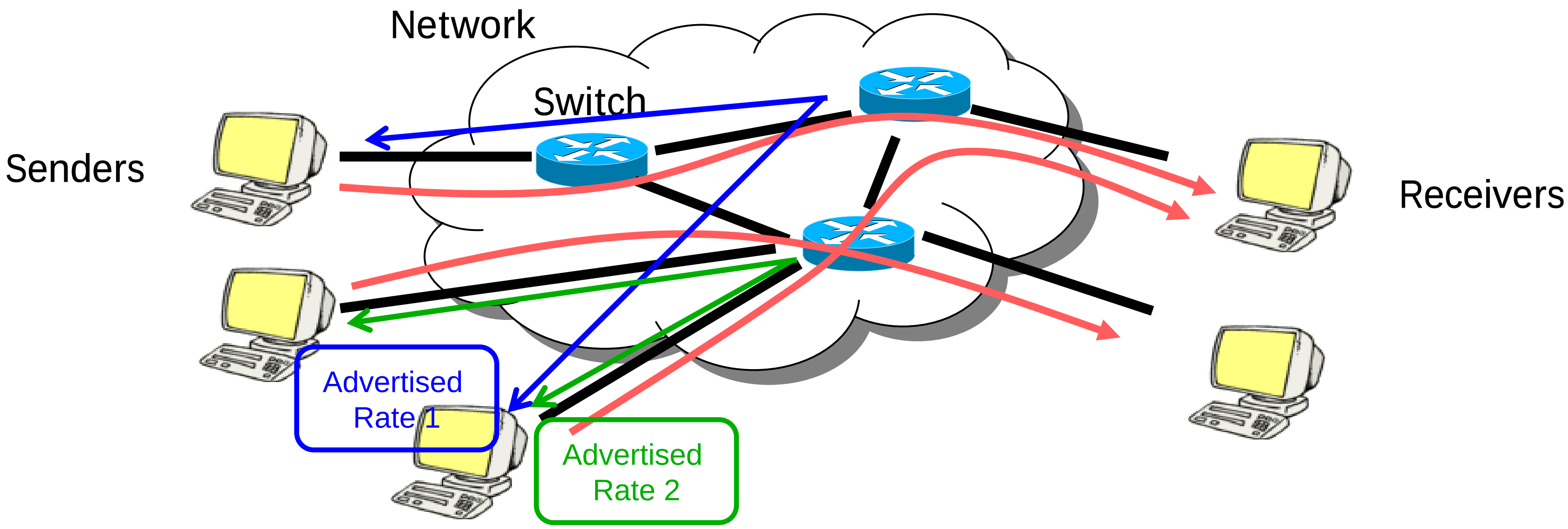
Approaches: Overview



- Session-based schemes (e.g. TCP variants)
- **Extending router/switch based rate allocation schemes to end nodes**
- **New end-node based rate allocation and sharing scheme**

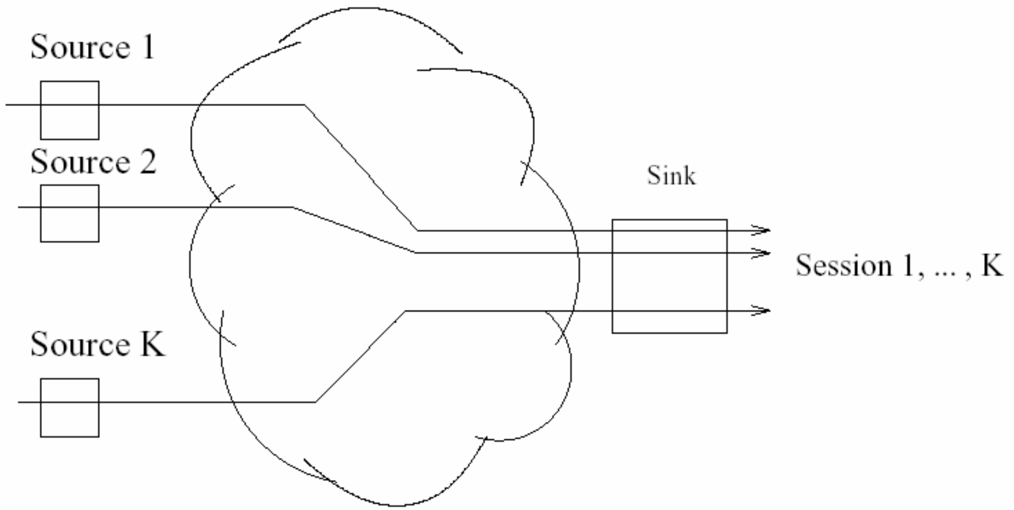
OptIPuter

Switch-based Schemes: Max-min Fair Sharing for ABR Traffic



- *Consistent Marking* Schemes: [Charny93] [Hou99]
- Flows are divided in two groups.
  - Flows that are bottlenecked elsewhere -- Mark
  - Flows that are bottlenecked here – Sharing the remaining capacity

Most Switch-based Schemes do not work well in lambda networks

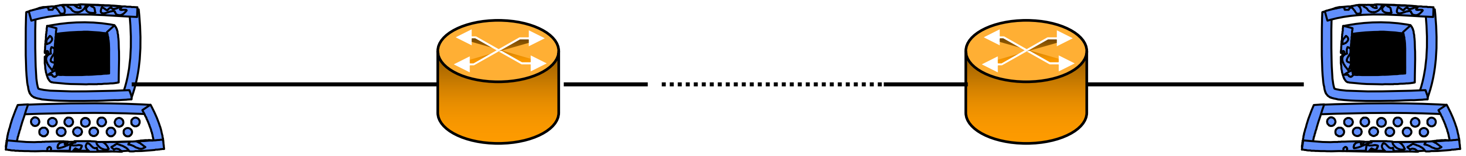


Session	Session demand	Eq. rate	adv. rate
1	1	0.7	0.7
2	0.1	0.1	0.7
3	0.1	0.1	0.7
4	0.1	0.1	0.7

- **Example:**
  - **3 out of 4 sessions with limited demands;**
  - **The same advertising rate 0.7 is fed back**
  - **Potential congestion at the receiver side when three 'thin' sessions increase their demands**
- **Same explicit rate feedback does not work in highspeed environment**

Router-assisted: XCP [Katabi, et. al. 2002]

- **The router explicitly allocate its bandwidth to each flows**
- **Decouple congestion control with fairness control**



Congestion Controller

Goal: Matches input traffic to link capacity & drains the queue

Looks at aggregate traffic & queue

Algorithm:

MIMD on Aggregate traffic changes (Δ)

Δ ~ Spare Bandwidth *Spare*

Δ ~ - Queue Size *Queue*

So, $\Delta = \alpha\, d_{avg}\, Spare - \beta\, Queue$

Fairness Controller

Goal: Divides Δ between flows to converge to fairness

Algorithm: (**AIMD** for each flow)

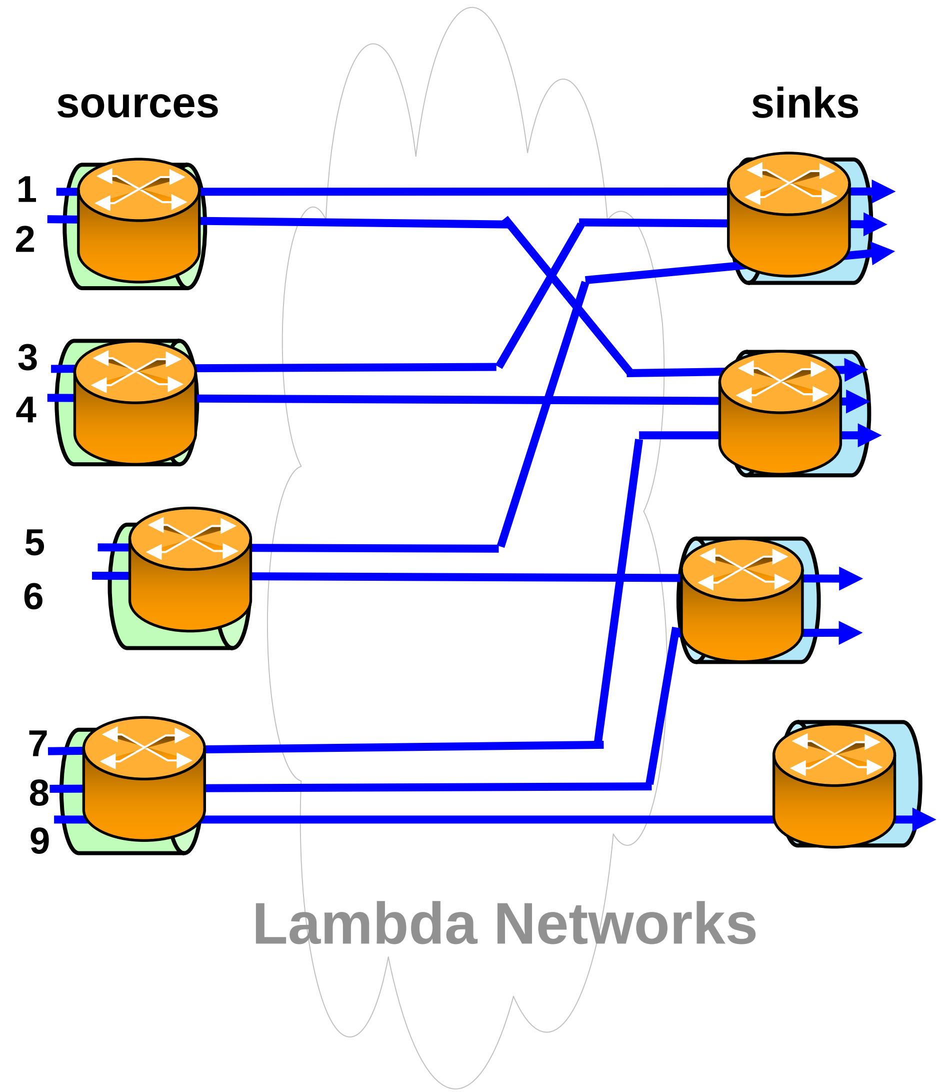
If Δ > 0 ⇒ Divide Δ equally between flows

If Δ < 0 ⇒ Divide Δ between flows proportionally to their current rates

(Source: Dina Katabi , MIT)

OptIPuter

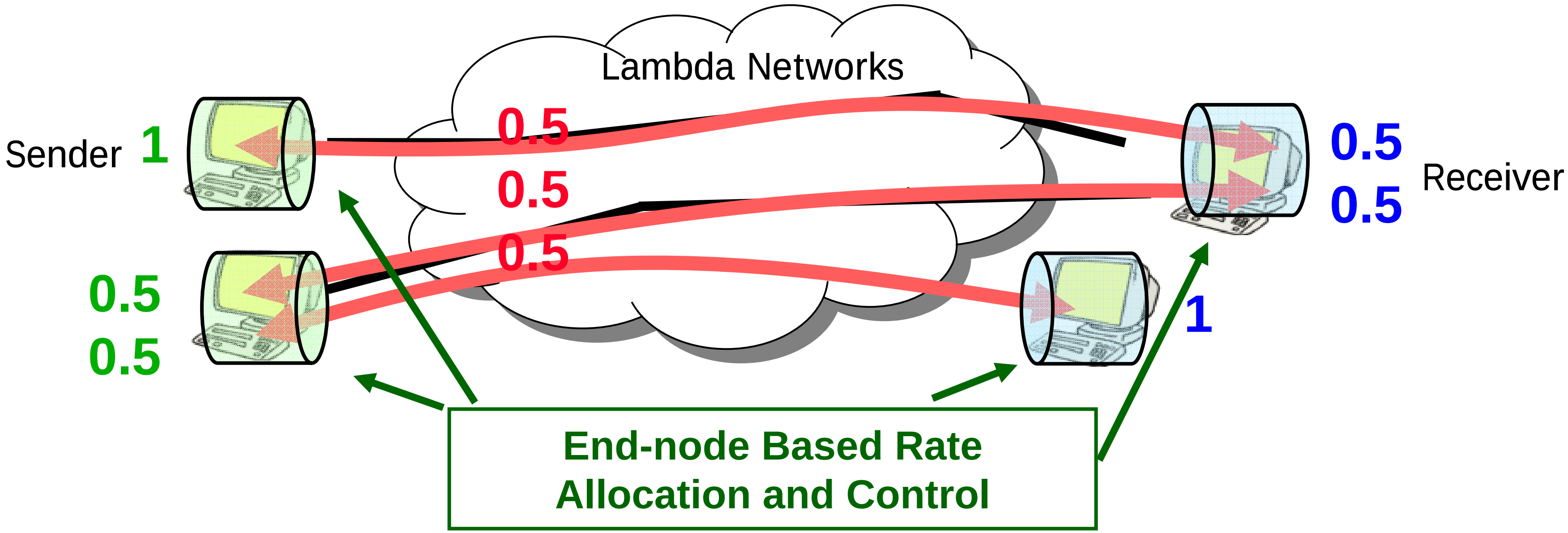
endpointXCP: Running XCP on End Nodes



- Let each end node function as an XCP router
- Run the same XCP algorithm
- = A networked cased for XCP

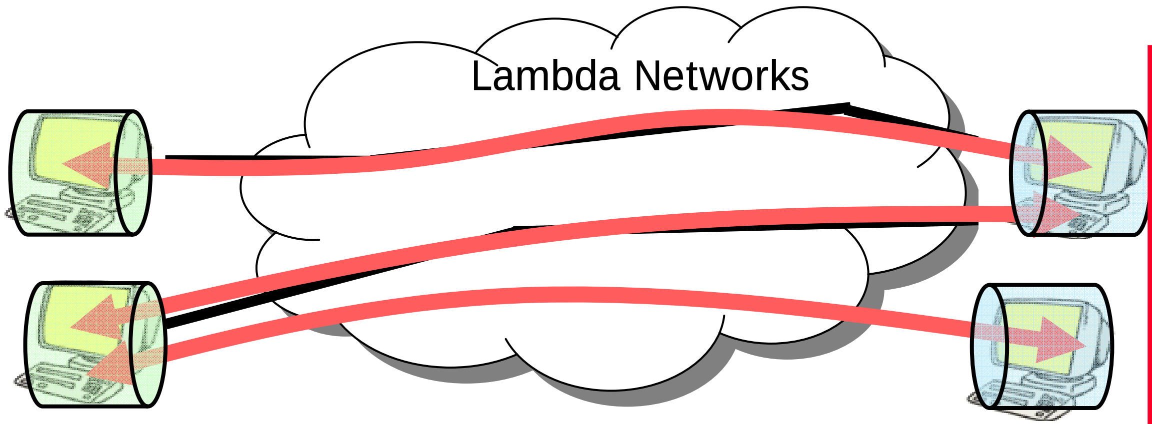
OptIPuter

End-node Based Rate Allocation and Control (GTP)



- **Approach:**
  - **Each source and sink *approximates max-min rate allocation* and feed back different expected rate for different sessions**
  - **The new session rate is the minimum among the expected rates at source, sink, and its desired rate.**

Proposed Approach (Overview)



Notions

- C^v capacity of end node v
- $x_k(t)$ rate of session k at time t
- D_k: RTT of session k
- M_k: demand of session k
- $\hat{x}_k^r$ expected rate at receiver
- $\hat{x}_k^s$ expected rate at sender

- **Each end node has local information on C^v, N, $X(t) = (x_1(t), x_2(t), \ldots x_N(t))$**
- **Each end node asynchronously conduct rate allocation**

$$\hat{x}^r(t+1) = g(x(t), C^v)$$

- **New 'assigned rate':**

$$\min(\hat{x}_k^s(t), \hat{x}_k^r(t))$$

- **Real rate update:**

$$x_k(t+1) = \min(\hat{x}_k^s(t), \hat{x}_k^r(t), M_k)$$

Proposed Approach: A Close-up View

- **Each end node has local information on C^v, N, $X(t) = (x_1(t), x_2(t), \ldots x_N(t))$**

- **Start with the one with lowest rate**
  - **Higher priority for low rate sessions**
  - **Calculates the session target rate x_f:**

$$x_f = \frac{\text{remaining bandwidth}}{\text{\# of unallocated flows}}$$

- **Using rate adaptation to achieve a smooth transition**

$$\hat{x}_k^r(t+1) = x_k(t) + \alpha(x_f - x_k(t))$$

Example

- Three sources, one sink, and three sessions;
- Sink node capacity: 100

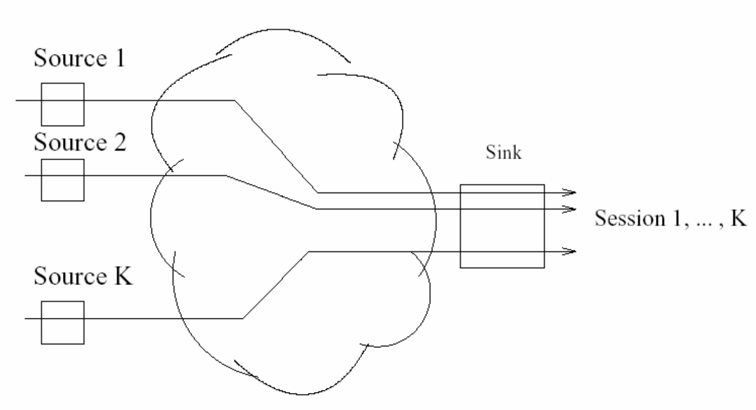
X_t	(10,	30,	50)
X_f:	33	45	60
X_{t+1}	14.6	33	52
	.	.	.
	33	33	33

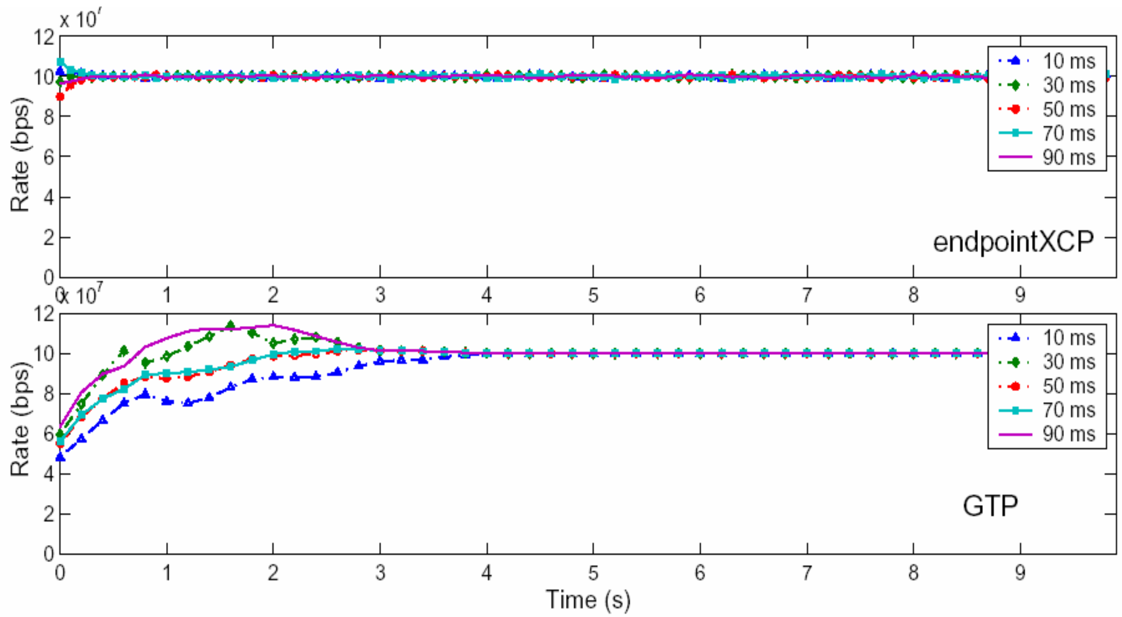
1. Sessions with smaller rates are given higher priority to be considered;
2. Adjust sessions with higher rates to fully utilize the capacity

Comparison Studies: endpointXCP and GTP

- **NS simulations**
- **Multipoint-to-point and multipoint-to-multipoint (networked) scenarios**
- **Metrics**
  - **Converge to max-min fair?**
  - **Convergence speed?**

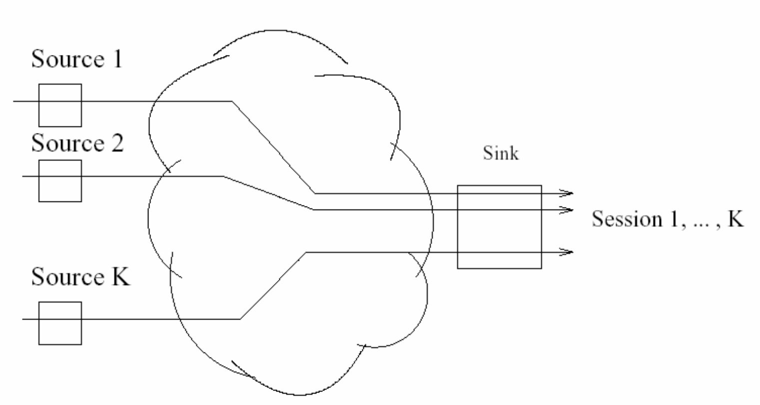
Comparison Studies: the 5-to-1 Case (1)

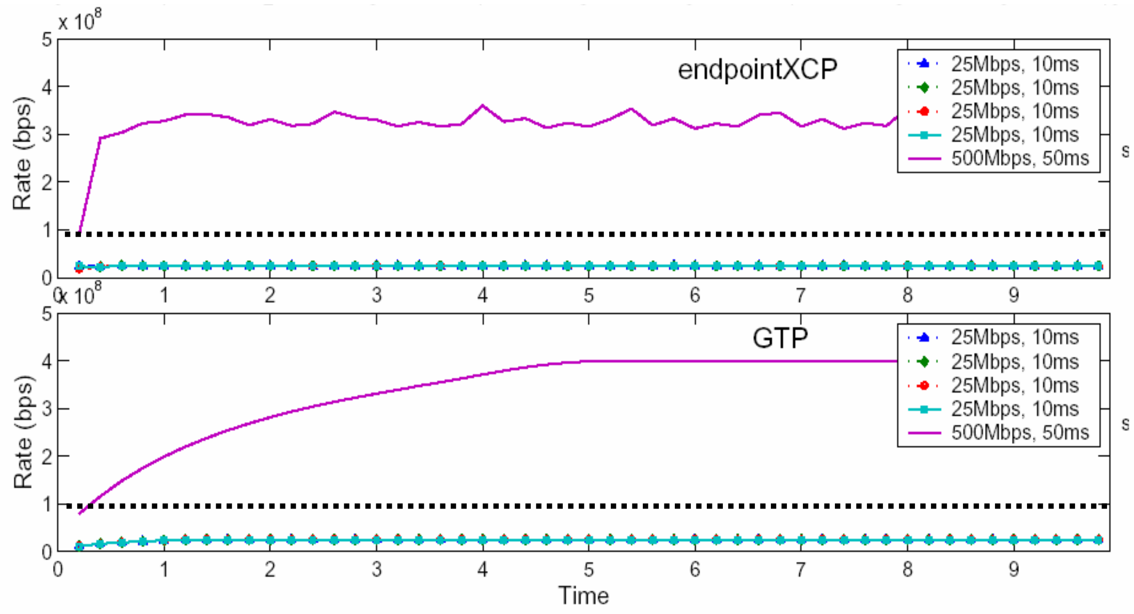




- **Five sessions with various RTT (10 – 50ms); sink capacity is 500Mbps**
- **Both endpoint XCP and GTP lead to fair sharing of the sink capacity across sessions**
- **endpoint XCP quickly converges – it's adaptation parameter is 0.22 while 0.1 is used for GTP.**

Comparison Studies: the 5-to-1 Case (2)

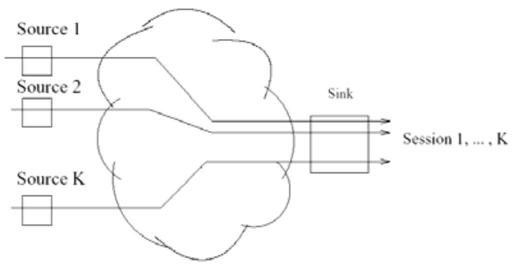


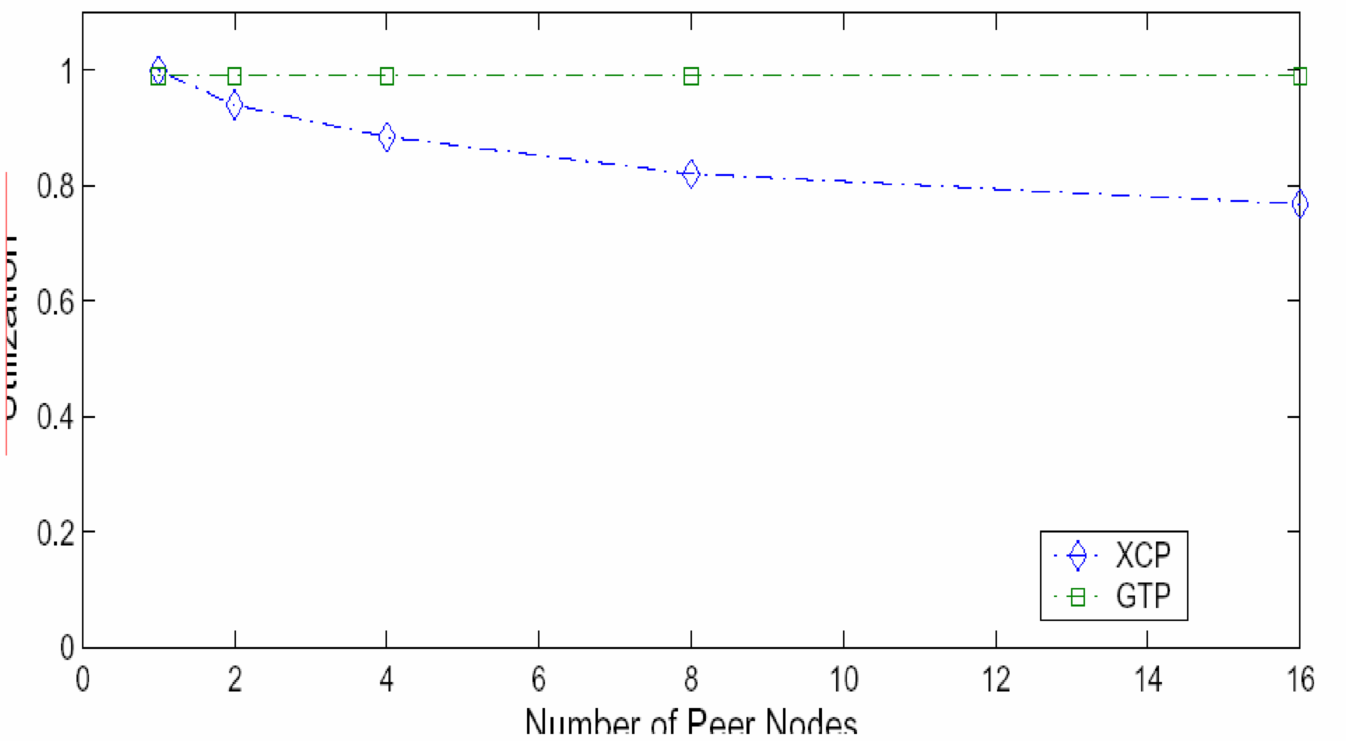


- **Five sessions with various RTT (10, 50ms)**
- **Four sessions are 'thin' sessions with only 25Mbps desired rate**
- **endpointXCP does not lead to max-min fair rate allocation; it's adaptation parameter is 0.22 while 0.1 is used for GTP.**

Comparison Study: Various # of Sessions

- **M-to-1; The link utilization of single ‘fat’ session when sharing with different number of ‘thin’ sessions**
- **The aggregate desired rate from ‘thin’ sessions is half of the sink capacity**



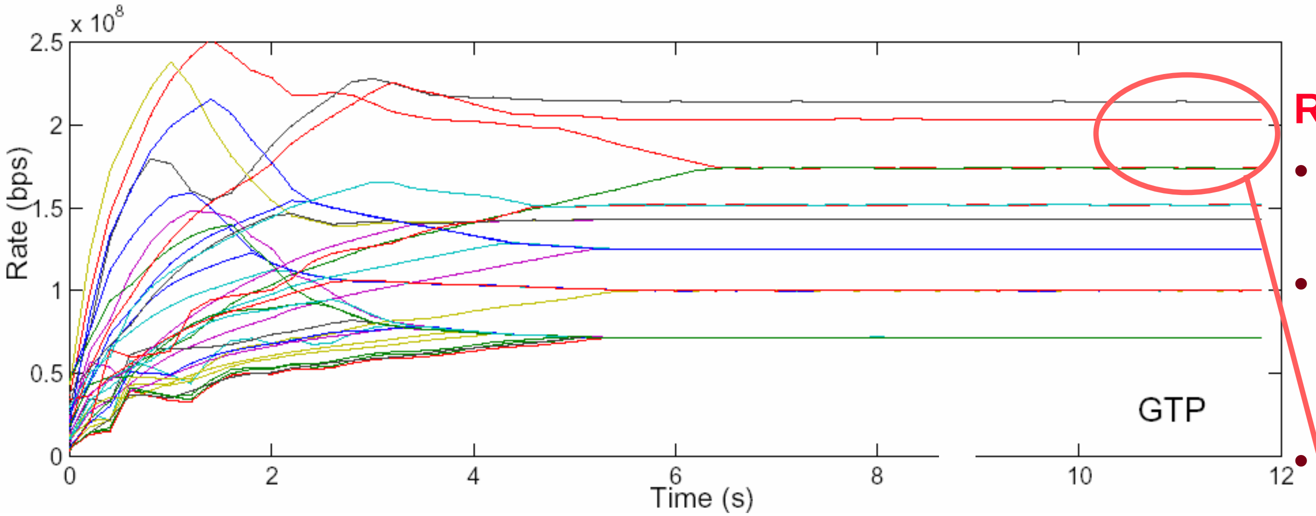


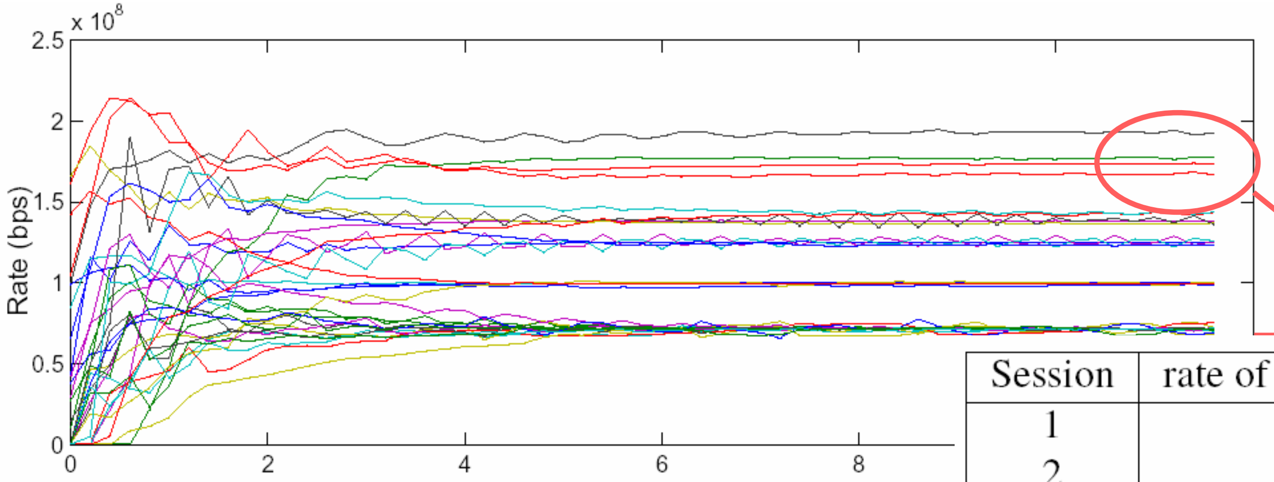
Comparison between GTP and endpointXCP: 8 to 8, 32 sessions

- **8 sources and 8 sinks;**
- **Each source initiate 4 sessions to 4 random sinks**
- **RTT: 1-100ms**
- **Node capacity: 500 Mbps**

Source	Sink	Source	Sink
1	1, 2, 2, 7	5	1, 3, 4, 5
2	1, 2, 3, 4	6	2, 2, 3, 4
3	2, 3, 4, 4	7	2, 5, 8, 8
4	3, 4, 4, 8	8	1, 4, 5, 6

Comparison between GTP and endpointXCP: 8 to 8, 32 sessions (2)





Results

- **GTP converges to the max-min rate allocation**
- **Throughput**
  - **GTP 3.50Gbps**
  - **XCP 3.11Gbps**
- **Fairness**

Session	rate of XCP (Mbps)	rate of GTP (Mbps)
1	192.6	214.2
2	177.7	203.5
3	173.9	174.0
4	166.9	174.0

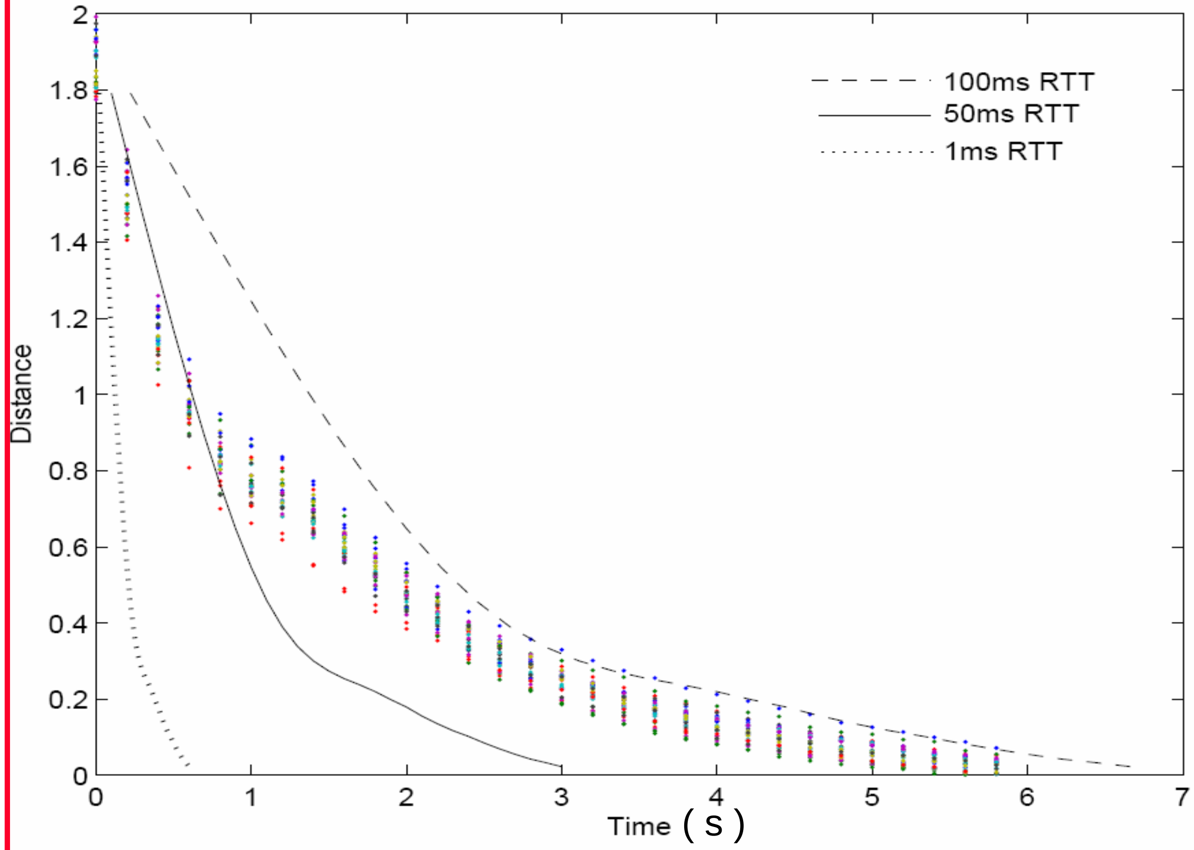
Experiment Result: Validate Convergence Property with Large Networks

- **1024-node network: 512 sources and 512 sinks**
- **Each source initiates 4 sessions to random sinks**
- **RTT: 1-100 ms; Random step sizes and control intervals for each session.**

- **2-norm Distance:**

$$D(t) = \left[\sum_{i=1}^{K} (x_i(t) - x_i^*)^2 \right]^{1/2}$$

- **30 test cases**
- **Compare with fixed RTT cases**



2-norm distance between current rates and the max-min rate allocation

OptIPuter

Discussion and Conclusion

- We study the problem of fair sharing end node capacities in lambda networks
- Added difficulties by unknown desired rates and large bandwidth-delay product
- End-node based approaches (endpointXCP and GTP) are able to achieve the fair sharing goal.
- endpointXCP achieves 'constrained max-min fair'; GTP achieves max-min fair
- endpointXCP needs kernel level implementation; GTP can be at user (or system middleware) level
- These rate allocation schemes can be extended to support capacity allocation with larger granularity:
  - Traffic shaping (to be placed on top of other aggressive session-based transport protocols)
  - dynamic allocation of lambdas based on demands

- Questions?

OptIPuter