REBOOK:
a Network Resource Booking Algorithm

draft-montessoro-rebook-00

Pier Luca Montessoro, Riccardo Bernardini
montessoro@uniud.it, riccardo.bernardini@uniud.it

Multimedia Networking and Applications lab
DIEGM - University of Udine, Italy

The research group

(a multidisciplinary approach)

- Pier Luca Montessoro, coordinator, full professor in computer science (networking and software development)
- Franco Blanchini, full professor in controls (distributed control functions)
- Mirko Loghi, assistant professor in computer science (networking, hardware and software development)
- Riccardo Bernardini, assistant professor in telecommunications (multimedia encoding and networking)
- Daniele Casagrande, assistant professor in controls (distributed control functions)
- Stefan Wieser, research assistant in computer science (networking and software development)

Our possible contribution to ICN

- ICN can benefit from congestion- and flow-controlled transport of objects from a given location to the interested receiver

- REBOOK provides deterministic, dynamic and scalable resource reservation
  - → maximum delivery time for generic NDOs
  - → adequate transport performance for multimedia streaming services

- REBOOK can be useful for some instances of ICN
- (We are looking for feedbacks!)

REBOOK

- IS NOT another reservation protocol
- IS a distributed algorithm for efficient status information handling within intermediate nodes
- provides an open framework for congestion avoidance/control, fast packet forwarding and other features
- can be applied to existing or new protocols
- provides interaction and feedbacks between the network and the hosts/applications
- provides circuit performance for packet forwarding, for free
- high degree of flexibility (IPv4, IPv6, multicast)

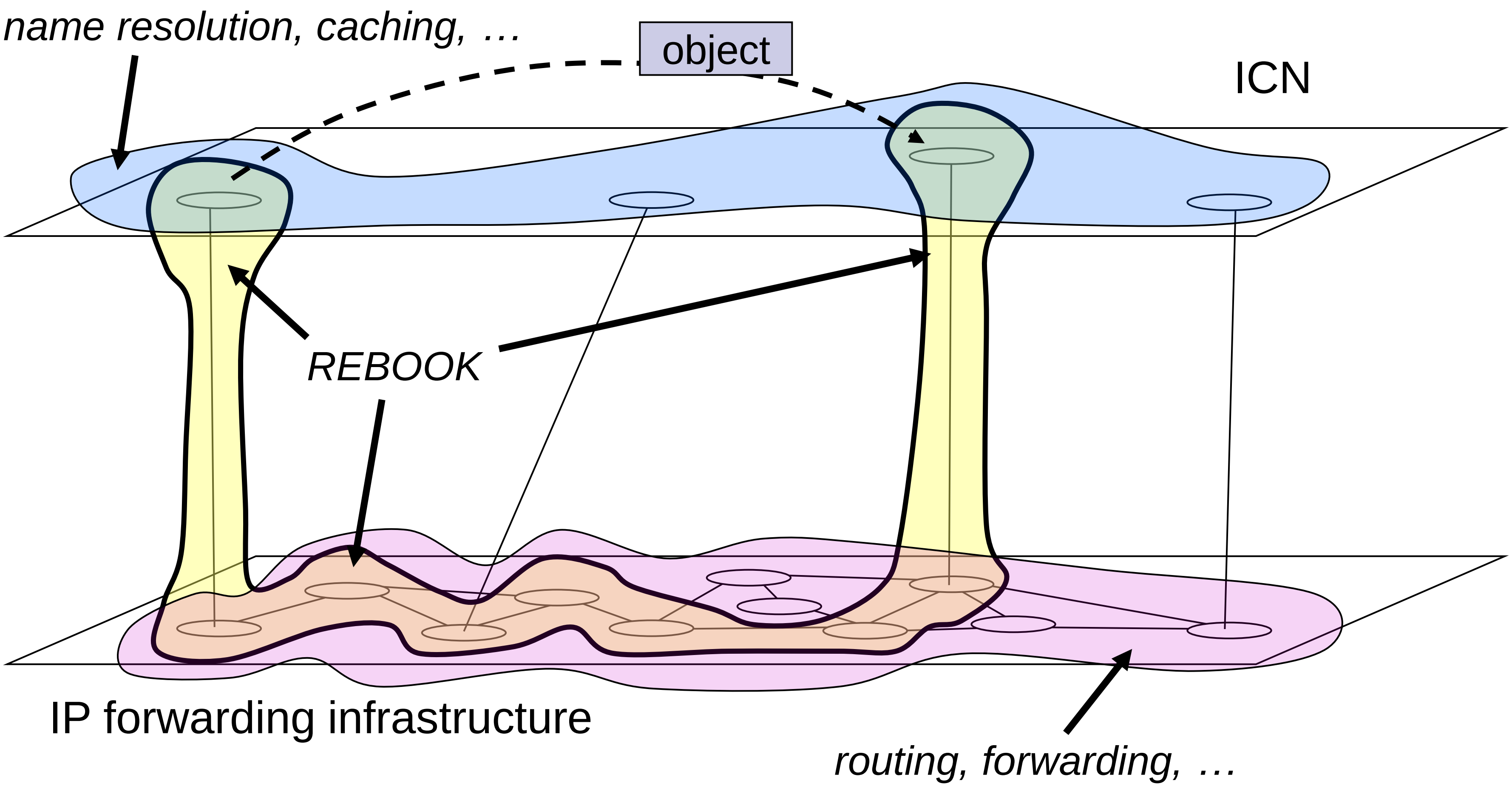
REBOOK and ICN

- REBOOK: new paradigm
  - routers, senders and receivers cooperate and handle per-flow state information

- ICN: new architecture
  - routers, senders and receivers are merged
    - cooperation becomes natural
    - they can trust each other
  - REBOOK can be useful to improve the transport services for ICN based on packet switching

- Deployment
  - REBOOK is designed for incremental deployment
  - it works even along partially rebook-aware routes
  - we guess ICN represents an ideal environment for its implementation and deployment

REBOOK and ICN



The Question

"Routers cannot keep state information for each connection (flow) traversing a node. It does not scale".

- In practical applications, is it still true with today's technology?

A tale of space and time…

Available memory

Computation time

Space

In 4 GB of memory:

- ~86 millions of flow information @ 50 bytes per flow

86 millions of flows means:

- ~688 Gbps @ 8 kbps per flow
- ~33 Tbps @ 384 kbps per flow

Not an issue for the control plane of ICN nodes routing modules

Time: here comes REBOOK

The enabling algorithm:
DLDS (Distributed Linked Data Structure)

During setup

- store resource reservation information in routers

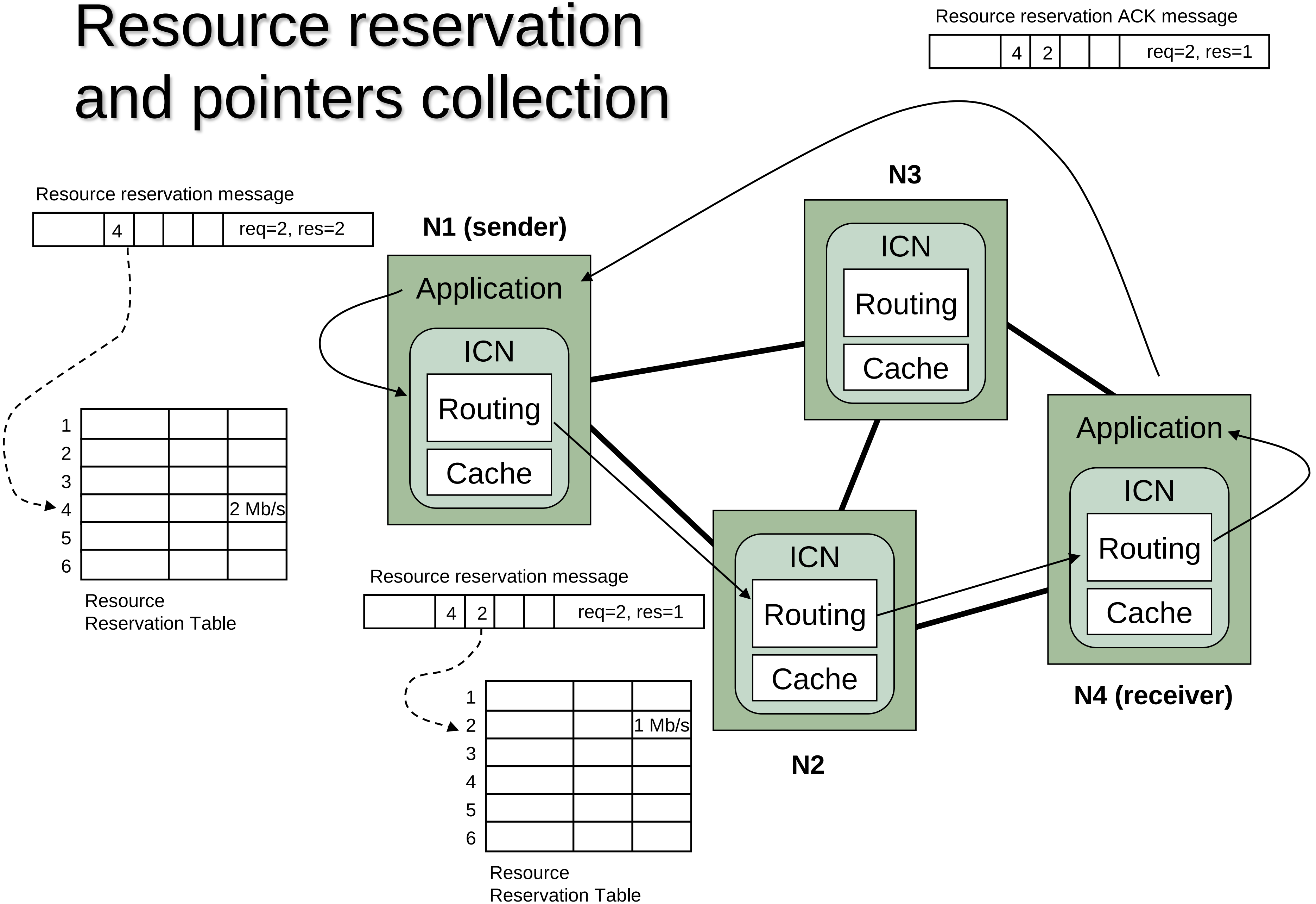
  AND

- keep track of pointers (memory addresses or indexes in tables) along the path

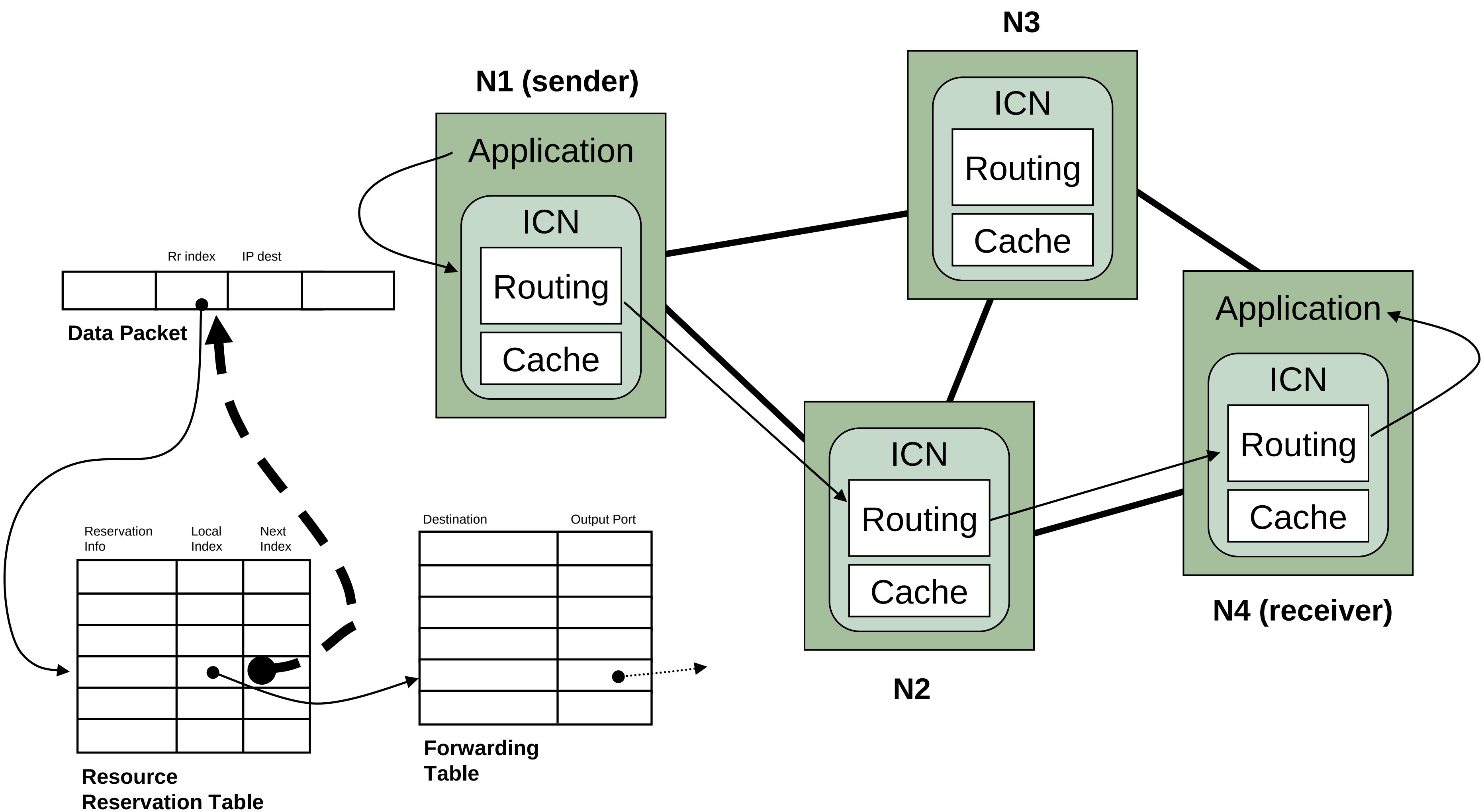
Afterwards

- use the pointers to access status information without searching

Resource reservation and pointers collection



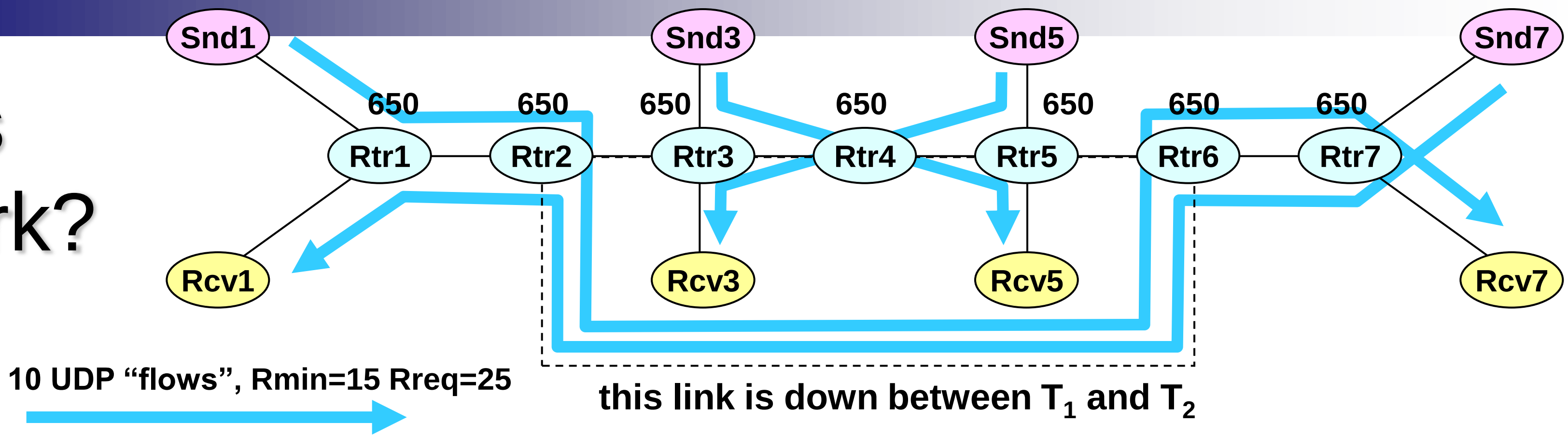
Fast packet forwarding

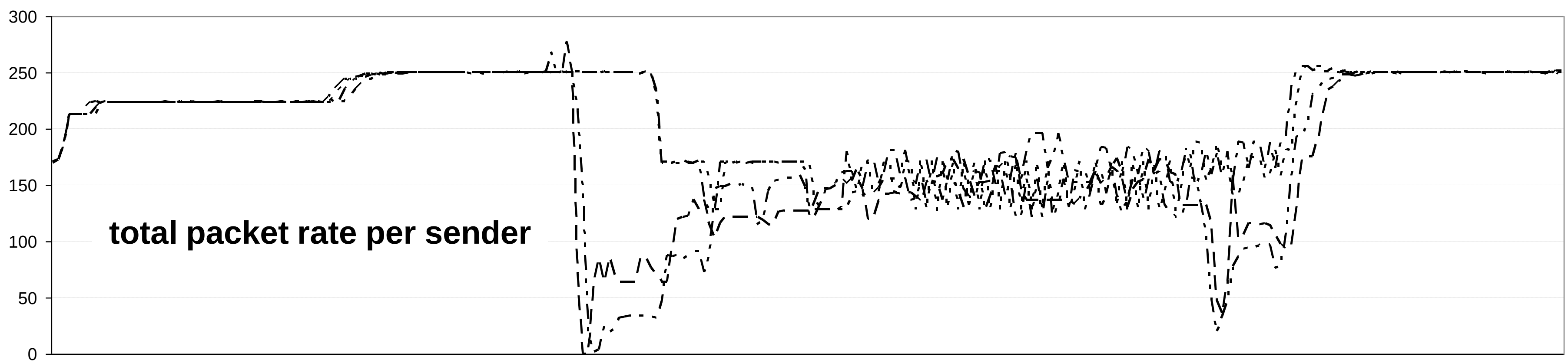


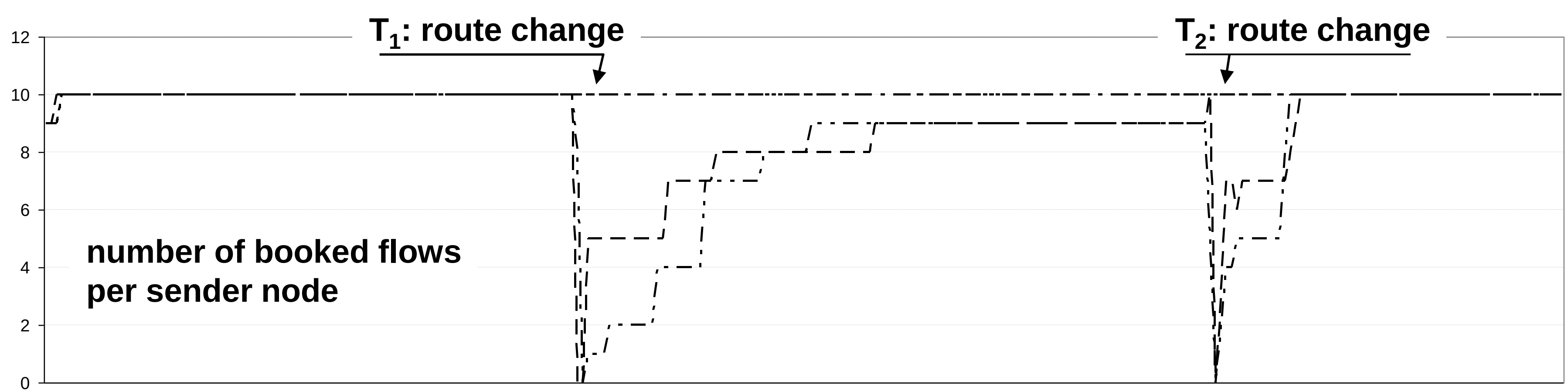
A few problems

- route changes, disappearing flows, end nodes or routers faults
  - high speed consistency check
  - highly efficient, low priority table cleanup process
- need to dynamically change assigned resource amounts
  - partial release
  - distributed control function for optimality and fairness

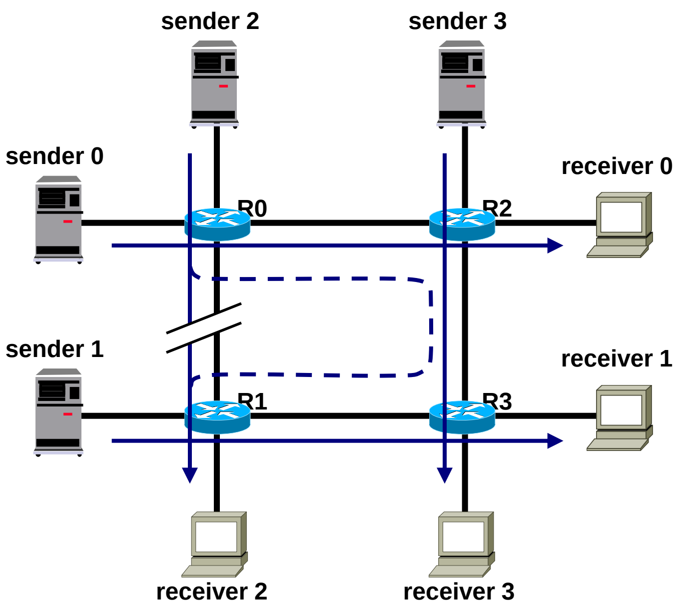
Does it work?


Snd1
Snd3
Snd5
Snd7
Rtr1
Rtr2
Rtr3
Rtr4
Rtr5
Rtr6
Rtr7
Rcv1
Rcv3
Rcv5
Rcv7
650
650
650
650
650
650
650
10 UDP “flows”, Rmin=15 Rreq=25
this link is down between T_1 and T_2

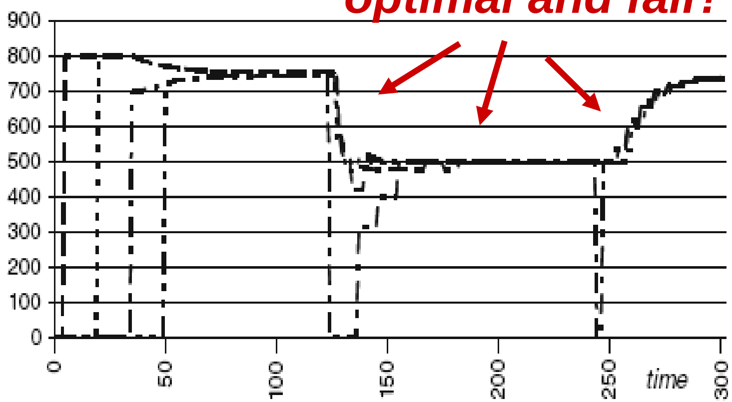

300
250
200
150
100
50
0
total packet rate per sender

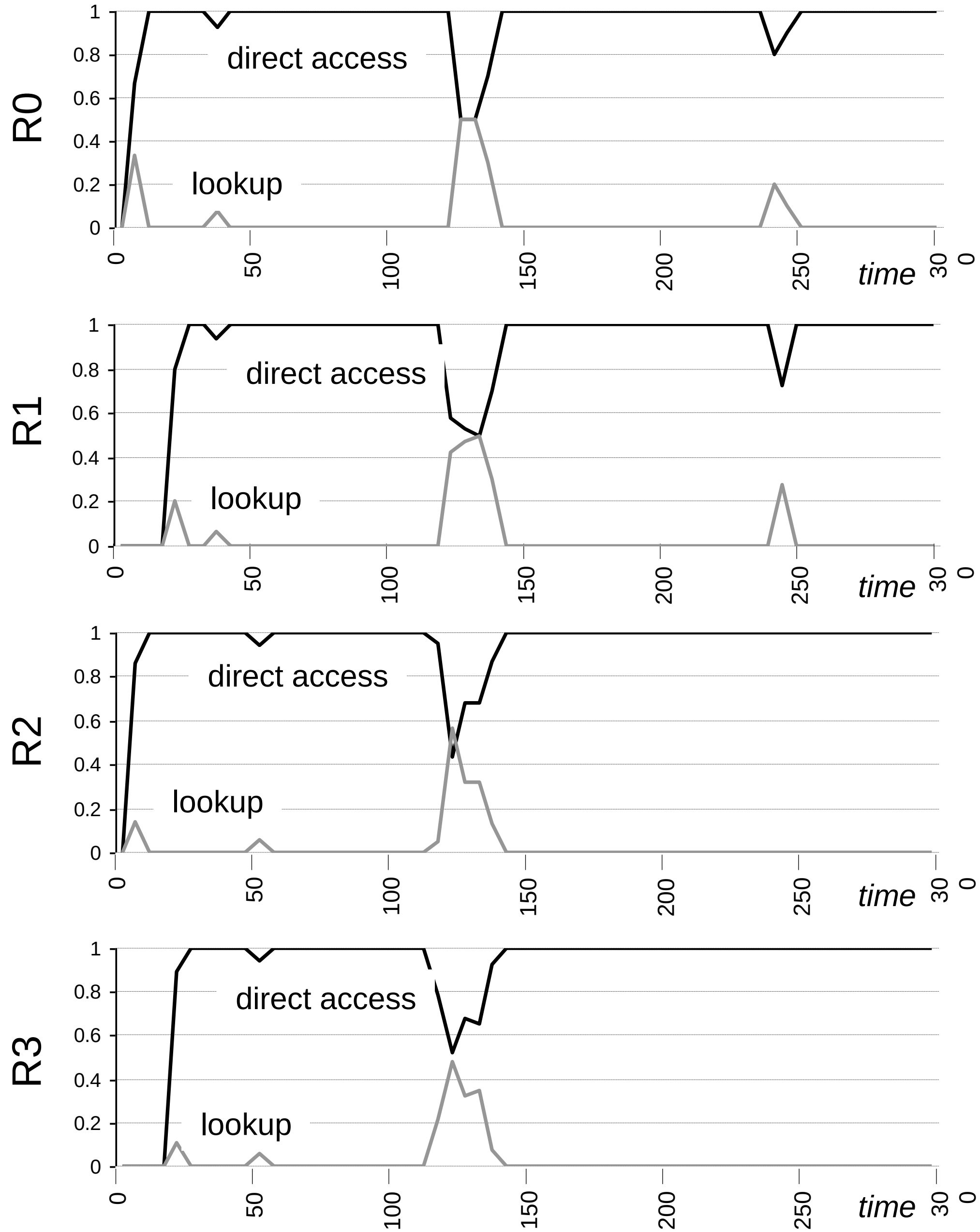

12
10
8
6
4
2
0
T_1: route change
T_2: route change
number of booked flows
per sender node

Does it work? *(cont'd)*



optimal and fair!





"… and running code"

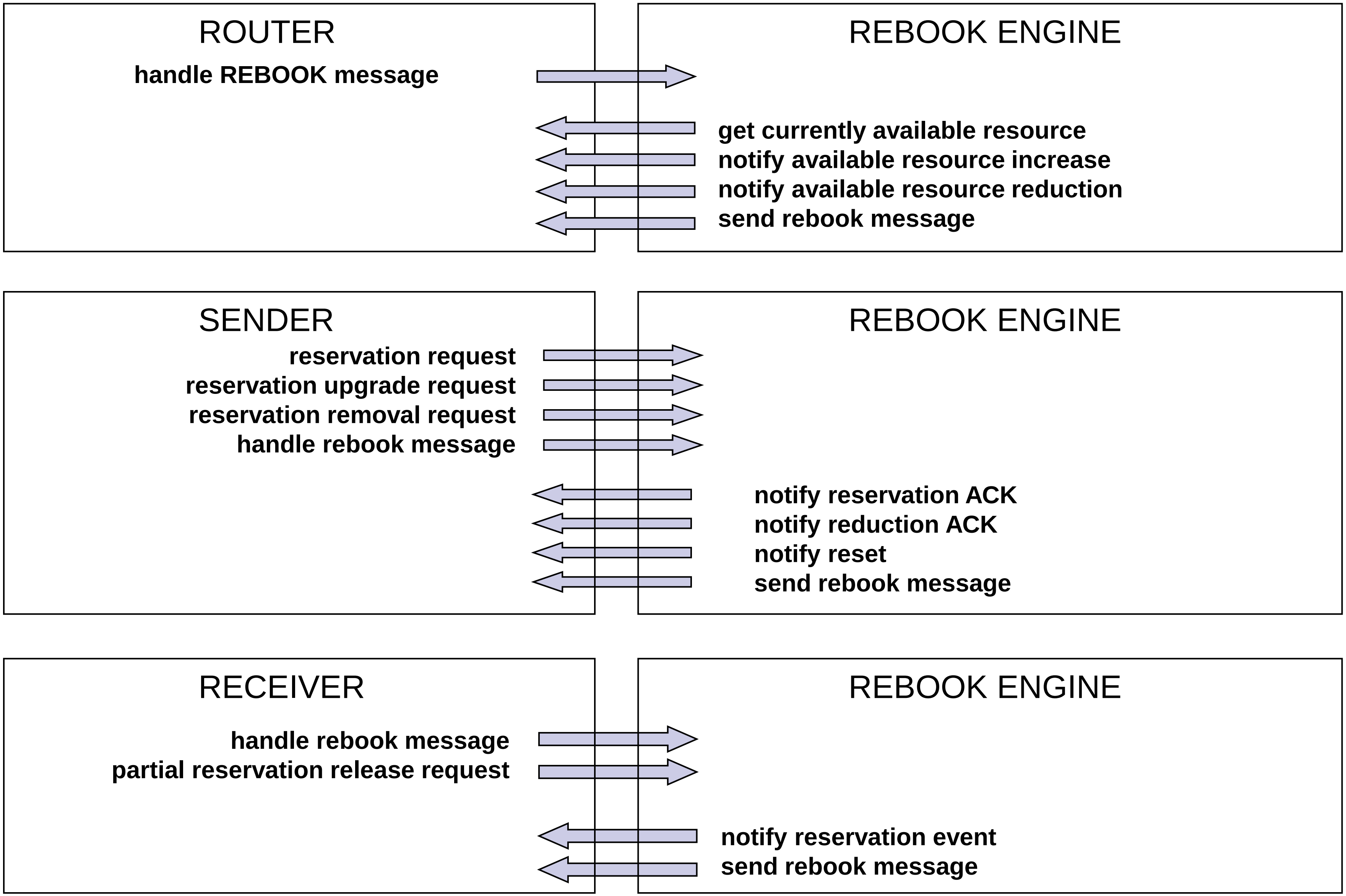
- Current prototype
  - Extremely lightweight hosting protocol
  - Add-on modules for applications and routing engines
  - C/C++ static or dynamic link library
  - Multi-platform (Linux gcc, Microsoft Visual Studio)

Object code size (gcc compiler, Intel Core 2)

Module	Size
Router	30 KB
Sender	20 KB
Receiver	8 KB

- Under development:
  - Embedding in Linux kernel
  - Usage of unassigned IP Option Alert flag values

Prototype



ROUTER
handle REBOOK message
REBOOK ENGINE
get currently available resource
notify available resource increase
notify available resource reduction
send rebook message
SENDER
reservation request
reservation upgrade request
reservation removal request
handle rebook message
REBOOK ENGINE
notify reservation ACK
notify reduction ACK
notify reset
send rebook message
RECEIVER
handle rebook message
partial reservation release request
REBOOK ENGINE
notify reservation event
send rebook message

Performance

CPU times have been measured on a 1.6 GHz Intel® Core 2 computer

CPU times (DLDS and resource reservation management)

Activity	configuration	CPU time
setup (incl. res. reserv.)	10,000 flows	200 ns once per flow
setup (incl. res. reserv.)	10,000,000 flows	250 ns once per flow
Keepalive message handling	10,000 flows	100 ns every 5 seconds
Keepalive message handling	10,000,000 flows	190 ns every 5 seconds
RR table entries release	10,000 flows	25 ns per flow
RR table entries release	10,000,000 flows	48 ns per flow
RR table cleanup	10,000,000 entries	100 ms every 15 seconds

CPU times (direct access forwarding, including consistency check)

Activity	configuration	CPU time
DLDS forwarding table access	1,000,000 routes	10.57 ns per packet
DLDS forwarding table access	100,000,000 routes	10.65 ns per packet

Traffic Overhead (relative to a 10-minutes 384 kb/s multimedia flow)	
Distributed linked data structure setup	0.002 %
Keepalive message	0.08 %
Alert option, pointer and hop counter in data packets	0.6 %

Deployment

- No interaction with (nor change in) the underlying routing protocols is required
- Autonomous recovery of errors, faults and route changes
- If information stored in the DLDS becomes obsolete, packet handling is reverted to best-effort, lookup-driven forwarding
- Packets are never dropped nor misrouted
- It works even on partially REBOOK/DLDS-unaware paths
- It works across multiple Autonomous Systems
- It does not require any agreement between network managers
- It can be implemented in an extremely lightweight protocol

References

- Pier Luca Montessoro, Daniele De Caneva. "REBOOK: a deterministic, robust and scalable resource booking algorithm," DOI 10.1007/s10922-010-9167-8, Journal of Network and Systems Management (Springer), Pp. 1-29 ISSN: 1064-7570 (Print) 1573-7705 (Online)
- Pier Luca Montessoro, "Distributed Linked Data Structures for Efficient Access to Information within Routers", Proceedings of IEEE 2010 International Conference on Ultra Modern Telecommunications, 18-20 October 2010, Moscow (Russia), ISBN 978-1-4244-7286-4
- Pier Luca Montessoro, "Efficient Management and Packets Forwarding for Multimedia Flows," Journal of Network and Systems Management (Springer), 2012, DOI: 10.1007/s10922-012-9232-6
- Franco Blanchini, Daniele Casagrande, Pier Luca Montessoro, "A novel algorithm for dynamic admission control of elastic flows," Proc. of 50th FITCE congress, Palermo, Italy, August 31th – September 3rd, 2011, pp.110-115, ISBN: 978-1-4577-1208-1, DOI: 10.1109/FITCE.2011.6133421
- Pier Luca Montessoro, Stefan Wieser, Laszlo Böszörmenyi, "An Efficient and Scalable Data-Structure for Resource Reservation and Fast Packet Forwarding in Large Scale Multimedia Overlay Networks," IEEE CQR 2012, 15-17 May 2012, San Diego, CA
- Pier Luca Montessoro, international patent application on DLDS, UD2010A000178 (29/9/2011), PCT/IB2011/054281 (29/9/2011)

In the articles…

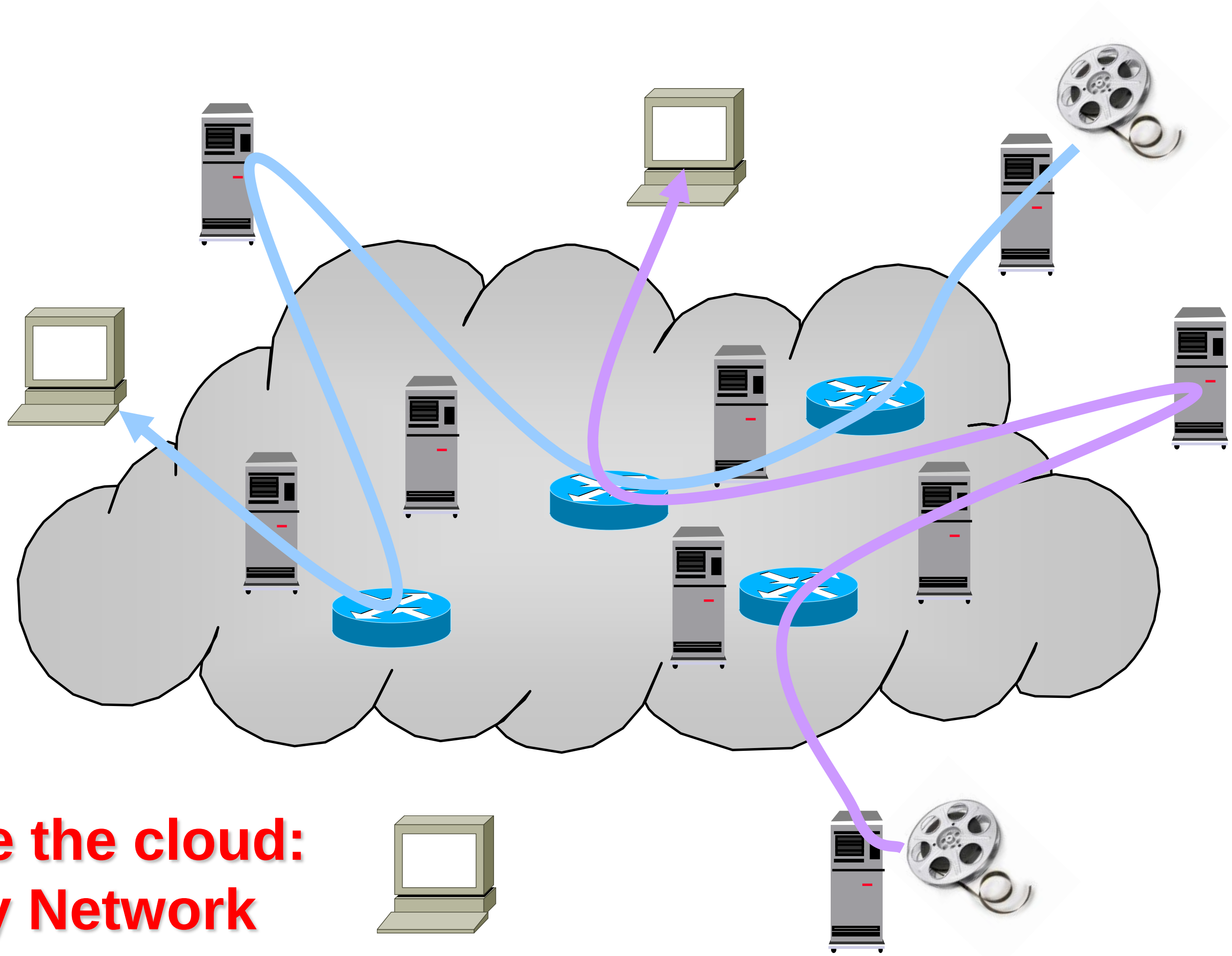
- Distributed control function for fairness and optimality
- Deployment
- Security
- Fast packet forwarding
- Implementation details

Conclusion

- Some instances of ICN can use REBOOK
  - for congestion- and flow-controlled transport of objects from a given location to the interested receiver
  - to provide fast packet forwarding in software-based routers or inexpensive hardware implementation
- Why ICN? Why REBOOK?
  - new architecture that overcome the rigid separation (and mistrust) between hosts/applications and the network

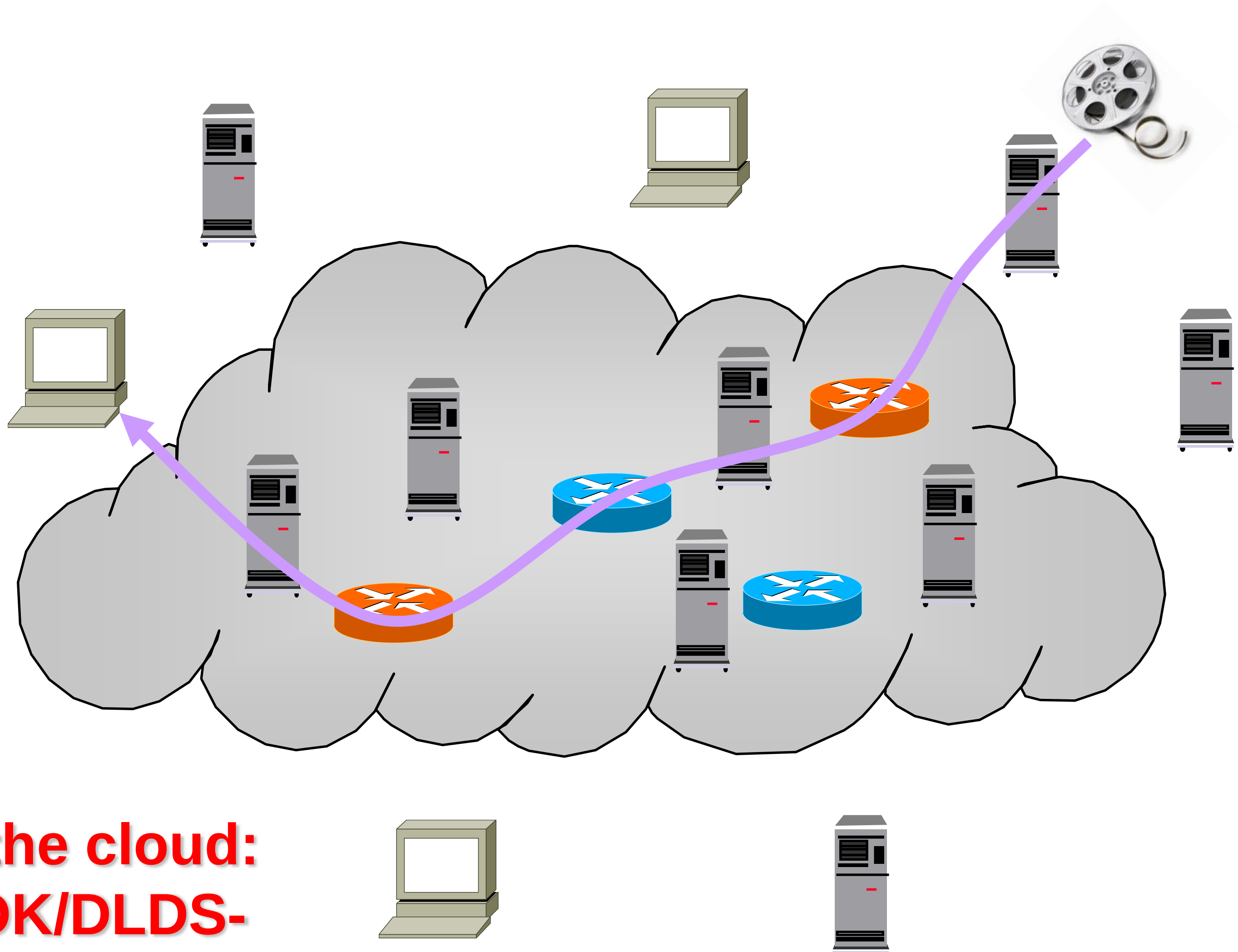
Thank you!

Other scenarios



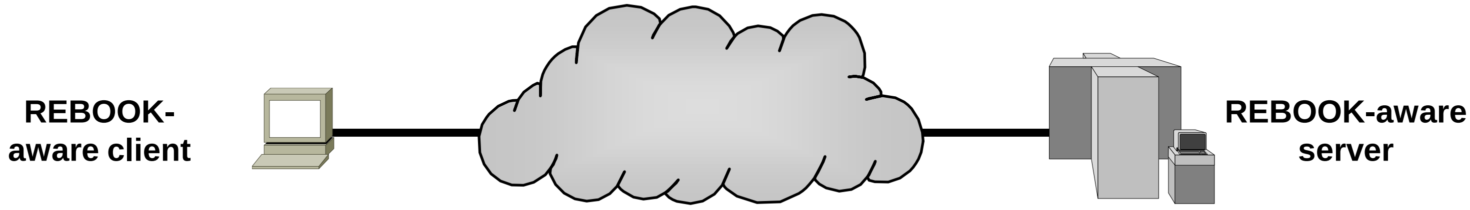
Outside the cloud:
Overlay Network

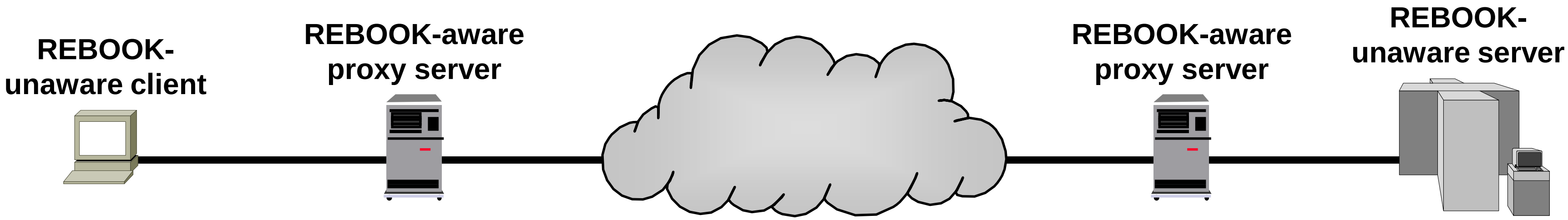
Other scenarios *(cont'd)*

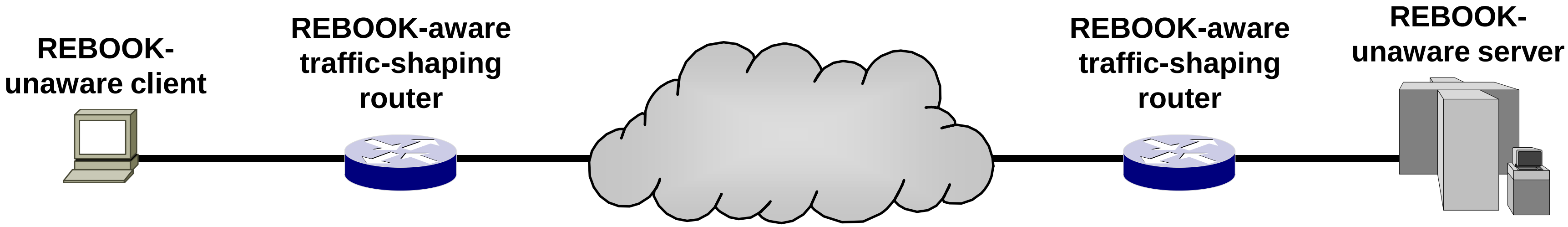


Inside the cloud: REBOOK/DLDS-aware routers

Other scenarios *(cont'd)*







Performance
(access to the forwarding table)

Speedup

(REBOOK-DLDS handling 10,000,000 routes, one flow each)

Reference	configuration	speedup
ART-16-8-8	~50 K routes	3
ART	~50 K routes	4.7
SMART	~50 K routes	4.7
CPE	~50 K routes	5.3
BSD Radix	~50 K routes	47
Binary trie	5,000 routes	138
LC-trie	5,000 routes	246
Modified LC-trie	5,000 routes	239
Prefix-tree	5,000 routes	131
DTBM	5,000 routes	191
7-FST	5,000 routes	114
2-MPT	5,000 routes	99