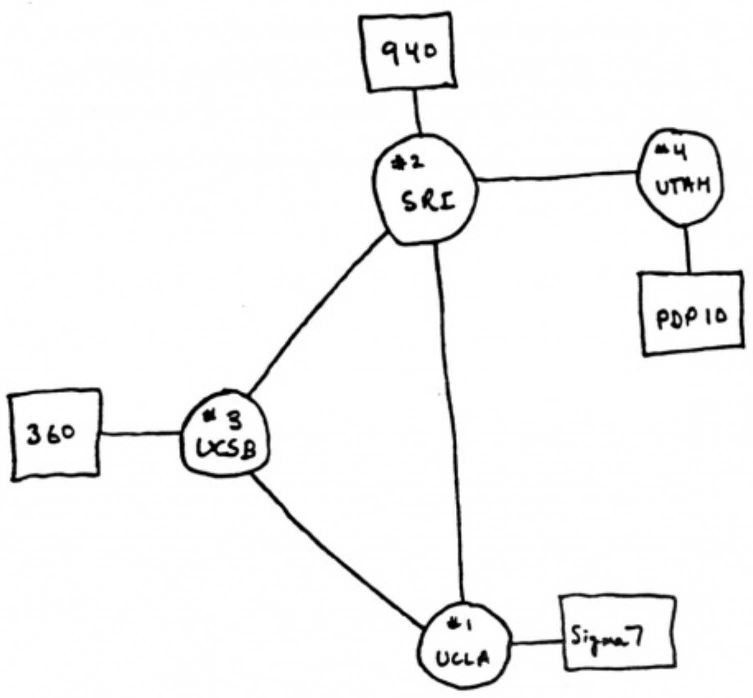
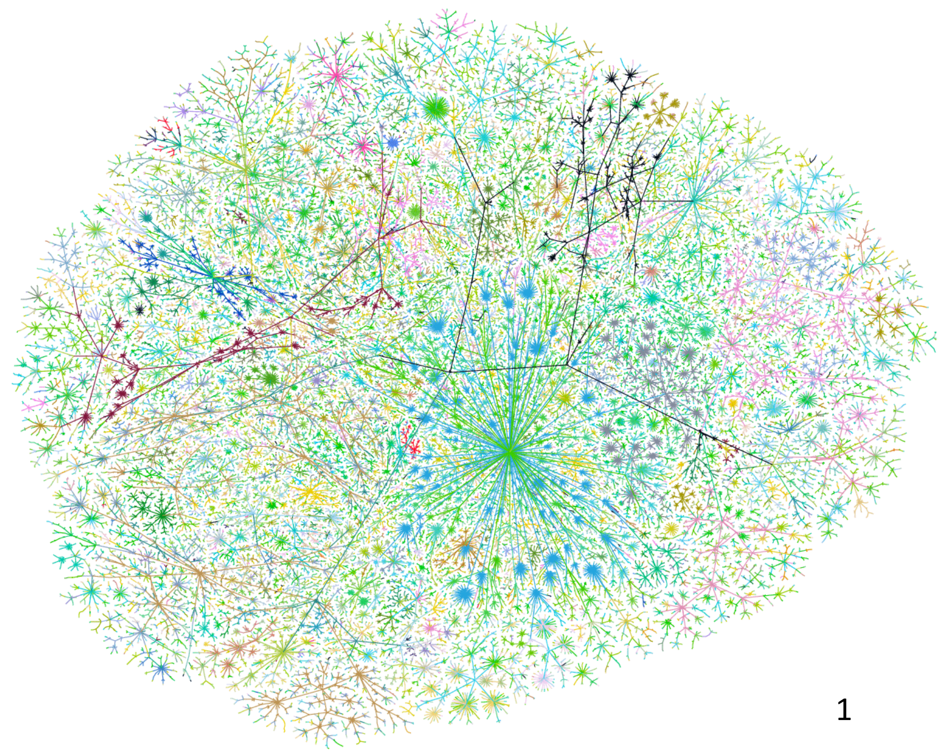
Hitting the Nail on the Head

Jennifer Rexford
Princeton University

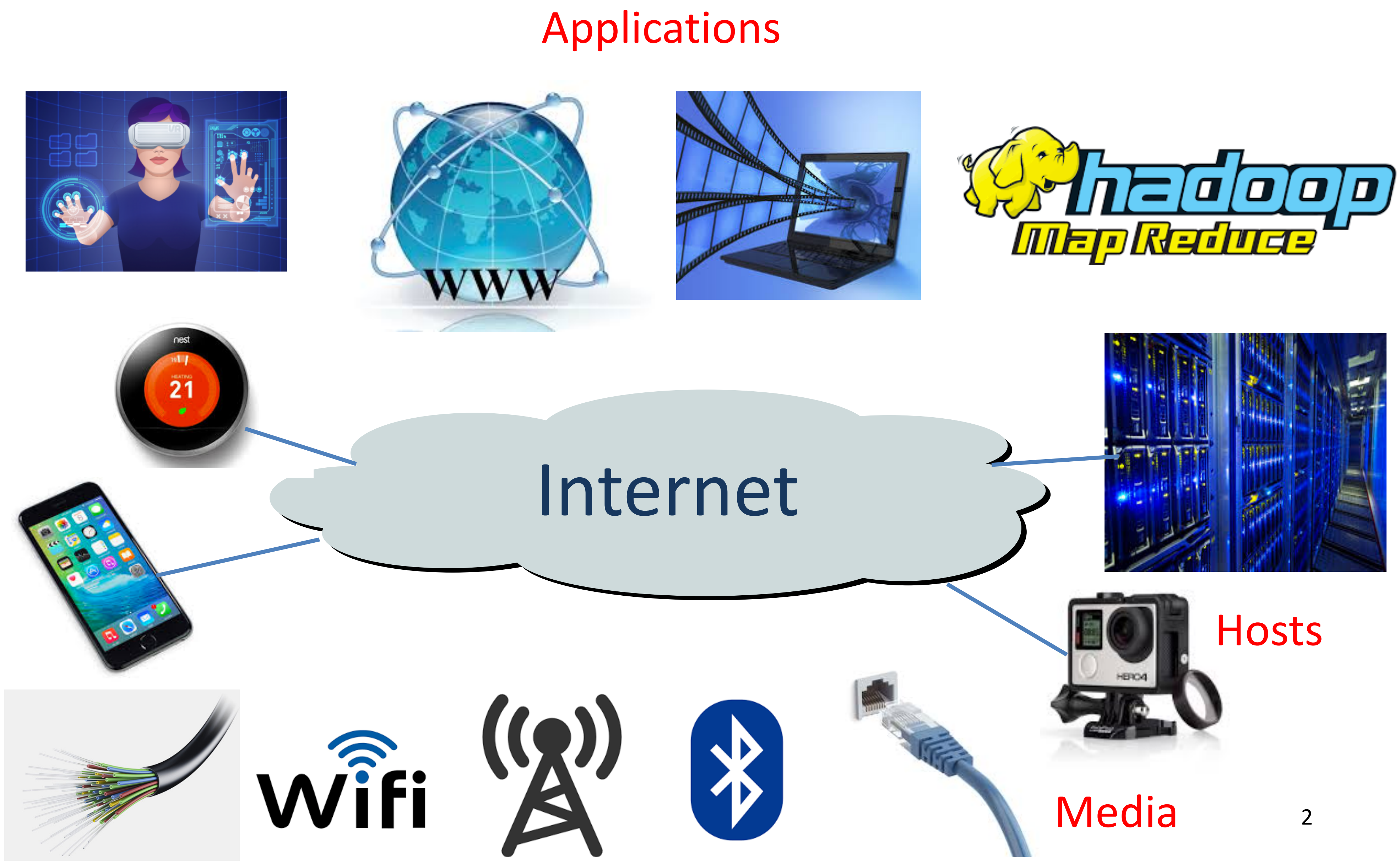
The Internet: An Exciting Time

- One of the most influential inventions of all time
  - A research experiment that escaped from the lab
  - … to become a global communications infrastructure





Near-Constant Innovation



A Grand Challenge

- Computer networks we can depend on
  - Performance, reliability, security, privacy, fairness, efficiency, autonomy, cost-effectiveness, …
- Build strong intellectual foundations
  - Better real-world networks
  - ... even as technology changes
- We cannot do it alone!

Hammers and Nails

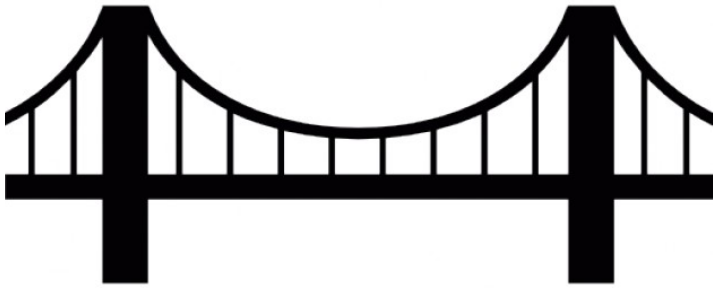
meet

Rich set of hairy research problems

Effective solution techniques

My Upbringing at at&t

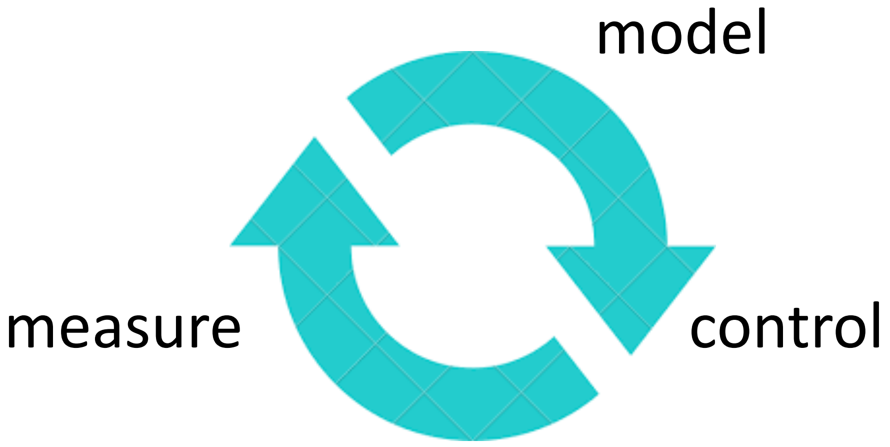
Algorithms, Optimization, Statistics



Network operations

networking research

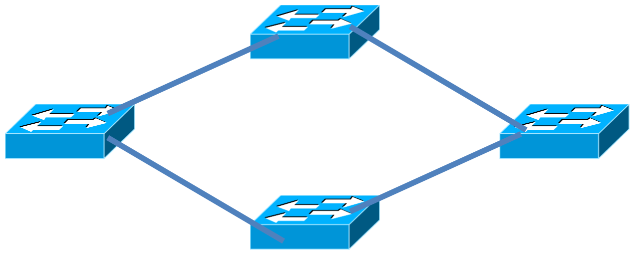
Albert Greenberg



Taking existing network as a given…

Three Example Projects

- High-level policies
  - Programming languages



- Distributed protocols
  - Optimization theory

- Traffic monitoring
  - Compact data structures

Mung Chiang

Protocols as Distributed Optimizers

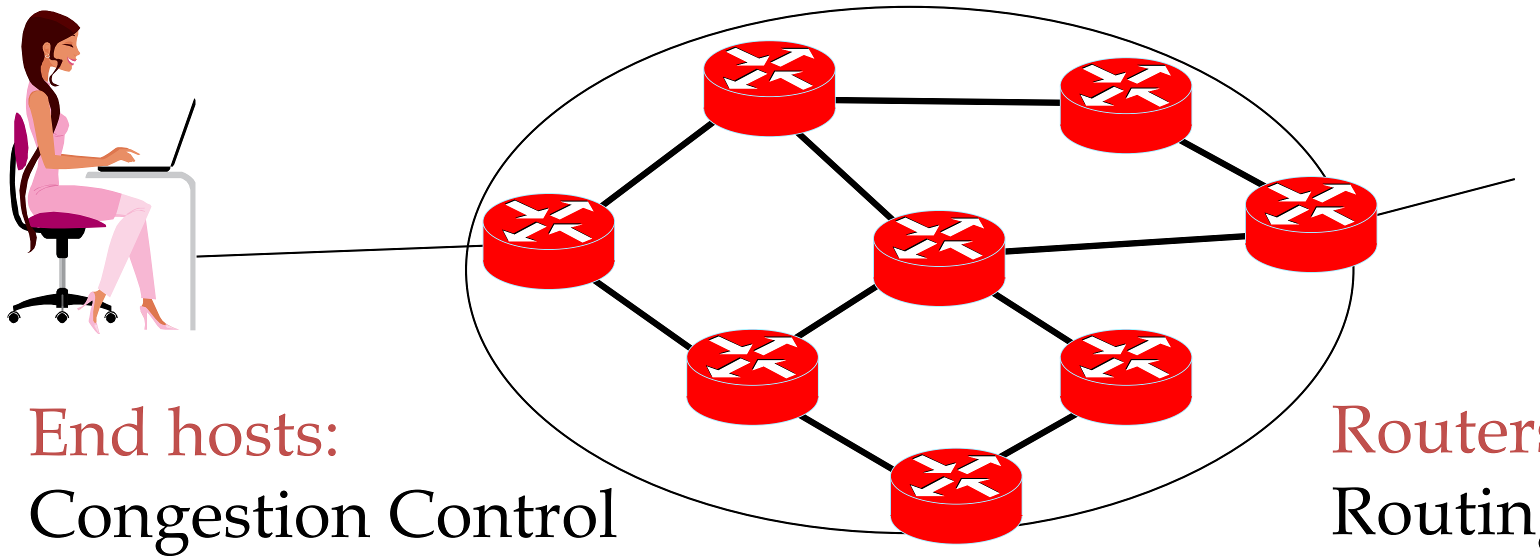
(optimization theory)

Jiayue He, Rui Zhang-Shen, Ying Li, Cheng-Yen Lee, Jennifer Rexford, and Mung Chiang, "DaVinci: Dynamically Adaptive Virtual Networks for a Customized Internet," in *ACM SIGCOMM CoNext Conference*, December 2008.

Traditional Traffic Management

- How much traffic should traverse each path?

Operator:
Traffic Engineering
(minimize congestion)

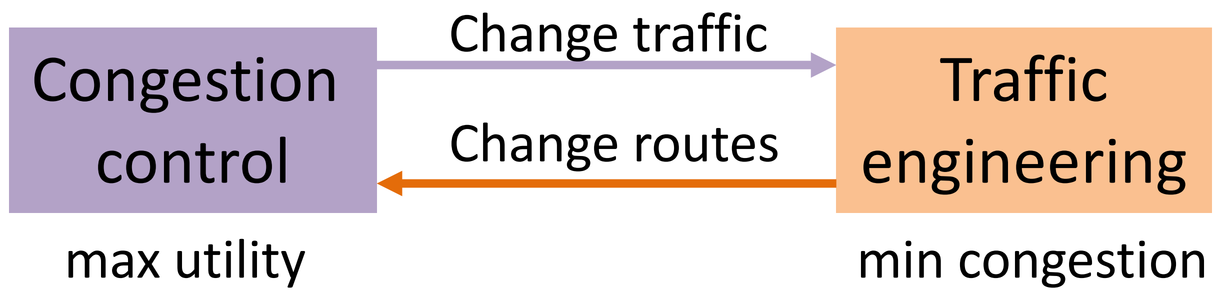


End hosts:
Congestion Control
(maximize utility, ensure fairness)

Routers:
Routing Protocol
(compute short paths)

Architectural Limitations

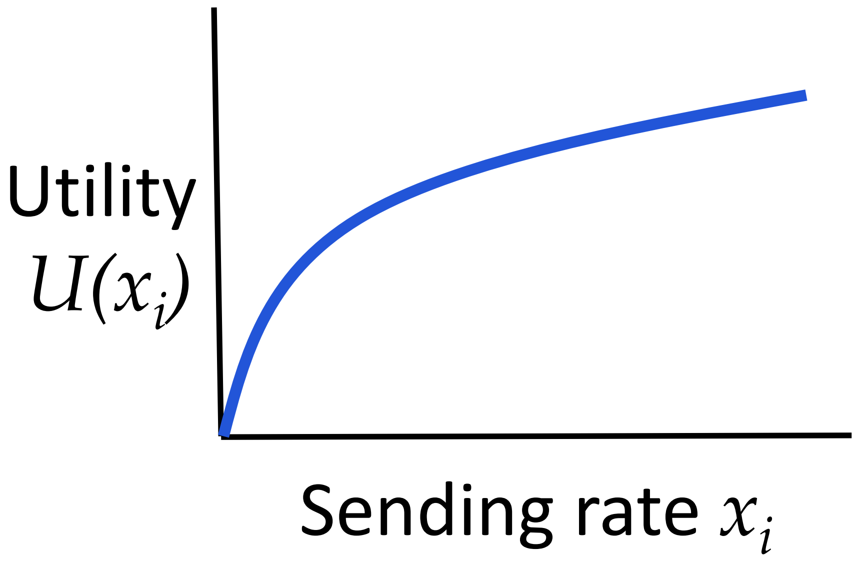
- Protocol interactions

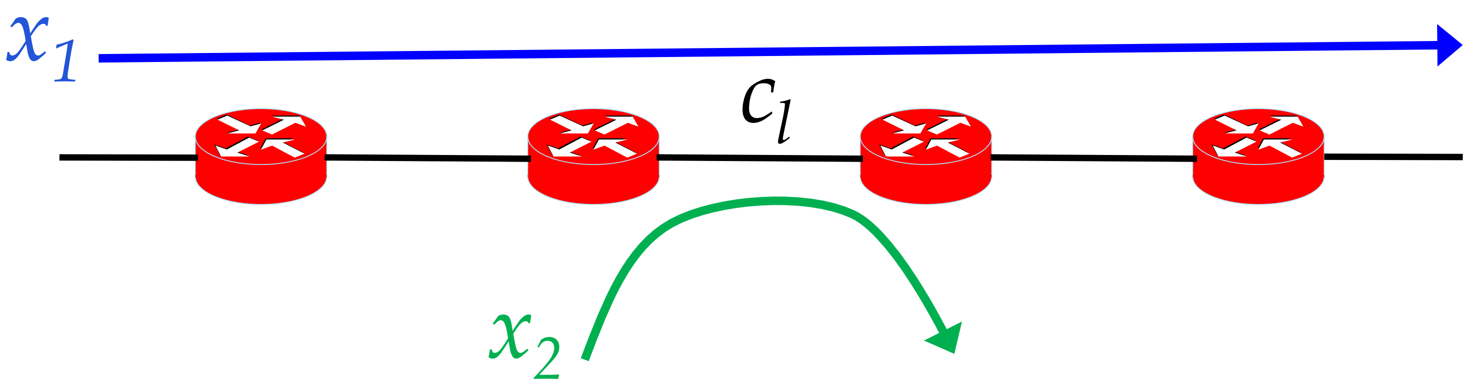


- Slow adaptation
  - Traffic engineering on longer timescale
- One-size-fits-all performance metrics
  - Throughput vs. delay-sensitive traffic

TCP as Network Utility Maximization

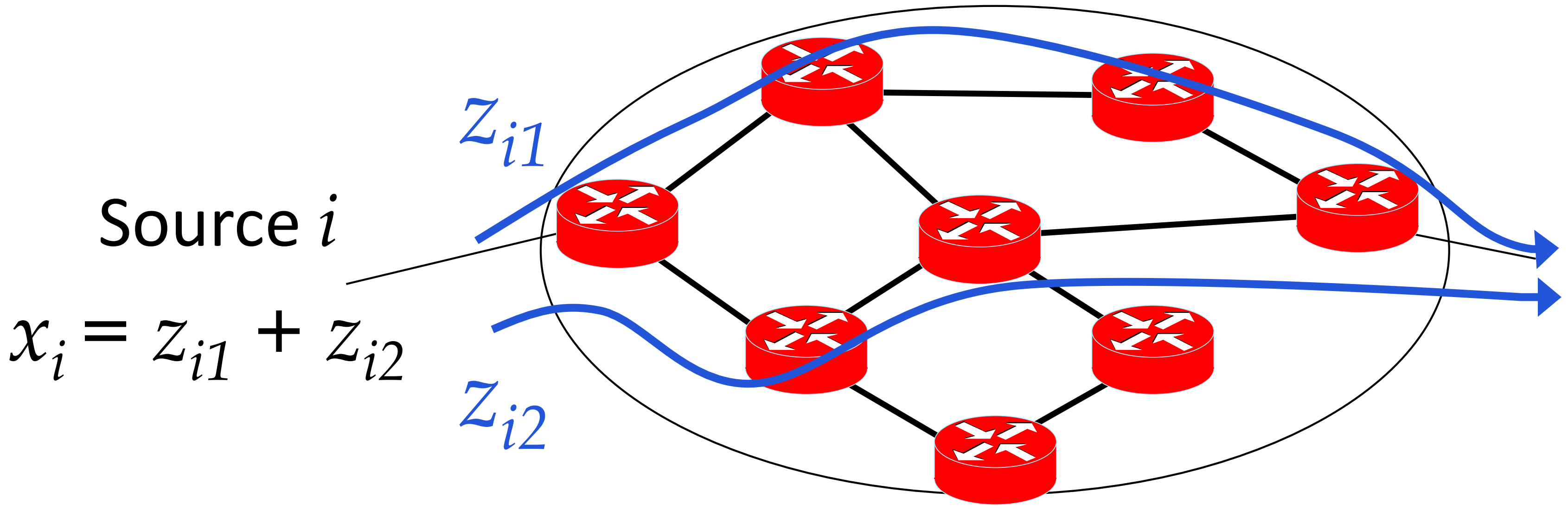
- Reverse engineering the problem TCP solves
  - Utility $U(x_i)$ as function of sending rate x_i
  - Objective $max \sum_i U(x_i)$ subject to link capacities c_l





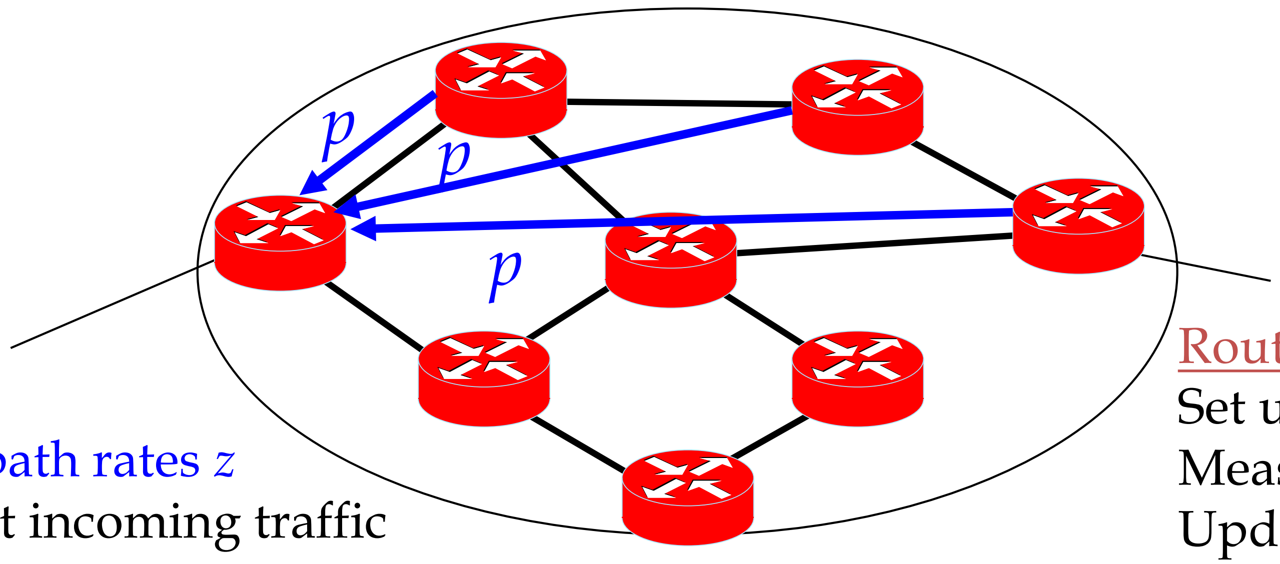
Designing Traffic Management

- Forward engineering of traffic management
  - Maximize total utility: $max \sum_i U(x_i)$
  - Routing as a variable: traffic z_{ij} on path j



Distributed Protocol

- Decompose to generate a distributed solution
  - Each link computes congestion "price" p_l
  - Each source computes path rates z_{ij} to maximize utility ($U(\sum_j z_{ij})$) subject to path cost (sum over p_l)



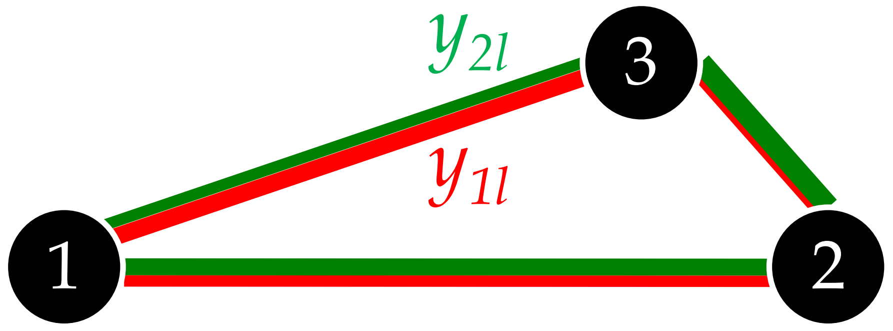
Sources:
Update path rates z
Rate limit incoming traffic

Routers:
Set up multiple paths
Measure link load
Update link prices p

Multiple Traffic Classes

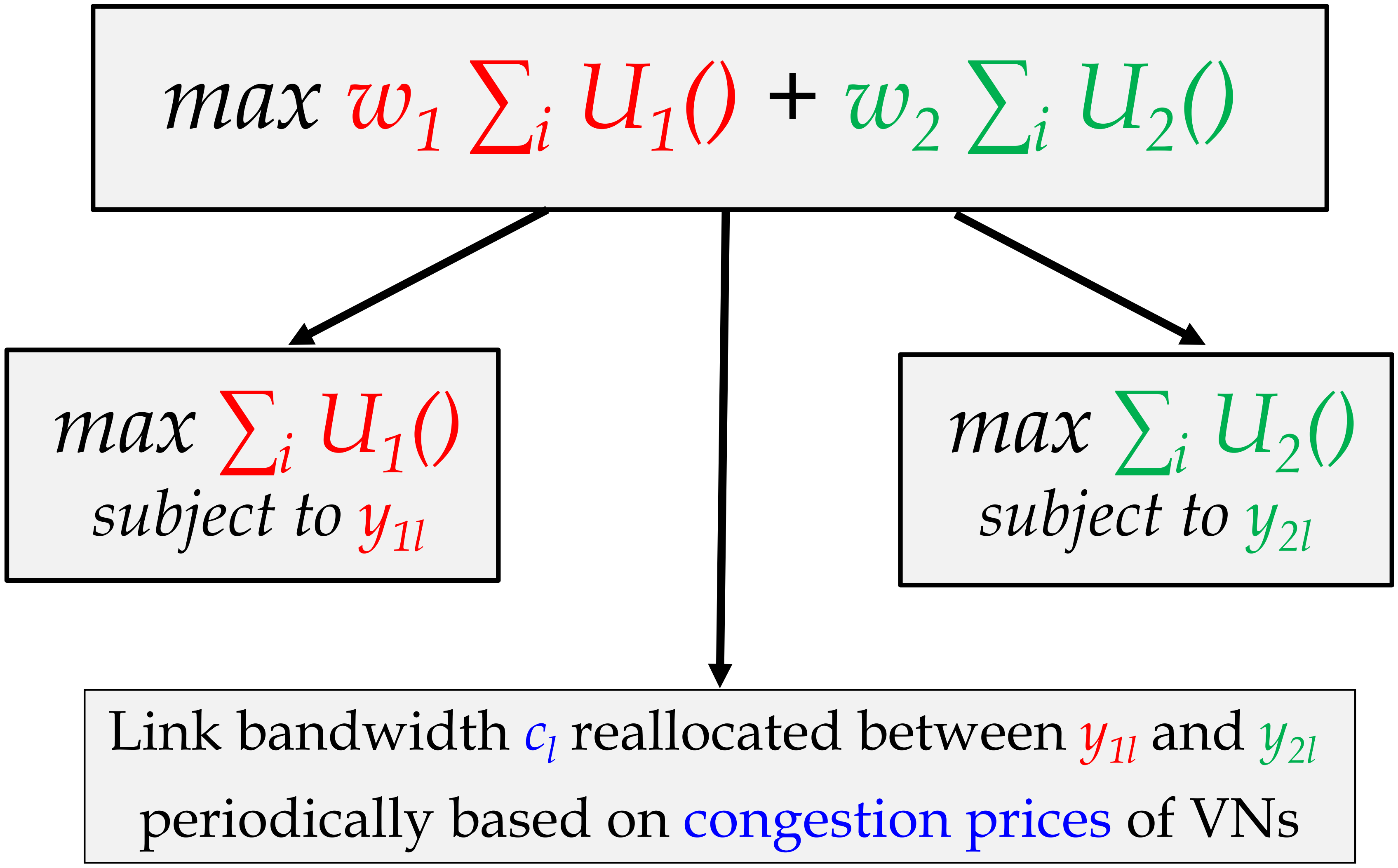
- Utility functions
  - Max throughput ($U_1()$)
  - Min latency ($U_2()$)
- Bandwidth shares
  - Share y_{1l} and y_{2l} on link l
  - Where $y_{1l} + y_{2l} = c_l$

$$max\ w_1 \sum_i U_1() + w_2 \sum_i U_2()$$



two virtual networks

Two-Stage Decomposition



$max\ w_1 \sum_i U_1() + w_2 \sum_i U_2()$

$max \sum_i U_1()$
$subject\ to\ y_{1l}$

$max \sum_i U_2()$
$subject\ to\ y_{2l}$

Link bandwidth c_l reallocated between y_{1l} and y_{2l} periodically based on congestion prices of VNs

Lesson: Protocols as Optimizers

- Start with a well-stated (optimization) problem
  - Then decompose into a distributed solution
- Benefits
  - Deeper understanding of how protocols work
  - Guarantees on optimality, convergence, etc.
- In reality, the process is iterative

Lesson: Research as Decomposition ☺

The network needs to solve this problem. →

← Too hard, please change the network.

Okay, how about this? →

- Make per-class convergence faster
  - Change objective to include link utilization u_l
  - … include both TCP utility and TE congestion!

$$max\ \left(\sum_i U(x_i) - \sum_l f(u_l/c_l)\right)$$

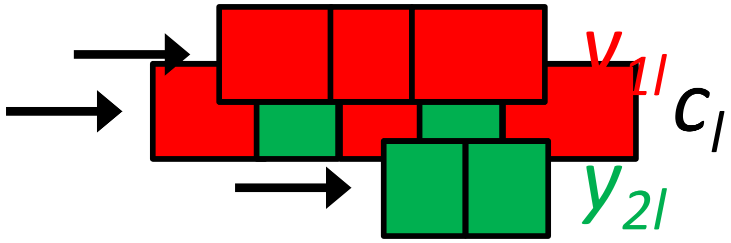
Lesson: Research as Decomposition ☺

The network needs to solve this problem. →

← Too hard, please change the network.

Okay, how about this? →

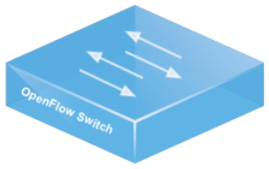
- Make the multi-class problem tractable
  - Separate queue for each traffic class per link
  - ... to decouple the per-class utility functions!



Lesson: People and Timing

- The right graduate students
  - Comfortable with both topics
  - Able to wrangle two busy advisors

- The right timing (eventually?)
  - Programmable switch hardware



David Walker

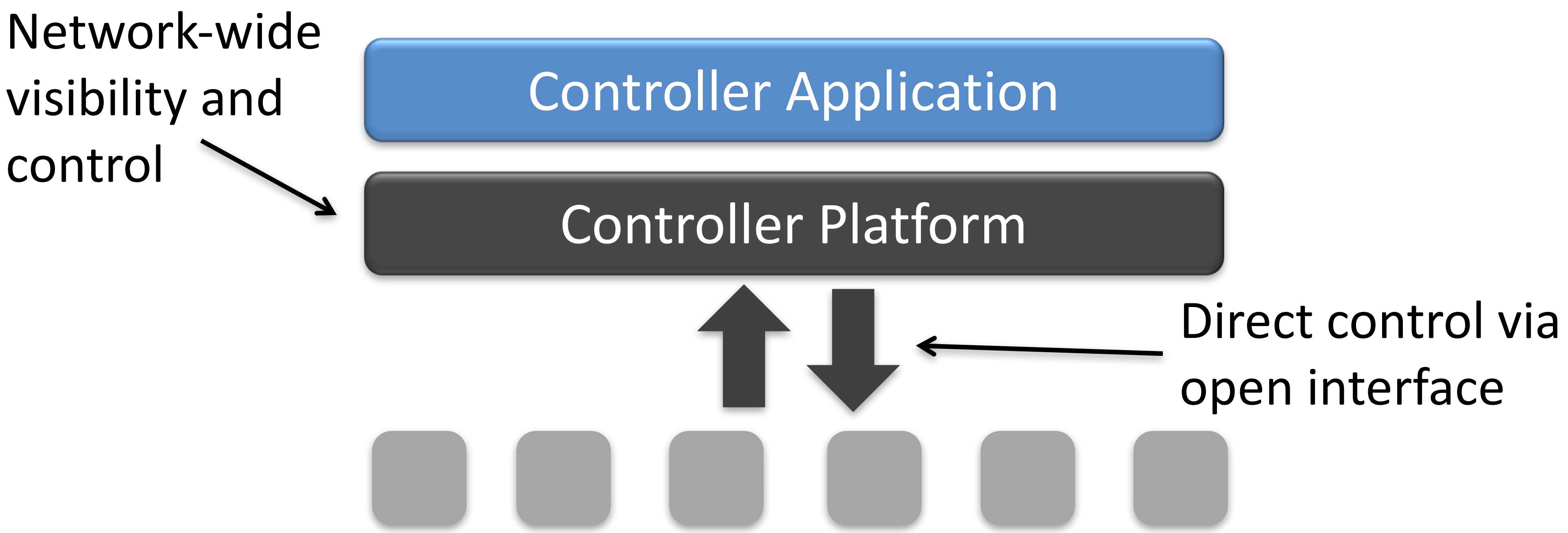
Nate Foster

Composition of Network Policies

(programming languages)

Nate Foster, Michael J. Freedman, Arjun Guha, Rob Harrison, Naga Praveen Katta, Christopher Monsanto, Joshua Reich, Mark Reitblatt, Jennifer Rexford, Cole Schlesinger, Alec Story, and David Walker, "Languages for software-defined networks," *IEEE Communications Magazine*, Feb 2013.

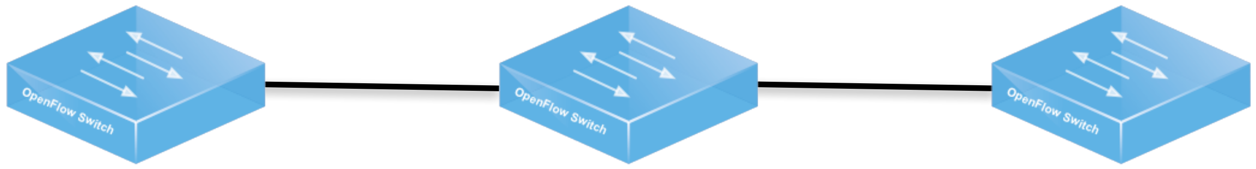
Software-Defined Networking (SDN)


Network-wide visibility and control
Controller Application
Controller Platform
Direct control via open interface

From distributed protocols to (centralized) controller applications

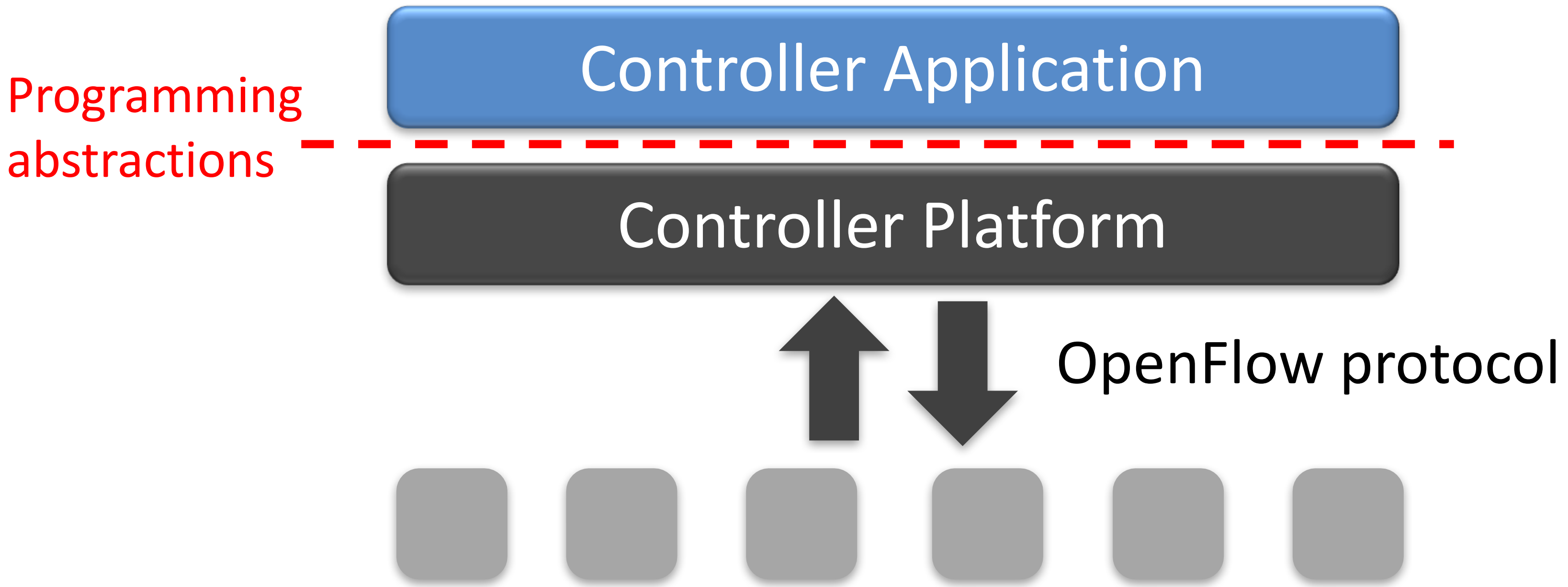
Simple, Open Data-Plane API

- Prioritized list of rules
  - Pattern: match packet header bits
  - Actions: drop, forward, modify, send to controller
  - Priority: disambiguate overlapping patterns
  - Counters: #bytes and #packets



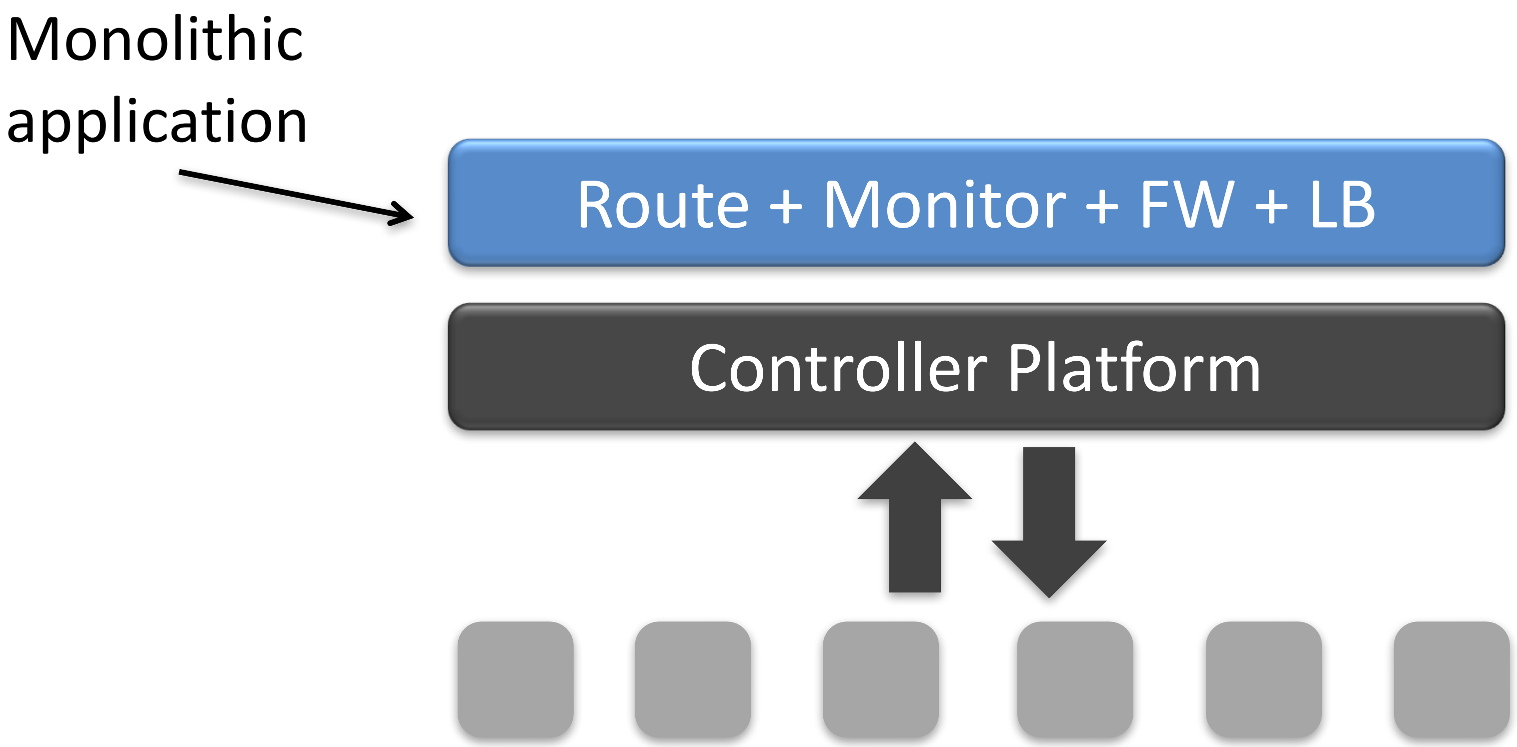
1. srcip=1.2.*.*, dstip=3.4.5.* → drop
2. srcip = *.*.*.*, dstip=3.4.*.* → forward(2)
3. srcip=10.1.2.3, dstip=*.*.*.* → send to controller

Writing SDN Controller Applications



OpenFlow is a mechanism, not a linguistic formalism.

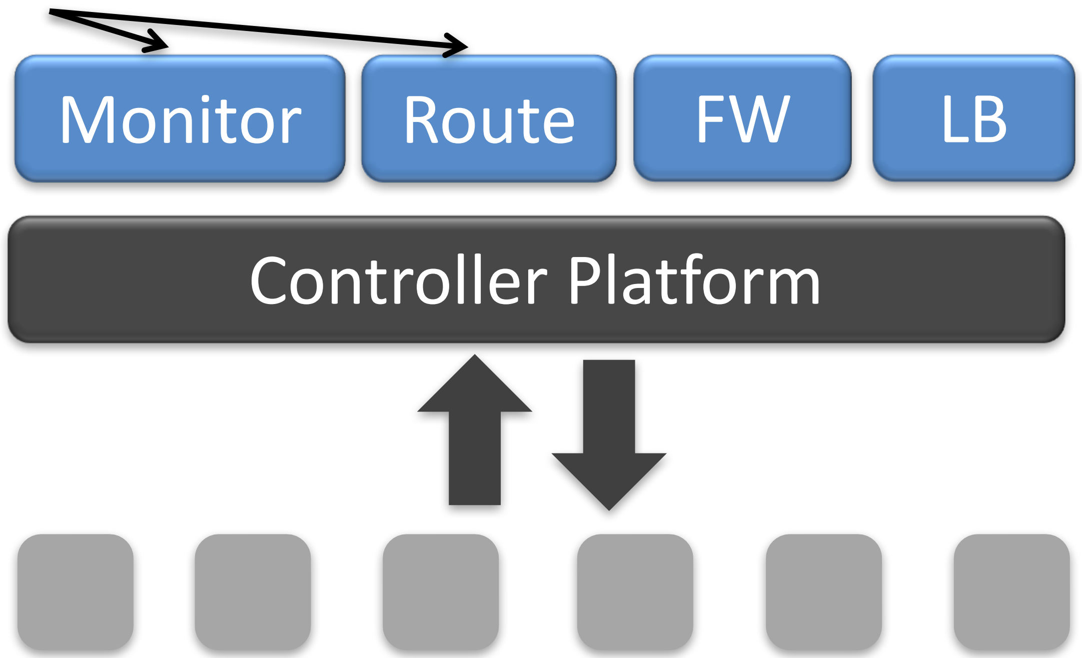
Combining Many Networking Tasks



Hard to program, test, debug, reuse, port, …

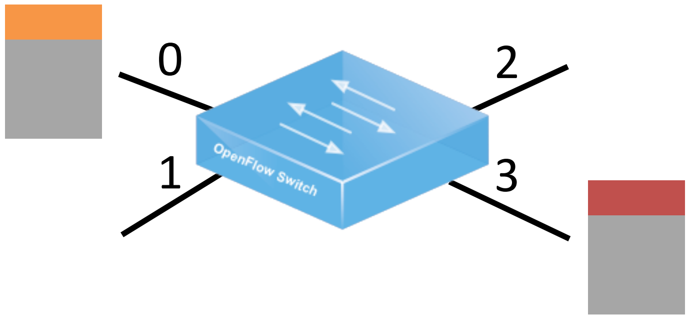
Modular Controller Applications

Each module *partially* specifies the handling of the traffic



Abstract OpenFlow: Policy as a Function

- Located packet
  - Packet header fields
  - Packet location (e.g., switch and port)
- Function of a located packet
  - To a *set* of located packets
    - Drop, forward, multicast
  - Packet modifications
    - Change in header fields and/or location

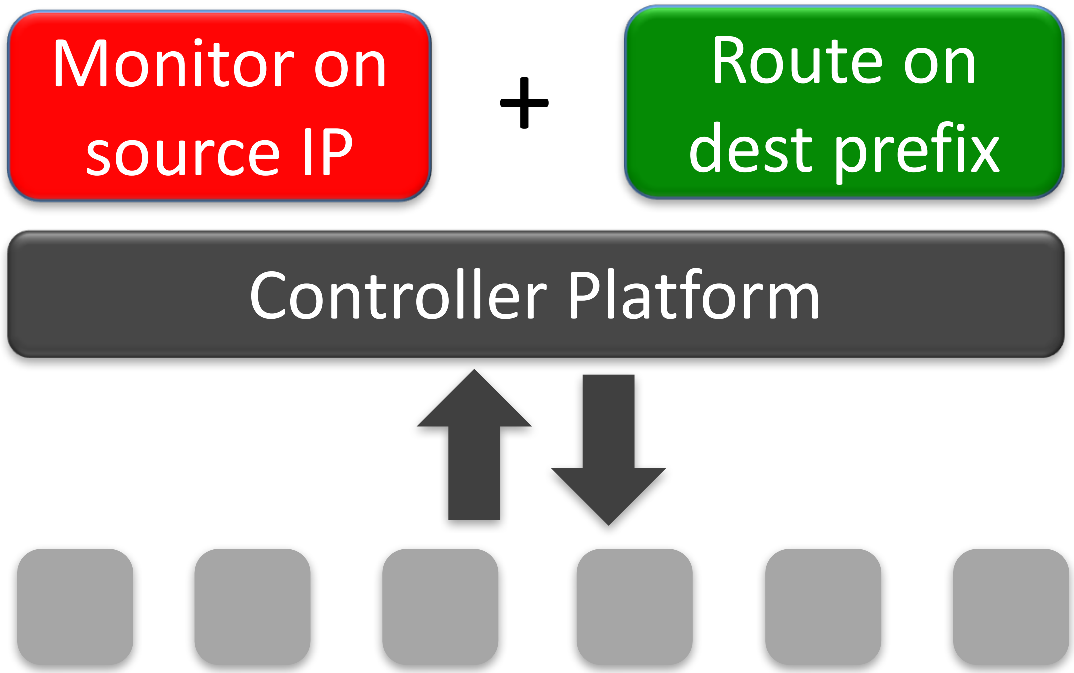


dstip == 1.2.3.4 & srcport == 80 → port = 3, dstip = 10.0.0.1

Parallel Composition (+)

srcip == 5.6.7.8 → count
srcip == 5.6.7.9 → count

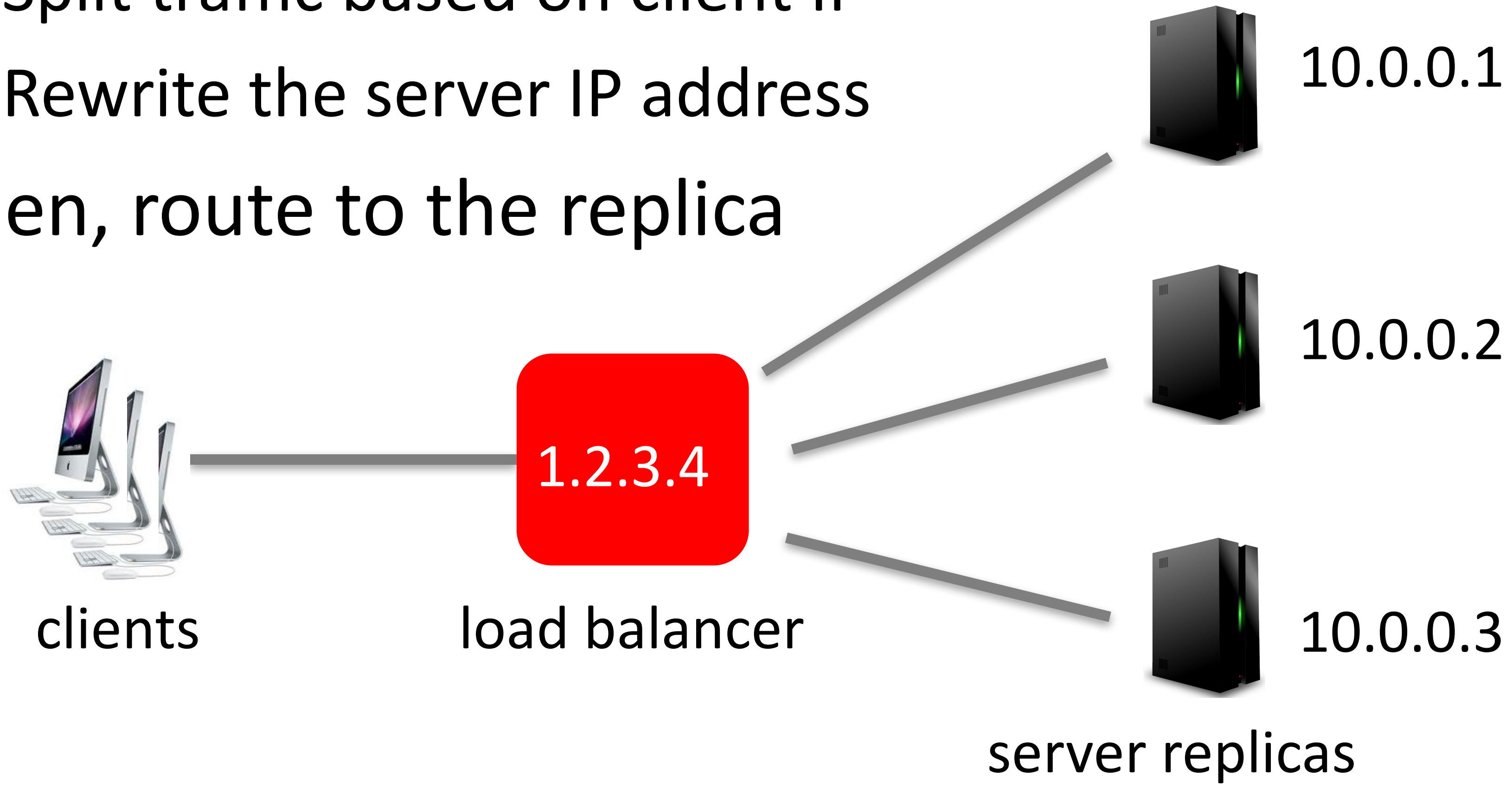
dstip == 1.2/16 → fwd(1)
dstip == 3.4.5/24 → fwd(2)



srcip == 5.6.7.8, dstip == 1.2/16 → fwd(1), count
srcip == 5.6.7.8, dstip == 3.4.5/24 → fwd(2), count
srcip == 5.6.7.9, dstip == 1.2/16 → fwd(1), count
srcip == 5.6.7.9, dstip == 3.4.5/24 → fwd(2), count

Example: Server Load Balancer

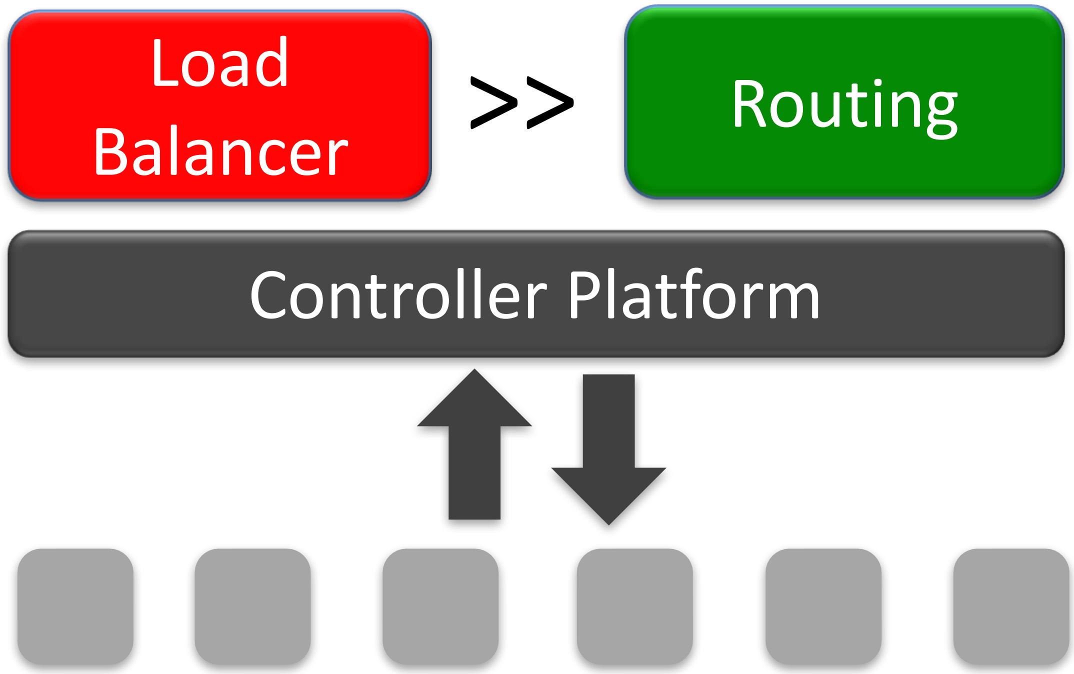
- Spread client traffic over server replicas
  - Public IP address for the service
  - Split traffic based on client IP
  - Rewrite the server IP address
- Then, route to the replica



Sequential Composition (>>)

srcip==0*, dstip==1.2.3.4 → dstip=10.0.0.1
srcip==1*, dstip==1.2.3.4 → dstip=10.0.0.2

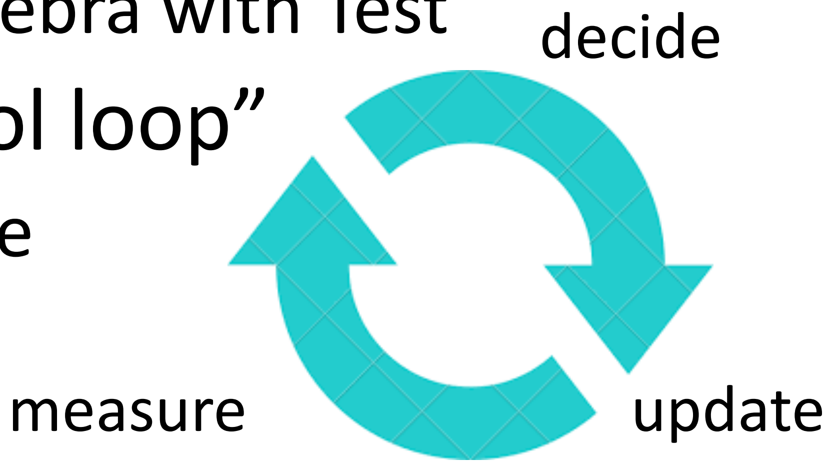
dstip==10.0.0.1 → fwd(1)
dstip==10.0.0.2 → fwd(2)



srcip==0*, dstip==1.2.3.4 → dstip = 10.0.0.1, fwd(1)
srcip==1*, dstip==1.2.3.4 → dstip = 10.0.0.2, fwd(2)

Lessons: Abstraction and Composition

- OpenFlow was an important first step
  - Generalizes many networking devices
  - Easy to explain to non-experts
- Thinking compositionally
  - Precisely defining the meaning of programs
  - Creating simple, reusable building blocks
  - NetKAT: Network Kleene Algebra with Test
- Abstractions for the “control loop”
  - Measure, decide, and update



Lessons: Embedded People (and Code)

- Nate Foster’s postdoc at Princeton
  – Embedded in the networking group
- Learning by doing
  – Writing SDN apps in NOX
  – … and then the abstractions followed
- Recruiting others
  – Open-source software and tutorials
  – Presentations, summer school, collaborations
  – Publishing in both PL and networking venues
- Thinking ahead: P4 language for new hardware

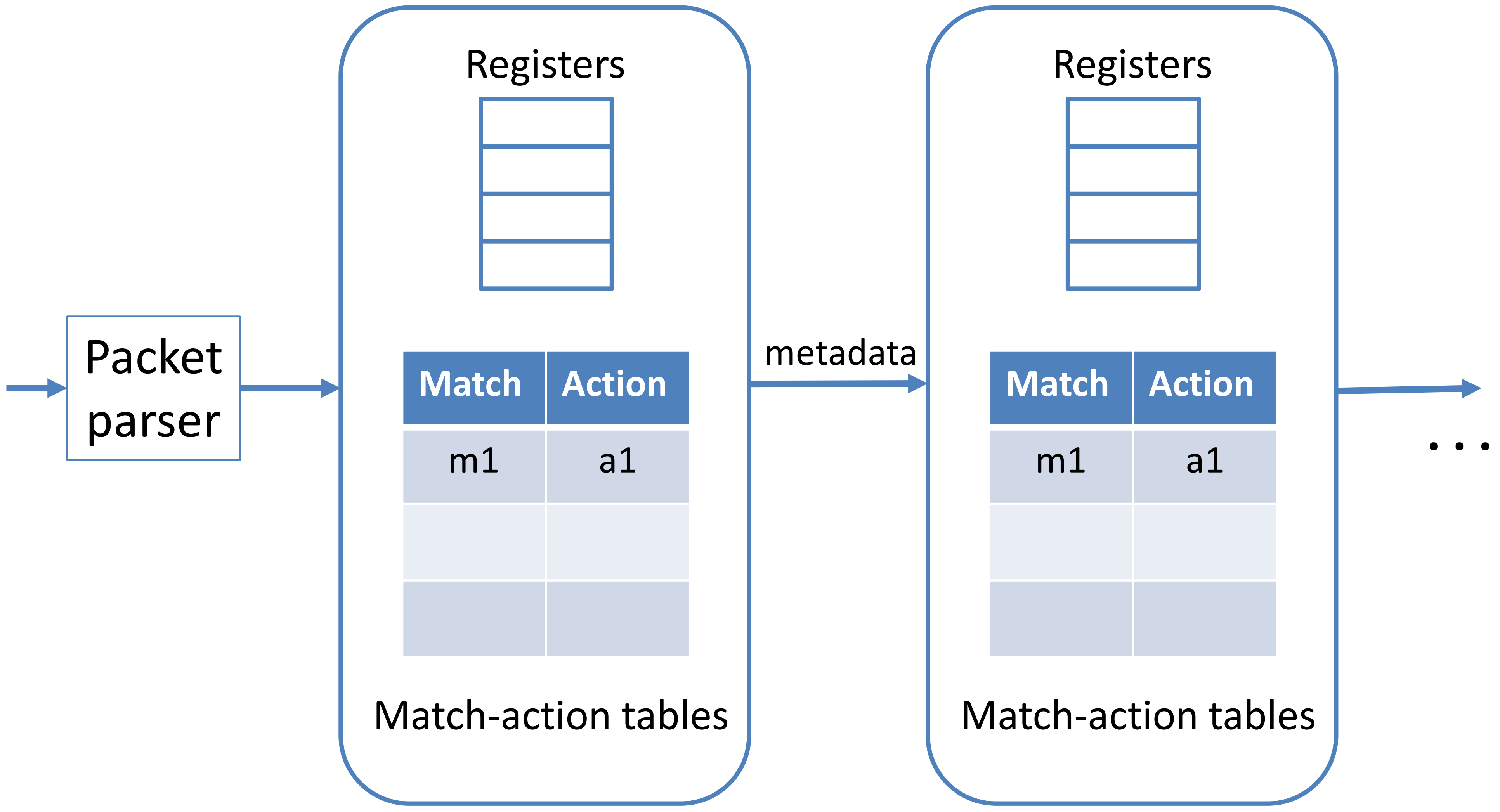
S. Muthukrishnan

Traffic Monitoring in the Data Plane

(compact data structures)

Vibhaalakshmi Sivaraman, Srinivas Narayana, Ori Rottenstreich, S. Muthukrishnan, and Jennifer Rexford, "Heavy-hitter detection entirely in the data plane," in *ACM Symposium on SDN Research*, April 2017.

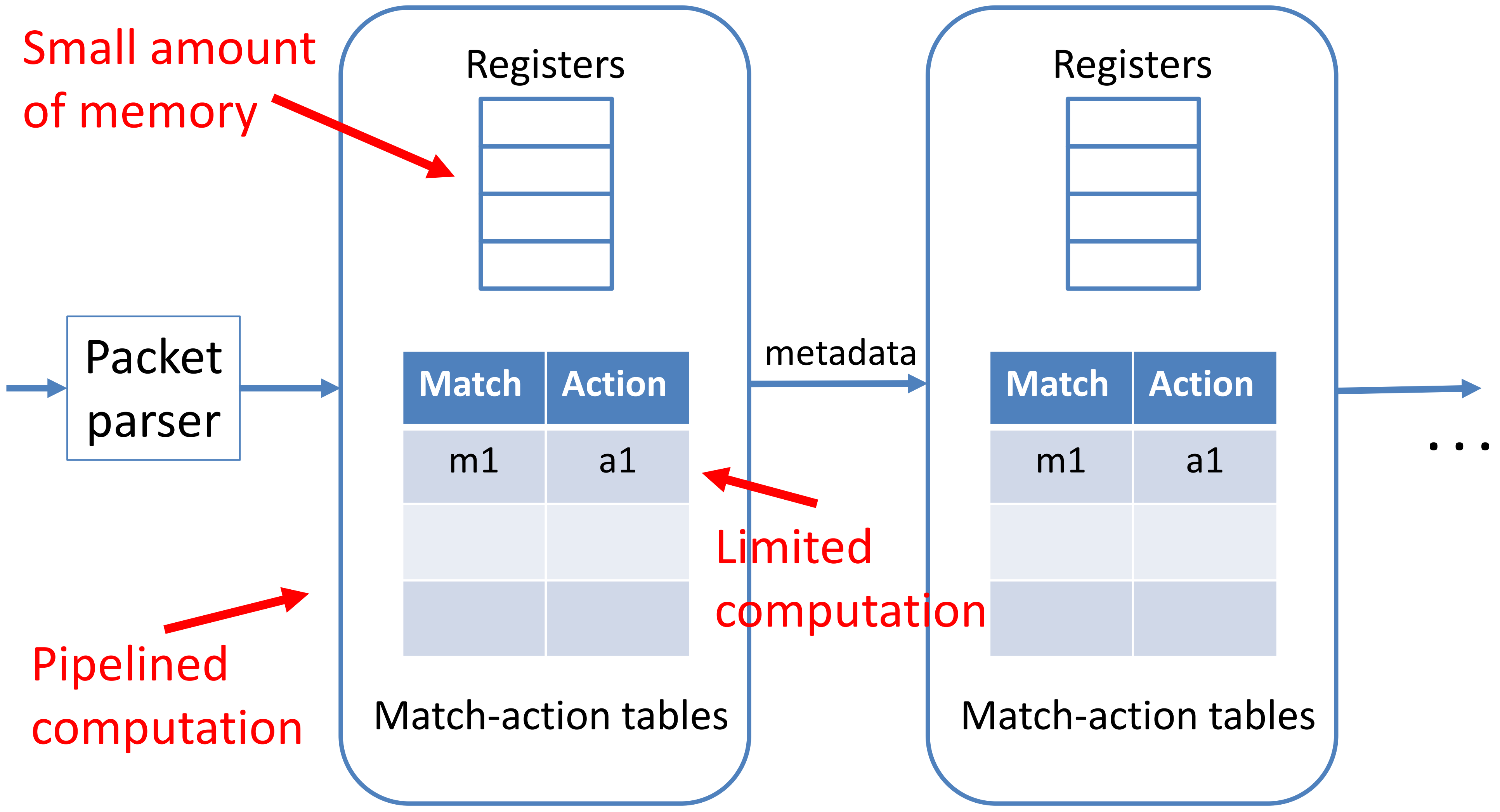
Programmable Packet Processing Hardware



Traffic Analysis in the Data Plane

- Streaming algorithms
  - Analyze traffic data
  - … directly as packets go by
  - A rich theory literature!
- A great opportunity
  - Heavy-hitter flows
  - Denial-of-service attacks
  - Performance problems
  - …

A Constrained Computational Model



Example: Heavy-Hitter Detection

- Heavy hitters
  - The *k* largest trafic flows
  - Flows exceeding threshold *T*
- Space-saving algorithm
  - Table of (key, value) pairs
  - Evict the key with the minimum value

New Key K7

Id	Count
K1	4
K2	2
K3	7
K4	10
K5	1
K6	5

Table scan

Approximating the Approximation

- Evict minimum of *d* entries
  - Rather than minimum of all entries
  - E.g., with *d = 2* hash functions

New
Key K7

Id	Count
K1	4
K2	2
K3	7
K4	10
K5	1
K6	5

Multiple memory accesses

Approximating the Approximation

- Divide the table over *d* stages
  - One memory access per stage
  - Two different hash functions

New
Key K7

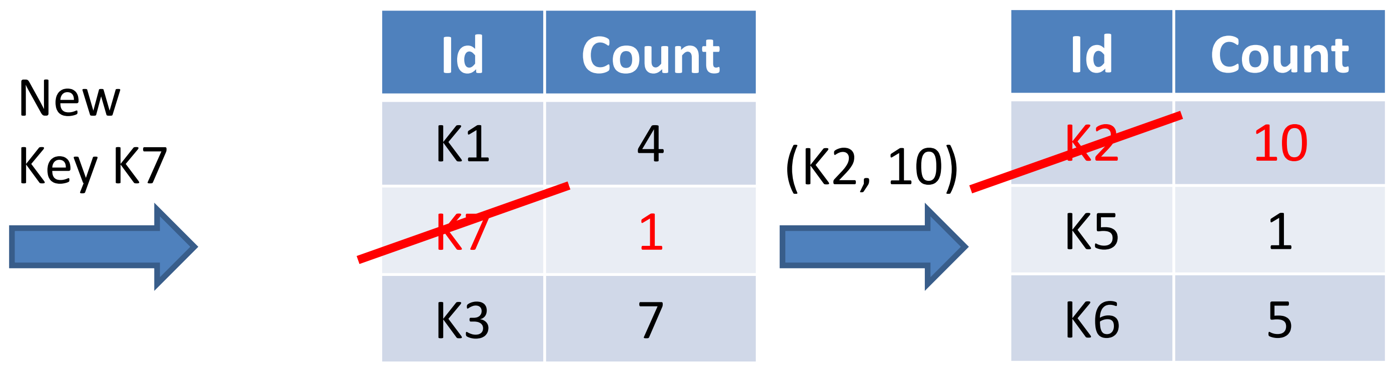
Id	Count
K1	4
K2	2
K3	7

Id	Count
K4	10
K5	1
K6	5

Going back to the first table

Approximating the Approximation

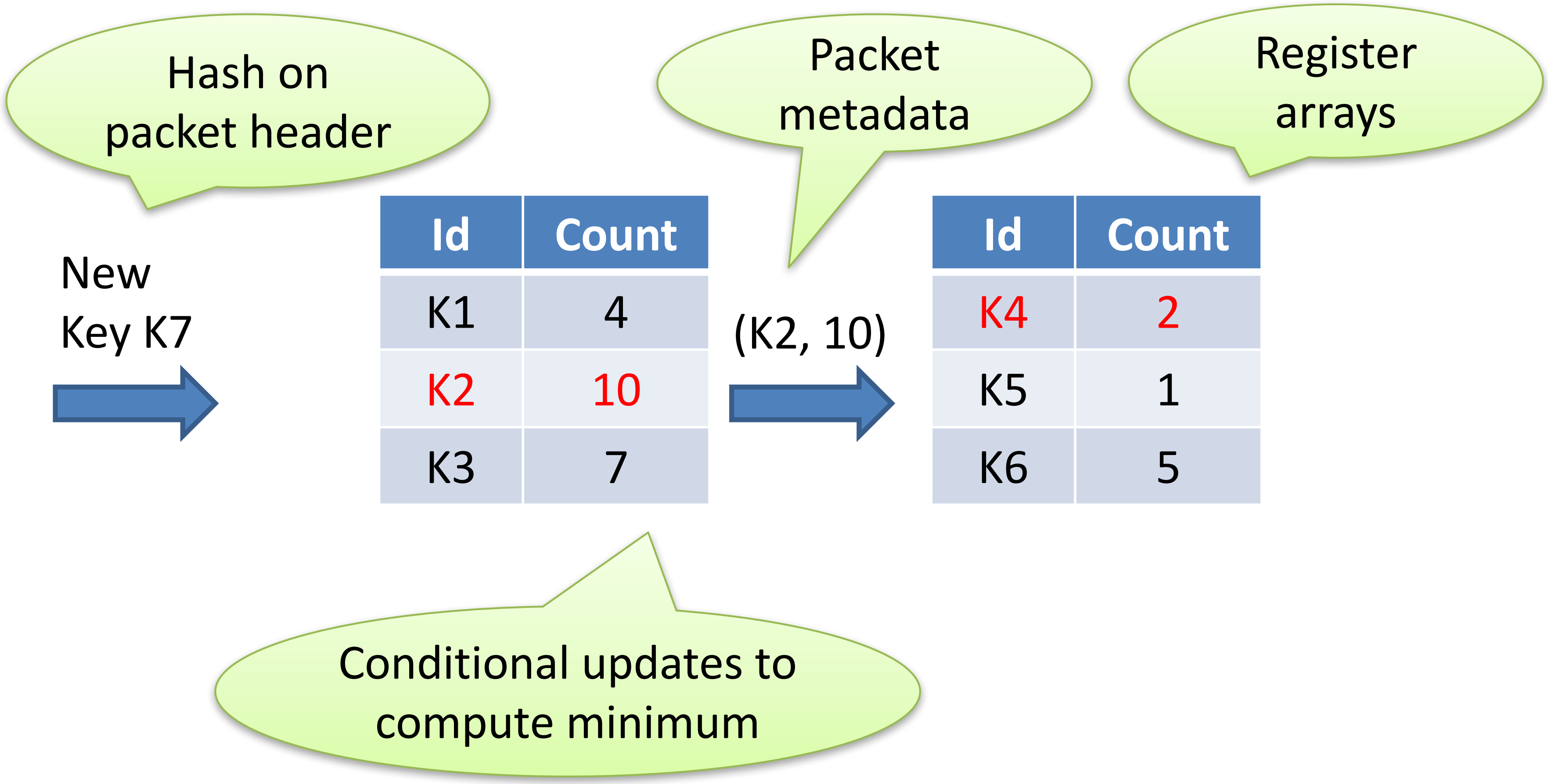
- Rolling min across stages
  - Avoid recirculating the packet
  - … by carrying the minimum along the pipeline



Id	Count
K1	4
~~K7~~	1
K3	7

Id	Count
~~K2~~	10
K5	1
K6	5

P4 Prototype and Evaluation



Id	Count
K1	4
K2	10
K3	7

Id	Count
K4	2
K5	1
K6	5

High accuracy with overhead proportional to # of heavy hitters

Lessons: Concrete vs. Abstract

- Getting concrete
  - Computational model
  - Example problem (e.g., heavy hitters)
  - Strawman solution (e.g., space saving)
- Iteratively designing
  - Relaxing the strawman solution
- Striving for general understanding
  - Provable bounds on accuracy vs. overhead
  - Ways to approach other analysis questions

“If I Had a Hammer…”

Should we all do interdisciplinary research?

Interdisciplinary Fun

- Intellectually exciting
  - Learn about other areas
  - Learn to think in new ways
  - Articulate your own field to others
- Striving for big impact
  - Create novel solutions in your field
  - Bring new problems to another field
  - Real problems are often interdisciplinary
- Socially fun
  - In-depth conversations
  - Not always about work ☺

Managing Risks

- One vs. many hammers
  - I've been an opportunistic collaborator
  - But, mastering *one* hammer is sometimes better

- Honing one hammer
  - Controlling your fate
  - Great grad school or postdoc adventure
  - Pick *hammer* wisely!

- Work by collaboration
  - Riskier for junior folks
  - Okay if you dig deep in your own field
  - Pick *nails* wisely!

Managing Risks

- Steep learning curve
  - Learning both the hammer and the nail
  - Learning the culture of both research fields

- Start with the hammer
  - Easier for a PL person to learn (part of) networking
  - … than for a networking person to learn PL
- Join an emerging community
  - Technical foundations and collaborators
  - Example projects and publications
  - But, don’t join *too late* in the game

Managing Risk

- Credit for the work
  - Publications in different field’s venues
  - Multiple authors

- Favoring publication venues for junior author
- Understanding institution’s evaluation process
- Can be *easier* to tell who’s “fingerprints” are on which parts of the work

Managing Risks

- Healthy interdisciplinary collaborations
  - Collaborators who stay engaged
  - … and truly “dig in” to the work

- Work that is exciting in both fields
- Similar value structures (e.g., proofs, code)
- Engaging students and postdocs
- Physical proximity

Conclusion

- Computer networking
  - Important real-world challenges
  - Intellectually rich space of problems
- Networks worthy of society's trust
  - Grand challenge across a range of fields
  - … if we can reach across the divide