Adriaan van Wijngaarden meets Scott

Domain-Theoretic Foundations for Probabilistic Network Programming

Alexandra Silva (UCL)

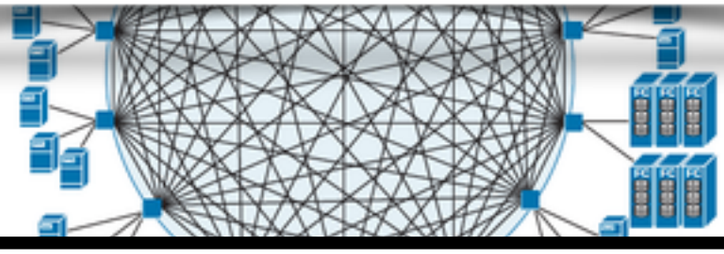
The gang


Integrated Photonics
Integration
Integrated Switching
Optics


NFA
DFA
FBL
R
L

Networks

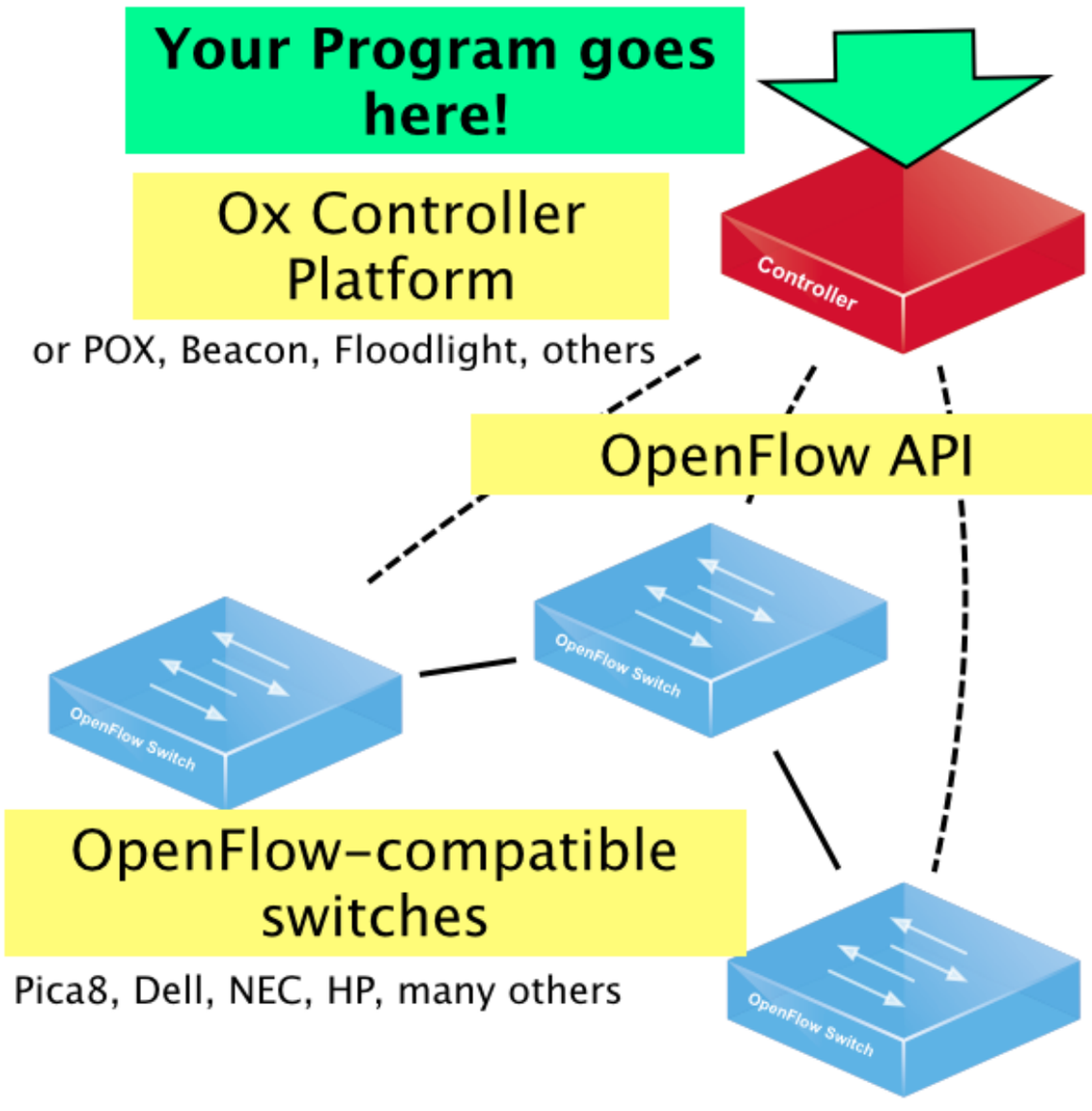
Network configuration largely a black art



- built and programmed the same way
- configured locally using proprietary interfaces

- ✓ Difficult to implement end-to-end routing policies and optimisations that require a global perspective
- ✓ Difficult to extend with new functionality
- ✓ Effectively impossible to reason precisely about behaviour

Software-Defined Networks



Openflow

[McKeown & al., SIGCOMM 08]

- Specifies capabilities and behaviour of switch hardware
- A language for manipulating network configurations

- Very low-level: easy for hardware to implement, difficult for humans to write and reason about

But…

✓ is platform independent
✓ provides an open standard that any vendor can implement

Verification of networks

Trend in PL&Verification after Software-Defined Networks

- Design *high-level languages* that model essential network features
- Develop *semantics* that enables reasoning precisely about behaviour
- Build *tools* to synthesise low-level implementations automatically

✤ Frenetic [Foster & al., ICFP 11]
✤ Pyretic [Monsanto & al., NSDI 13]
✤ Maple [Voellmy & al., SIGCOMM 13]
✤ FlowLog [Nelson & al., NSDI 14]
✤ Header Space Analysis [Kazemian & al., NSDI 12]
✤ VeriFlow [Khurshid & al., NSDI 13]
✤ NetKAT [Anderson & al., POPL 14]
✤ and many others . . .

NetKAT

NetKAT

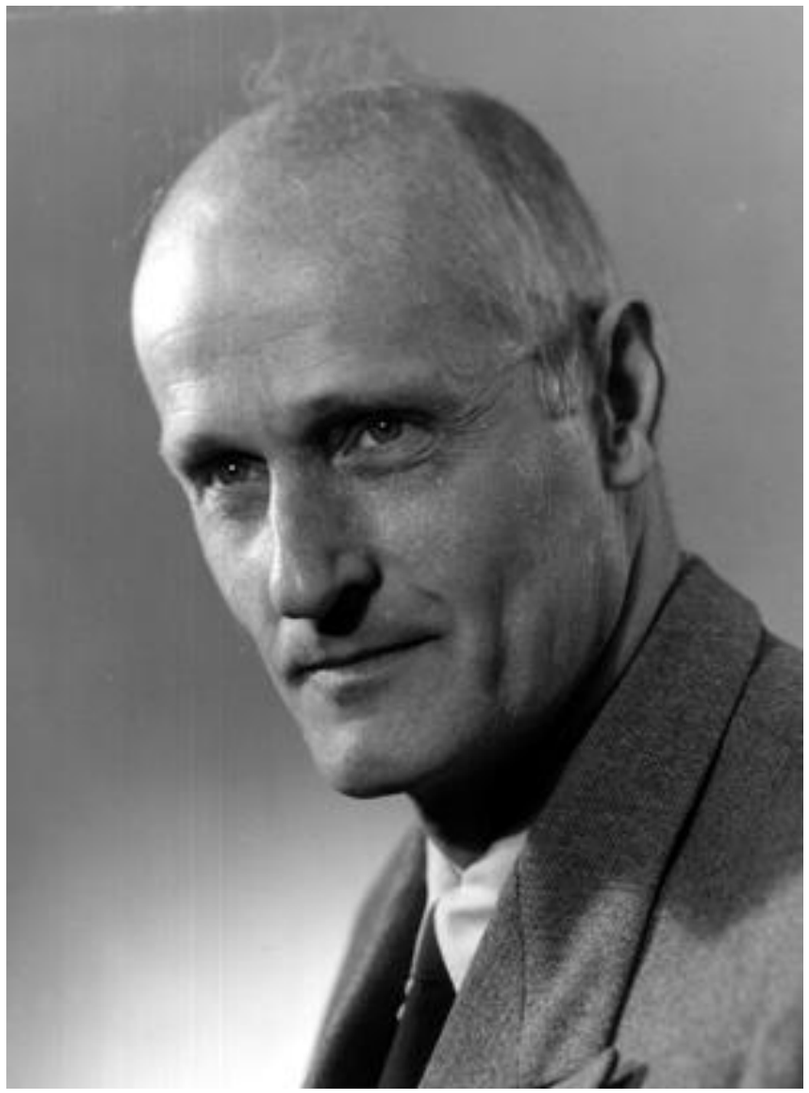
=

Kleene algebra with tests (KAT)

+

additional specialized constructs particular to
network topology and packet switching

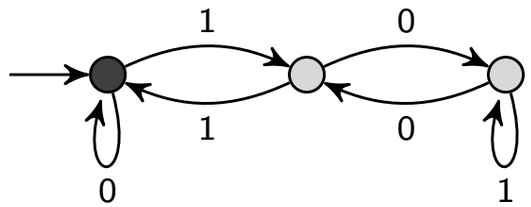
Net**KA**T



Stephen Cole Kleene
(1909–1994)

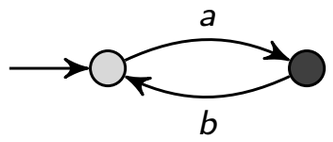
$(0 + 1(01^*0)^*1)^*$

{multiples of 3 in binary}



$(ab)^*a = a(ba)^*$

$\{a, aba, ababa, \ldots\}$



$(a + b)^* = a^*(ba^*)^*$

{all strings over $\{a, b\}$}

Net**KAT**

$(K, B, +, \cdot, ^*, ^-, 0, 1), \quad B \subseteq K$

- $(K, +, \cdot, ^*, 0, 1)$ is a Kleene algebra
- $(B, +, \cdot, ^-, 0, 1)$ is a Boolean algebra
- $(B, +, \cdot,$
- $p, q, r, .$
- $a, b, c, .$

KAT = simple imperative language

If b **then** p **else** q = b;p + !b;q

While b **do** p = (bp)*!b

NetKAT

- a packet π is an assignment of constant values n to fields x
- a packet history is a nonempty sequence of packets
  $\pi_1 :: \pi_2 :: \cdots :: \pi_k$
- the head packet is π_1

NetKAT

- assignments $x \leftarrow n$
  assign constant value n to field x in the head packet
- tests $x = n$
  if value of field x in the head packet is n, then pass, else drop
- dup
  duplicate the head packet

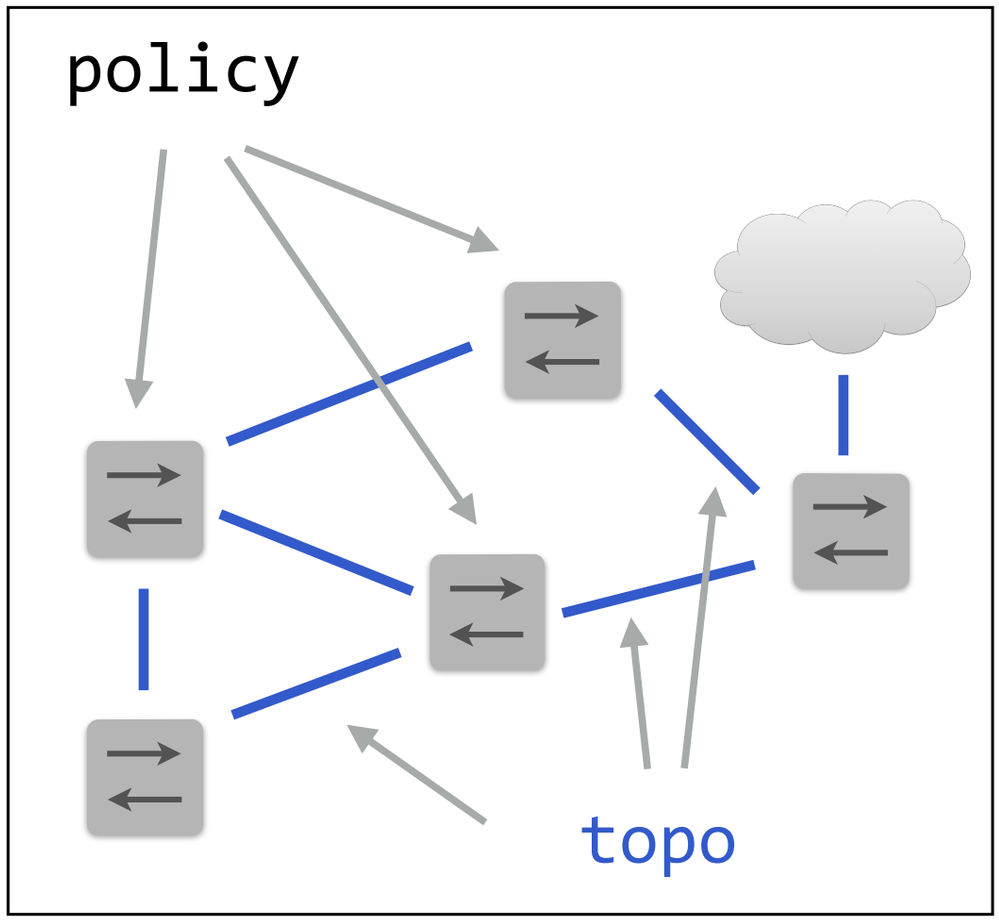
Networks in NetKAT

sw=6;pt=8;dst := 10.0.1.5;pt:=5

For all packets located at port 8 of switch 6, set the destination address to 10.0.1.5 and forward it out on port 5.

Networks in NetKAT

The behaviour of an entire network can be encoded in NetKAT by interleaving steps of processions by switches and topology



policy

+

(policy**;** topo**;** policy

+

(policy**;** topo**;** policy**;** topo**;** policy

⋮

(policy**;** topo**)***; policy

Semantics

$$[\![e]\!] : H \to 2^H$$

$$[\![x \leftarrow n]\!](\pi_1 :: \sigma) \triangleq \{\pi_1[n/x] :: \sigma\}$$

$$[\![x = n]\!](\pi_1 :: \sigma) \triangleq \begin{cases} \{\pi_1 :: \sigma\} & \text{if } \pi_1(x) = n \\ \varnothing & \text{if } \pi_1(x) \neq n \end{cases}$$

$$[\![\mathsf{dup}]\!](\pi_1 :: \sigma) \triangleq \{\pi_1 :: \pi_1 :: \sigma\}$$

Verification using NetKAT

Reachability

- Can host A communicate with host B? Can every host communicate with every other host?

Security

- Does all untrusted traffic pass through the intrusion detection system located at C?

Loop detection

- Is it possible for a packet to be forwarded around a cycle in the network?

Verification using NetKAT

Soundness and Completeness [Anderson et al. 14]

- $\vdash p = q$ if and only if $[\![p]\!] = [\![q]\!]$

Decision Procedure [Foster et al. 15]

- NetKAT coalgebra
- efficient bisimulation-based decision procedure
- implementation in OCaml
- deployed in the Frenetic suite of network management tools

Limitations

$$[\![e]\!] : H \to 2^H$$

✳Packet-processing **function**

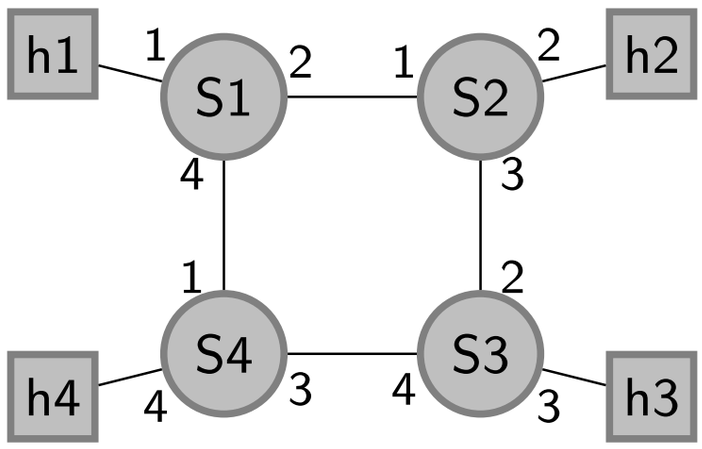
✳Applicability limited to simple connectivity or routing behavior

Probabilities are needed

- ✳ expected congestion
- ✳ reliability
- ✳ randomized routing

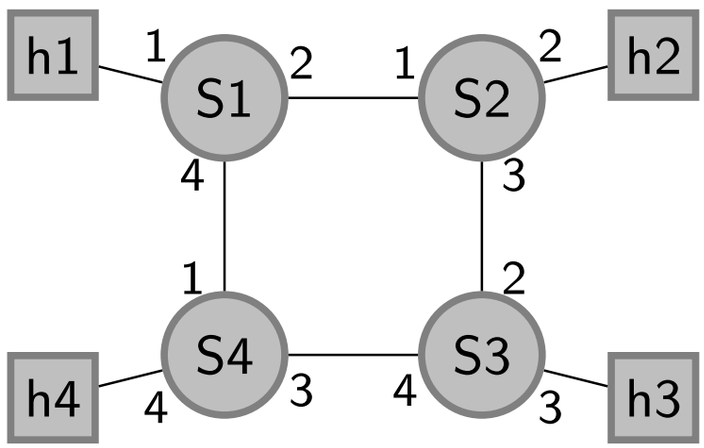
ProbNetKAT

$$p \bigoplus_r q$$



$$\mathsf{dst} = \mathsf{h}_3; \mathsf{pt} \leftarrow 2 \bigoplus_{.5} \mathsf{pt} \leftarrow 4$$

ProbNetKAT by example



$$net \triangleq in \,;\, (p \,;\, t)^* \,;\, p \,;\, out$$

$$l_{1,2} \triangleq (\mathsf{sw}{=}S_1 \,;\, \mathsf{pt}{=}2 \,;\, \mathsf{dup} \,;\, \mathsf{sw}{\leftarrow}S_2 \,;\, \mathsf{pt}{\leftarrow}1 \,;\, \mathsf{dup})$$

Ingress - egress

$$p_1 \triangleq (\mathsf{dst}{=}h_1 \,;\, \mathsf{pt}{\leftarrow}1)$$
$$\& \; (\mathsf{dst}{=}h_2 \,;\, \mathsf{pt}{\leftarrow}2)$$
$$in \triangleq (\mathsf{sw}{=}1 \,;\, \mathsf{pt}{=}1) \;\&\; (\mathsf{sw}{=}2 \,;\, \mathsf{pt}{=}2) \;\&\; \ldots$$
$$out \triangleq (\mathsf{dst}{=}h_3 \,;\, \mathsf{pt}{\leftarrow}3) \;\&\; (\mathsf{dst}{=}h_4 \,;\, \mathsf{pt}{\leftarrow}4)$$

Forwarding policy

$$p \triangleq (\mathsf{sw}{=}S_1 \,;\, p_1) \,\&\, (\mathsf{sw}{=}S_2 \,;\, p_2) \,\&\, (\mathsf{sw}{=}S_3 \,;\, p_3) \,\&\, (\mathsf{sw}{=}S_4 \,;\, p_4)$$

Topology

$$t \triangleq l_{1,2} \;\&\; l_{2,3} \;\&\; l_{3,4} \;\&\; l_{1,4}$$

Semantics

$$[\![p]\!] \in 2^{\mathsf{H}} \to \{\mu : \mathcal{B} \to [0,1] \mid \mu \text{ is a probability measure}\}$$

$\mathcal{B}$ Borel sets of 2^{H}

using Cantor topology

$$[\![x \leftarrow n]\!](a) = \delta_{\{\pi[n/x] : \sigma \mid \pi : \sigma \in a\}}$$

$$[\![x = n]\!](a) = \delta_{\{\pi : \sigma \mid \pi : \sigma \in a,\ \pi(x) = n\}}$$

$$[\![\texttt{dup}]\!](a) = \delta_{\{\pi : \pi : \sigma \mid \pi : \sigma \in a\}}$$

$$[\![\texttt{skip}]\!](a) = \delta_a$$

$$[\![\texttt{drop}]\!](a) = \delta_\varnothing$$

$$[\![p \mathbin{\&} q]\!](a) = [\![p]\!](a) \mathbin{\&} [\![q]\!](a)$$

$$(\mu \mathbin{\&} \nu)(A) \triangleq (\mu \times \nu)(\{(a,b) \mid a \cup b \in A\}).$$

$$[\![p +_r q]\!](a) = r[\![p]\!](a) + (1-r)[\![p]\!](a)$$

Semantics

$$[\![p]\!] \in 2^{\mathsf{H}} \to \{\mu : \mathcal{B} \to [0,1] \mid \mu \text{ is a probability measure}\}$$

$$[\![p^*]\!] =?$$

Ideally: $[\![p^*]\!] = [\![1 \& pp^*]\!]$

least fix point? which order?

Ad-hoc attempt: infinite stochastic process

Semantics

ProbNetKAT model **p**, input distribution **μ**

→ output distribution **ν** = μ >>= ⟦p⟧ ∈ Dist(2^H)

Congestion Query: Random Variable **Q** : 2^H → [0,∞]

$$Q(a) \triangleq \sum_{h \in a} \#_l(h)$$

Expected Congestion: E_{ν}[**Q**]

$$\mathop{\mathbf{E}}_{\nu}[Q] = \int Q \, d\nu$$

Issues with previous semantics

No practical implementation?

Iteration — infinite stochastic process instead of standard fixpoint

Lebesgue Integral

weak convergence non-monotonic

continuous distribution

Challenges in representing infinite distributions

many queries not continuous Cantor topology — no weak convergence!

The importance of continuity

$a_1, a_2, \cdots$ —limit→ a

simple (e.g. finite) objects a **can be approximated by** (a_n)

Perform computation f on a f continuous

$f(a_1), f(a_2), \cdots$ —limit→ $f(a)$

The importance of continuity for network analysis

$$\mu_1, \mu_2, \cdots \longrightarrow \mu$$

finite support!

$\mathbf{E}_\mu[f]$ — expected value of a continuous map is continuous

monotonically improving sequence of approximations for performance metrics such as latency and congestion

New semantics

$$[\![p]\!] \in 2^{\mathsf{H}} \to \{\mu : \mathcal{B} \to [0,1] \mid \mu \text{ is a probability measure}\}$$

$\mathcal{B}$ Borel sets of

using Scott topology

$$[\![p^*]\!] = \mathsf{lfp}\ \ X \mapsto 1 \,\&\, [\![p]\!]; X$$

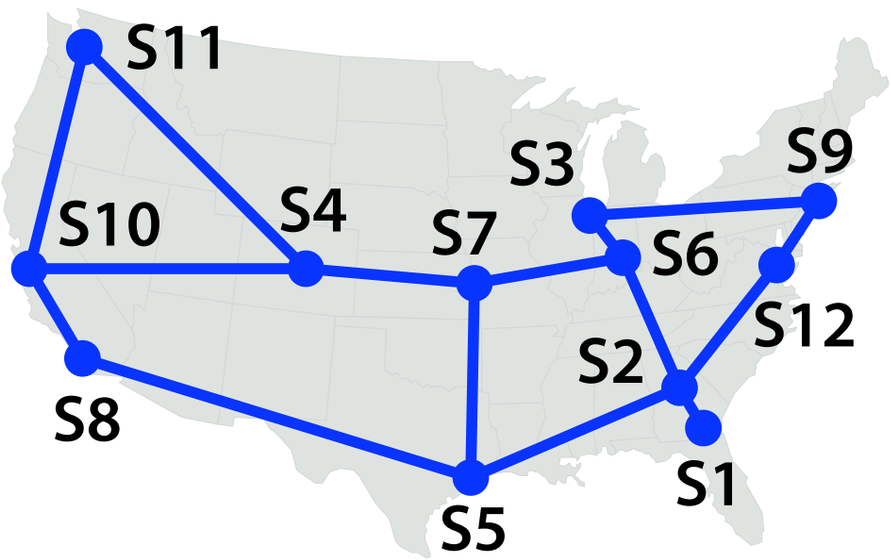
Finite approximations → Practical implementation

Implementation and Case studies

Interpreter in OCaml

Approximates the answer monotonically

Several case studies



Internet2's Abilene backbone network

Conclusions

First language-based framework for specifying and verifying **probabilistic network behavior**.

Order theoretic semantics

*van Wijngaarden would emphasize and promote the mathematical aspects of computing (…) in **design principles of programming languages**.*

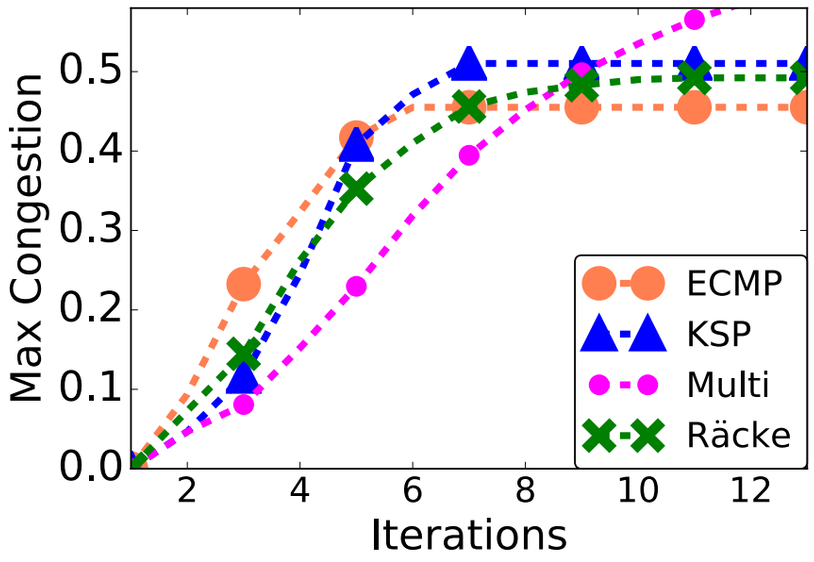
source:wikipedia

Analysis of several randomised routing protocols on real-world data

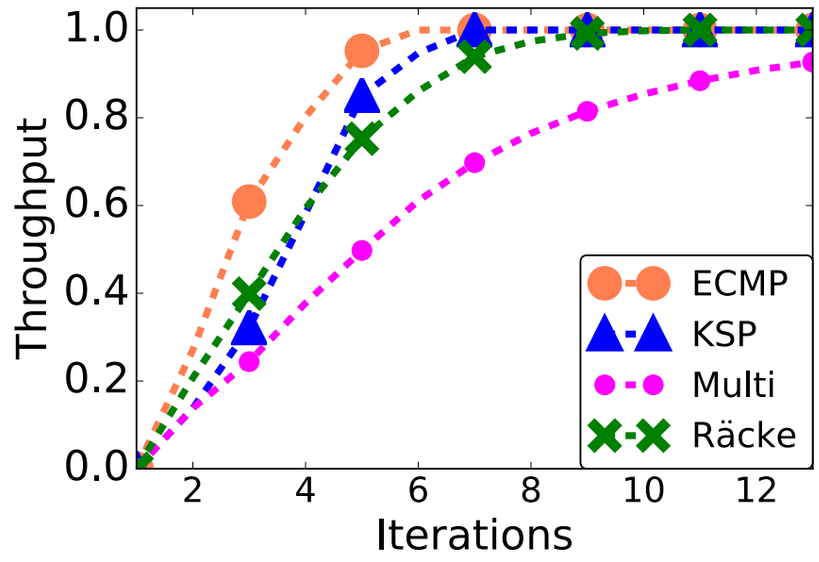
Questions?




Analysed properties



(c) Max congestion



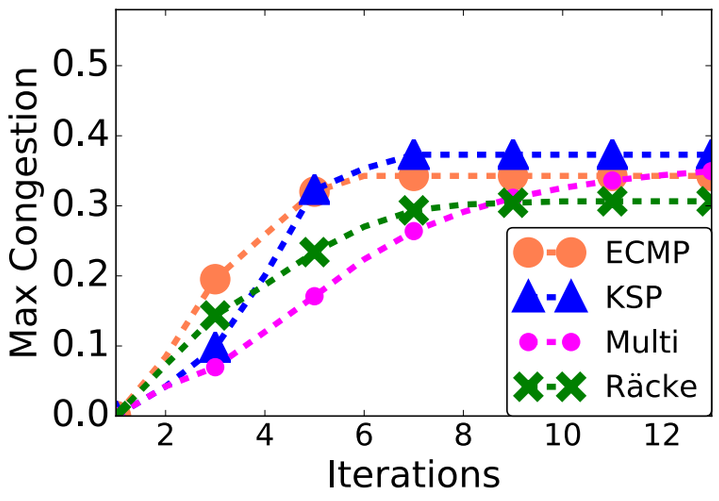
(d) Throughput

Values converge monotonically

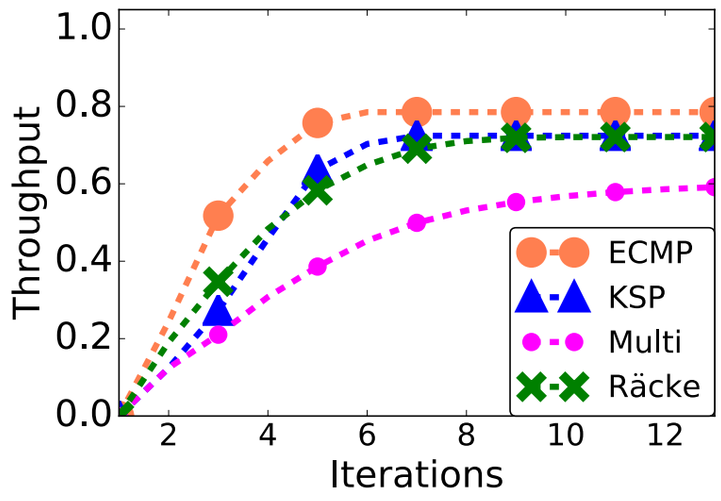
Analysed properties

Failures

$$\ell_{1,2} \triangleq \mathsf{sw}{=}S_1 \,;\, \mathsf{pt}{=}2 \,;\, \mathsf{dup} \,;\, ((\mathsf{sw}{\leftarrow}S_2 \,;\, \mathsf{pt}{\leftarrow}1 \,;\, \mathsf{dup}) \oplus_{0.9} 0)$$
$$\& \;\; \mathsf{sw}{=}S_2 \,;\, \mathsf{pt}{=}1 \,;\, \mathsf{dup} \,;\, ((\mathsf{sw}{\leftarrow}S_1 \,;\, \mathsf{pt}{\leftarrow}2 \,;\, \mathsf{dup}) \oplus_{0.9} 0)$$



(e) Max congestion



(f) Throughput

Routing

Equal Cost Multipath Routing (ECMP)

k-Shortest Paths (KSP)

Multipath Routing (Multi)

Oblivious Routing (Raecke)