Enabling SDN in old school networks with Software-Controlled Routing Protocols

Laurent Vanbever

Princeton University

ONS research track

March, 4 2014

Joint work with

Stefano Vissicchio (University of Louvain)

How do you go from a traditional network …

Traditional

How do you go from a traditional network to a SDN-enabled one?

Traditional → SDN-enabled

Well… not easily

Deploying SDN requires to upgrade network …

- devices
- management systems
- operators

challenging, time-consuming and therefore **costly**

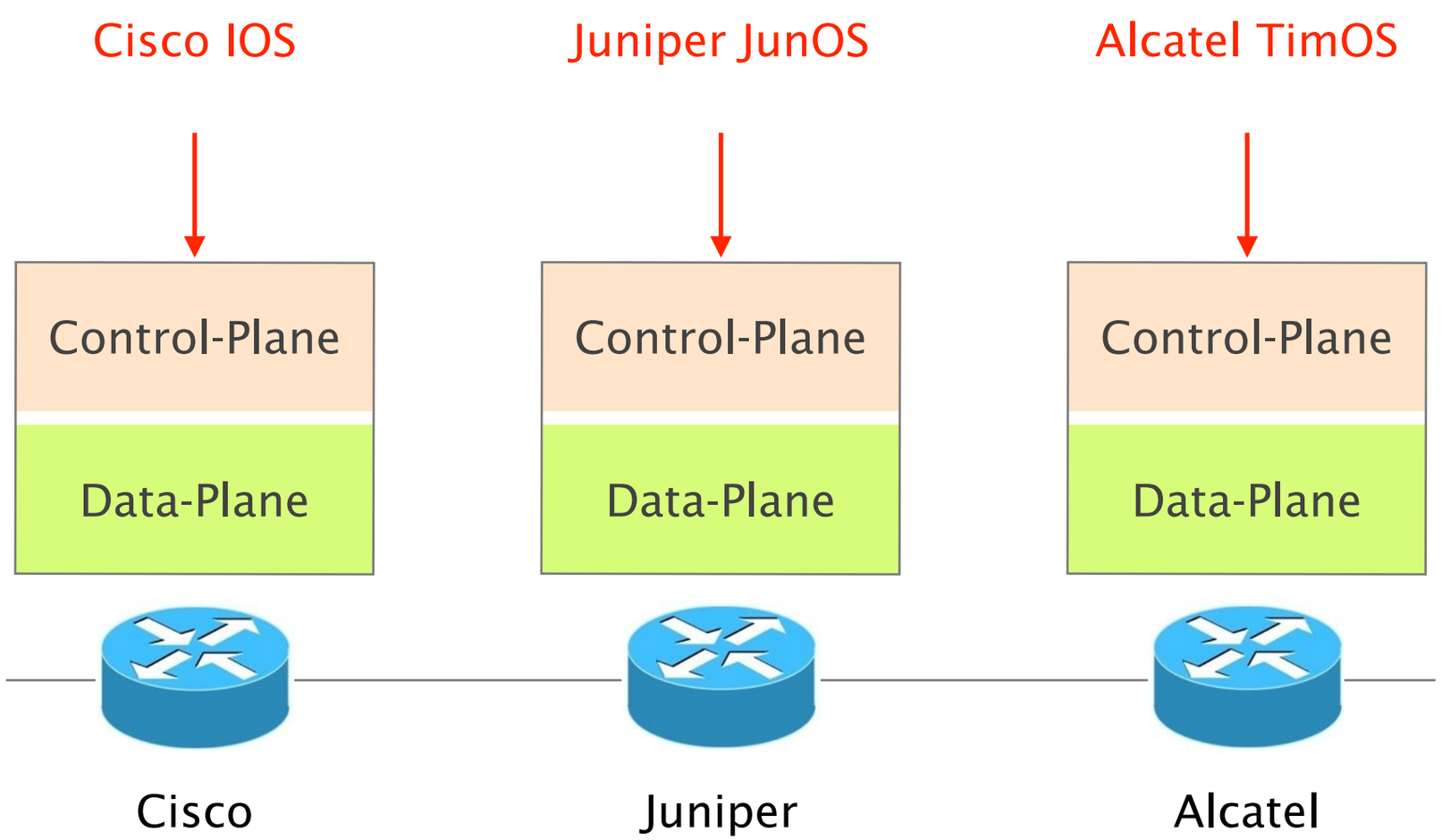
Wouldn’t it be great to manage

an existing network “à la SDN”?

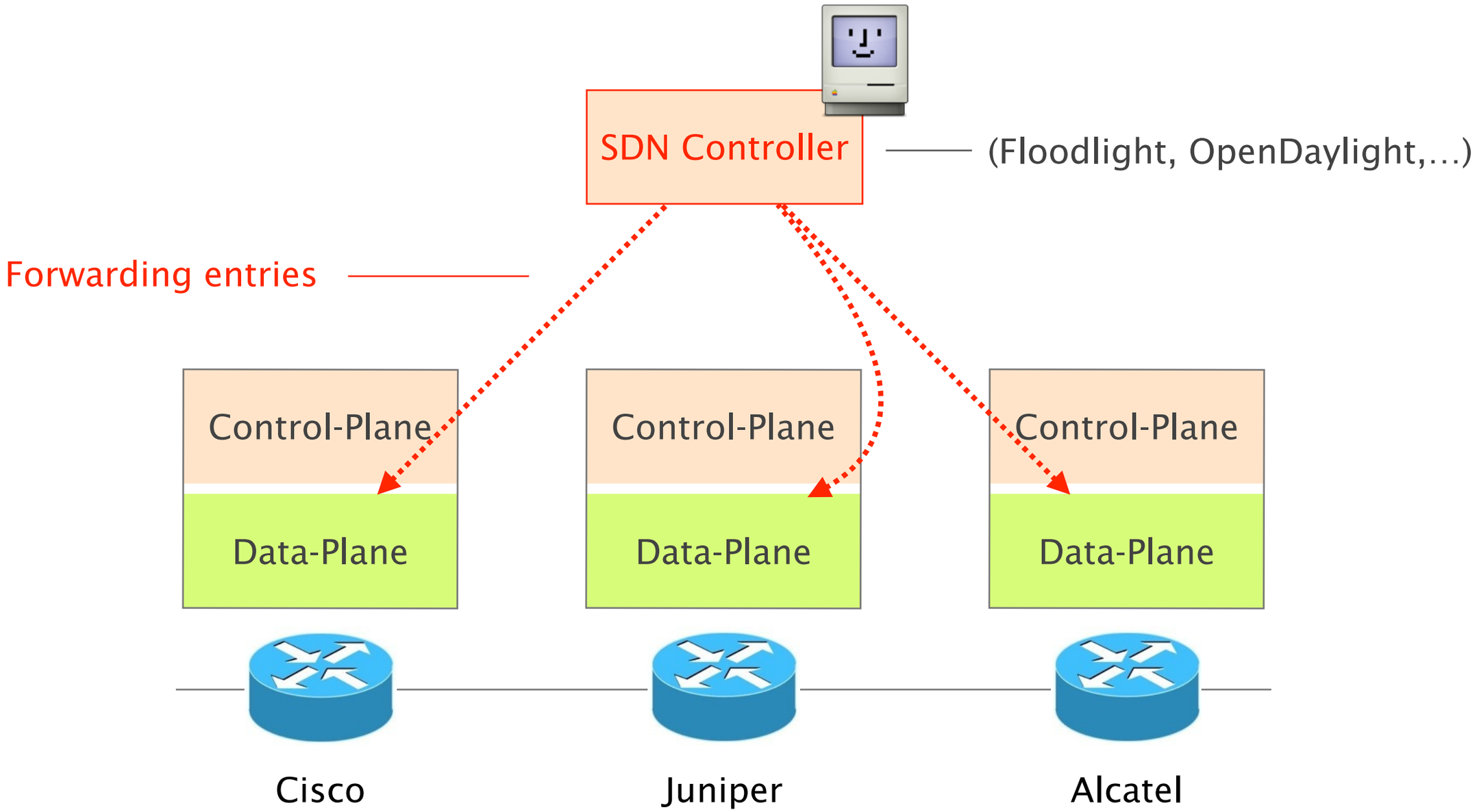
Wouldn't it be great to manage an existing network "à la SDN"?

what does it mean?

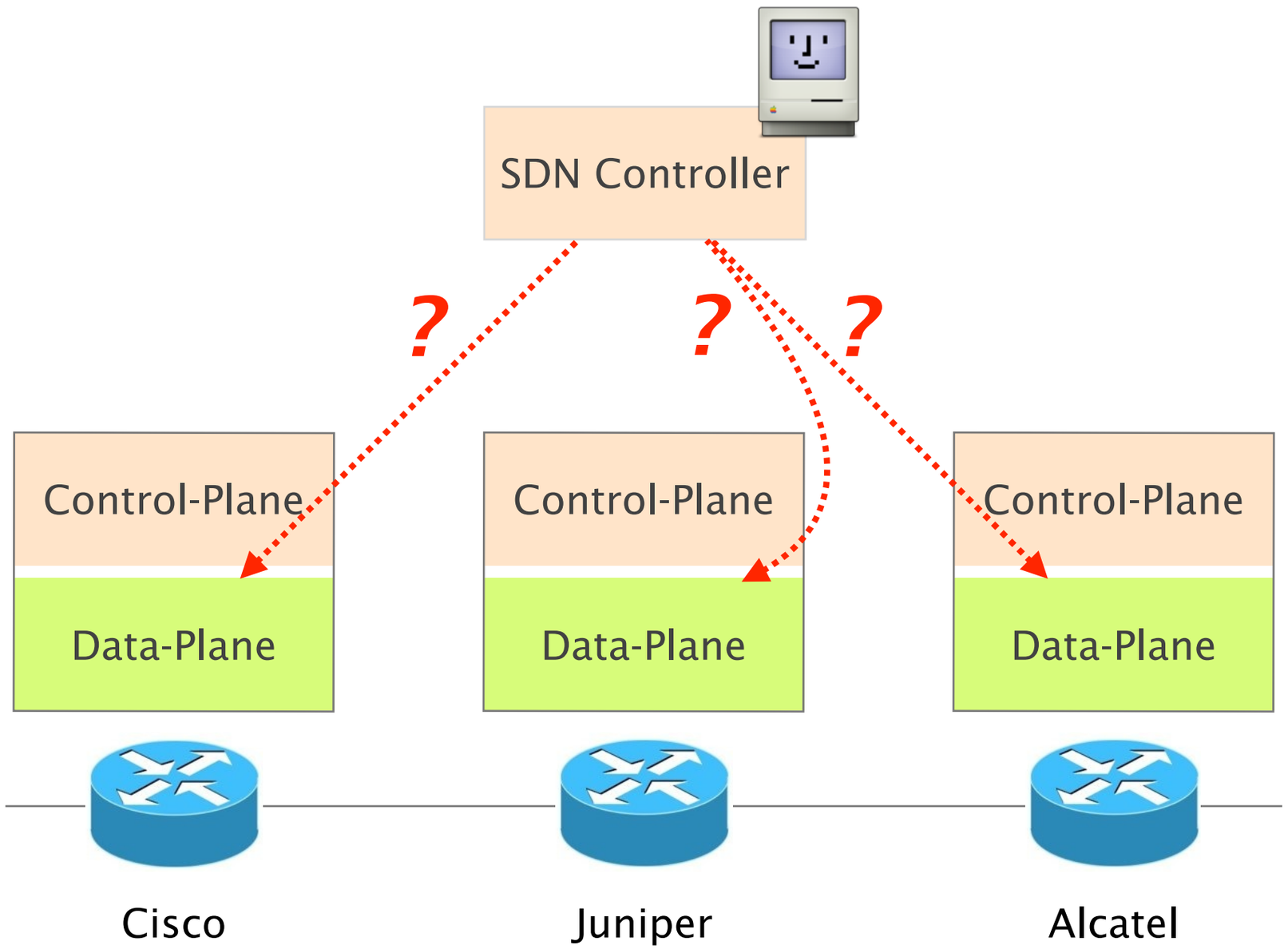
Instead of configuring your existing network using configuration “languages”…



… program it using your favorite SDN controller!

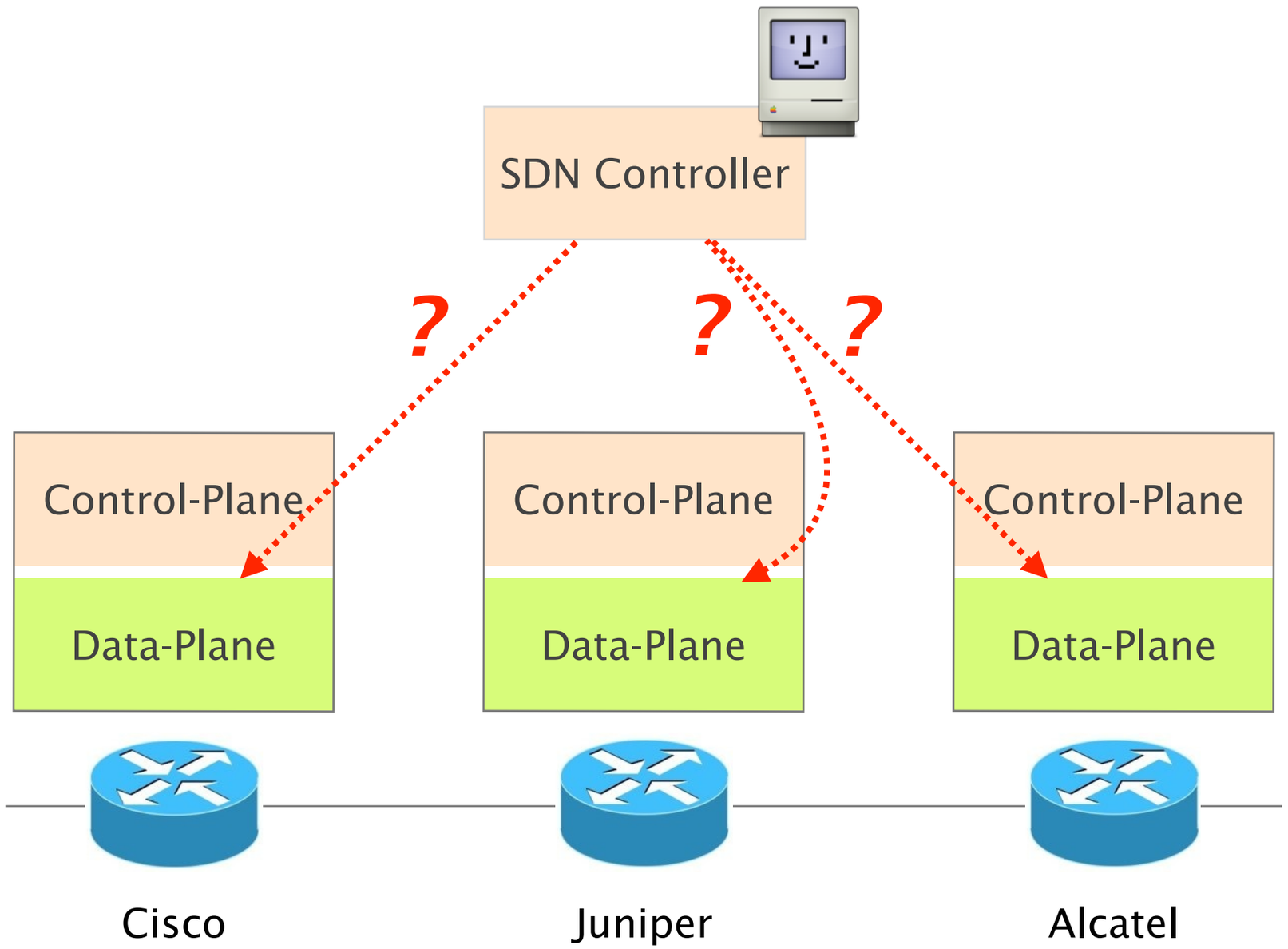


How can a SDN controller *program* forwarding entries in a router?



It uses an API that *any* router can understand

(hint: not OpenFlow)



Routing protocols are good candidates for such an API

Routing protocols…

- messages are standardized

  all routers must speak the same language

- behaviors are well-defined and understood

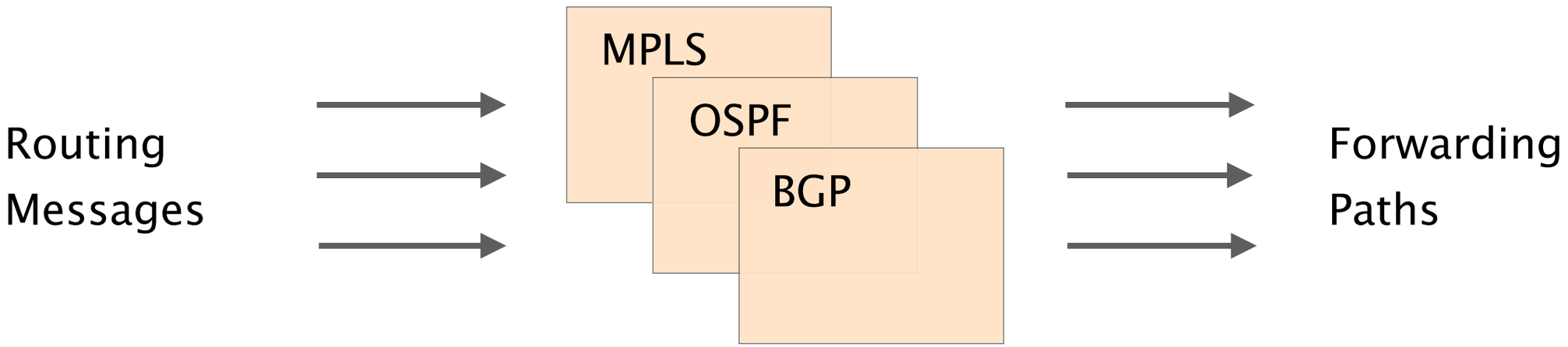
  e.g., shortest-path routing

- implementations are widely available

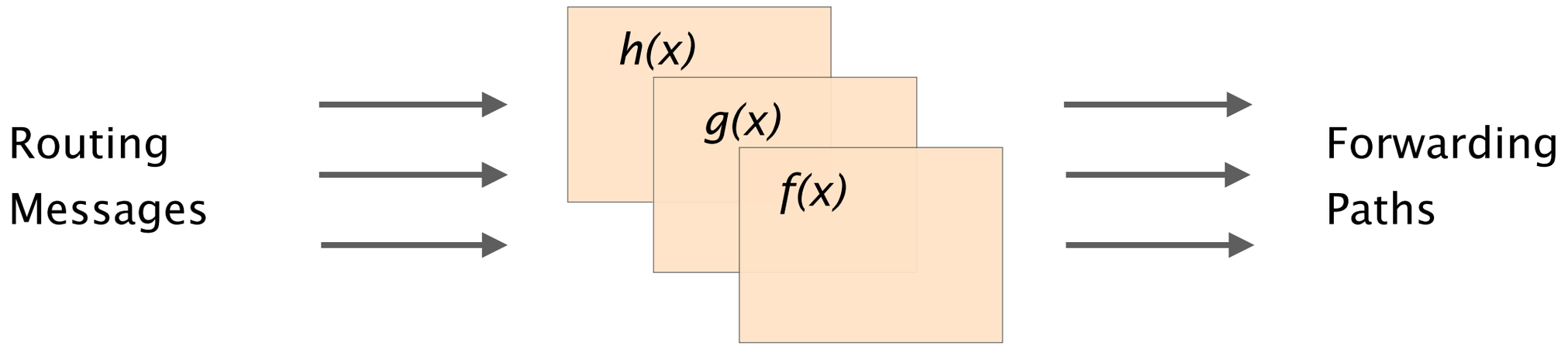
  nearly all routers out there speak OSPF

How does it work?

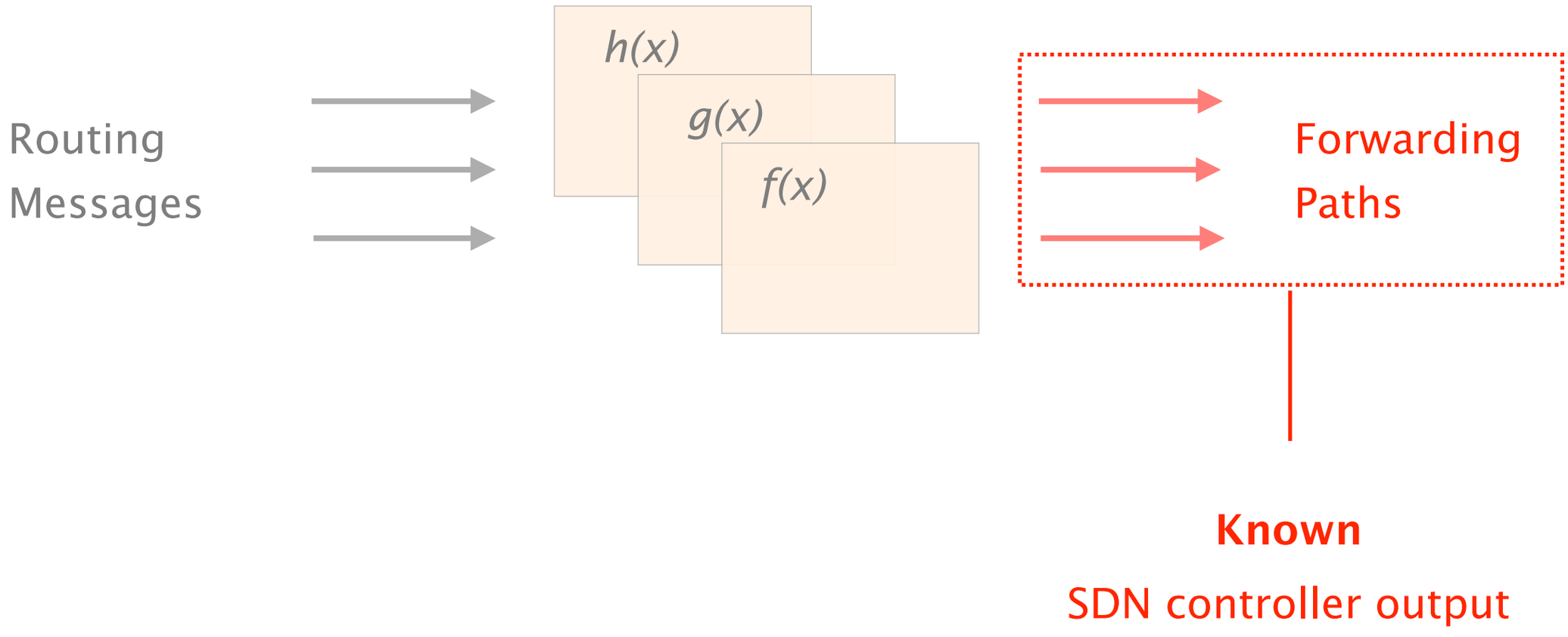
A routing protocol takes *routing messages* as input and computes *forwarding paths* as output



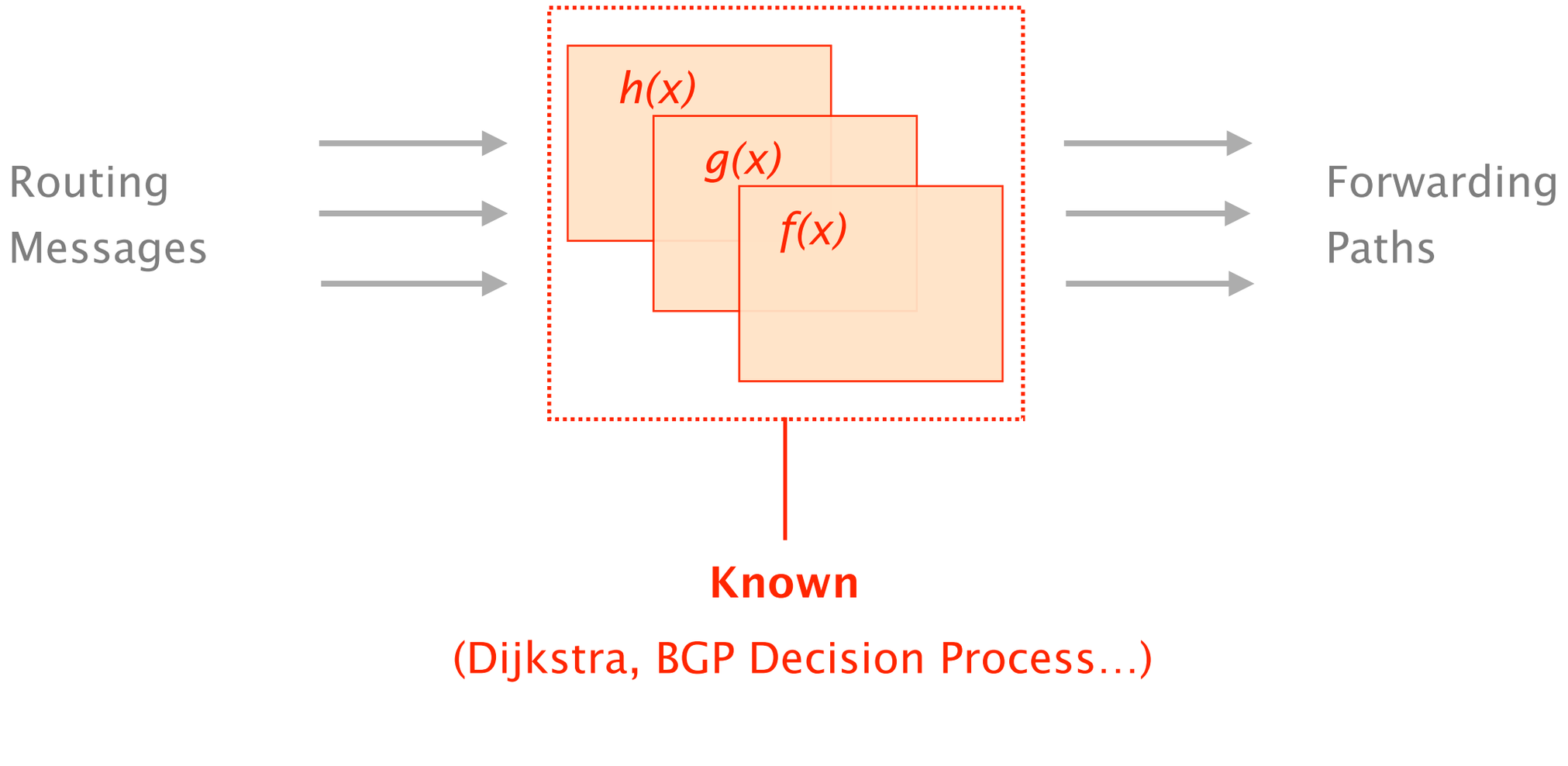
A routing protocol is therefore a function from *routing messages* to *forwarding paths*



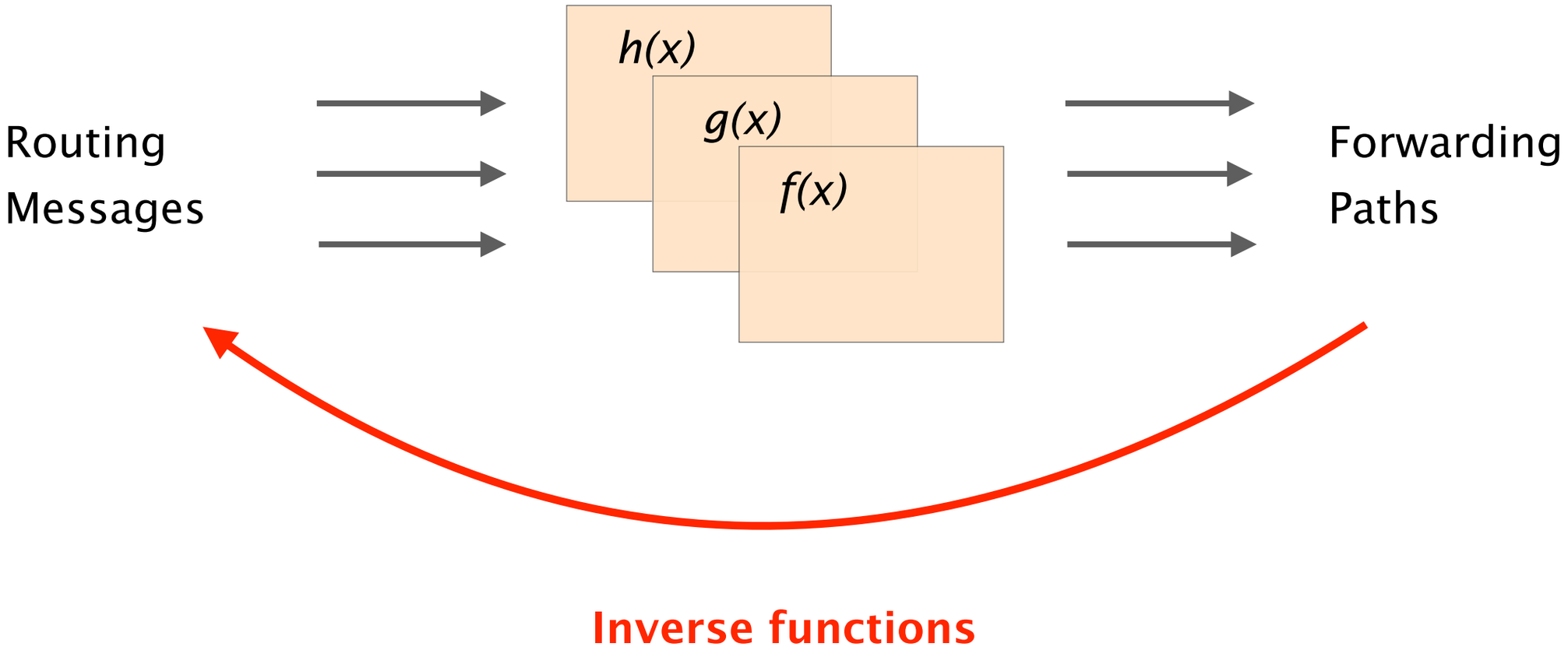
Forwarding paths are produced by the SDN controller



Functions are given by the routing protocol specification



Given a path and a function, our framework computes corresponding routing messages by inverting the function



The type of input to be computed depends on the routing protocol(s) running in the network

Type	Algorithm	Input
IGP	Link-State	Dijkstra	Network topology
BGP	Path-Vector	Decision process	Received routes

The type of input to be computed depends on the routing protocol(s) running in the network

Type	Algorithm	Input
IGP	Link-State	Dijkstra	Network topology
BGP	Path-Vector	Decision process	Received routes

Type	Algorithm	Input
IGP	**Link-State**	**Dijkstra**	**Network topology**
BGP	Path-Vector	Decision Process	Received routes

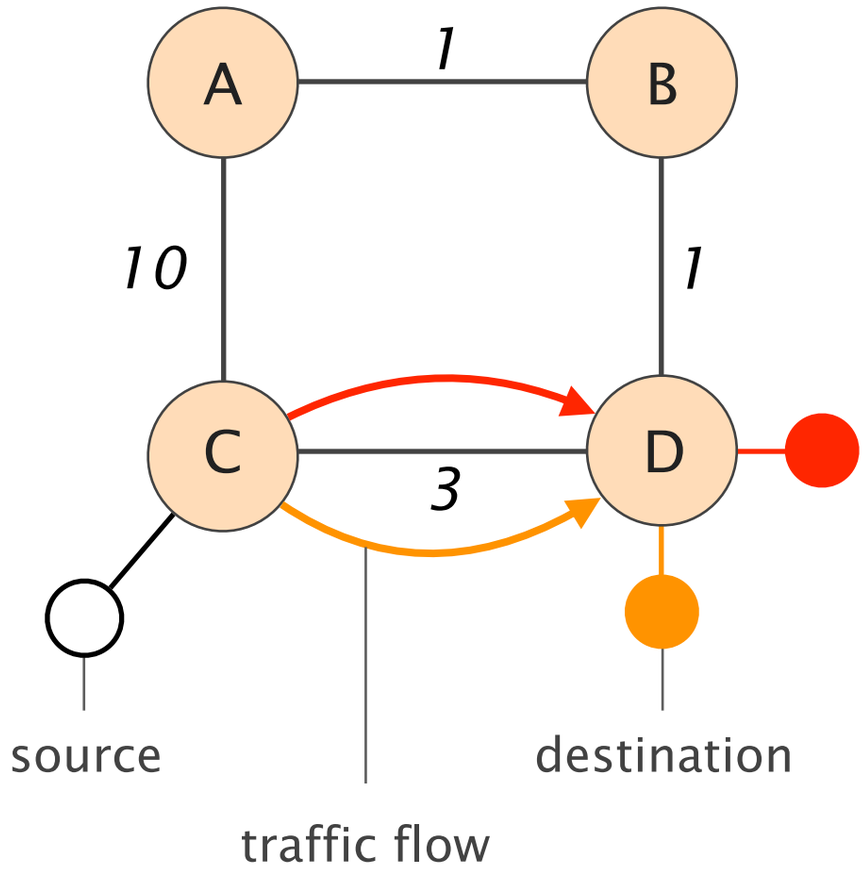
SDN-controlled IGP enables advanced SDNish TE functionalities, on top of a distributed protocol

SDN-controlled IGP enables to

- steer traffic on non-shortest paths
- create ECMP paths (on a per-destination basis)
- provision backup paths

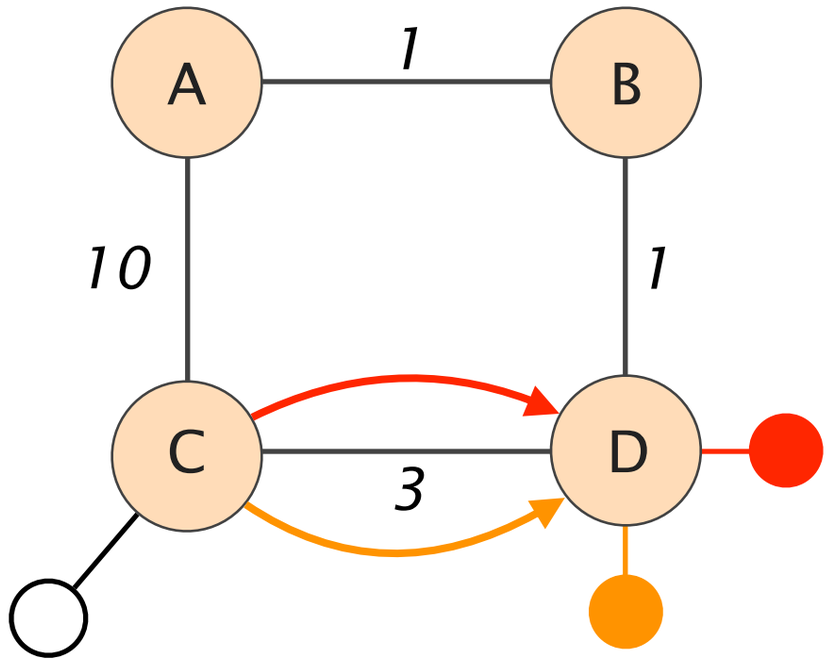
in a centralized manner, on existing networks

Consider this network where
a source sends traffic to 2 destinations

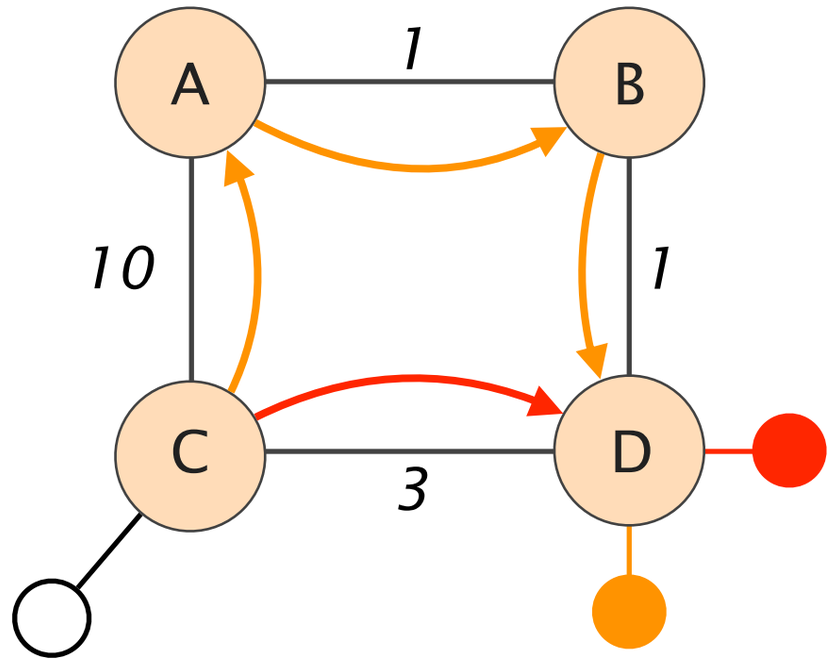


As congestion appears on the (C,D) link, operators might want to divert the orange flow to A

initial

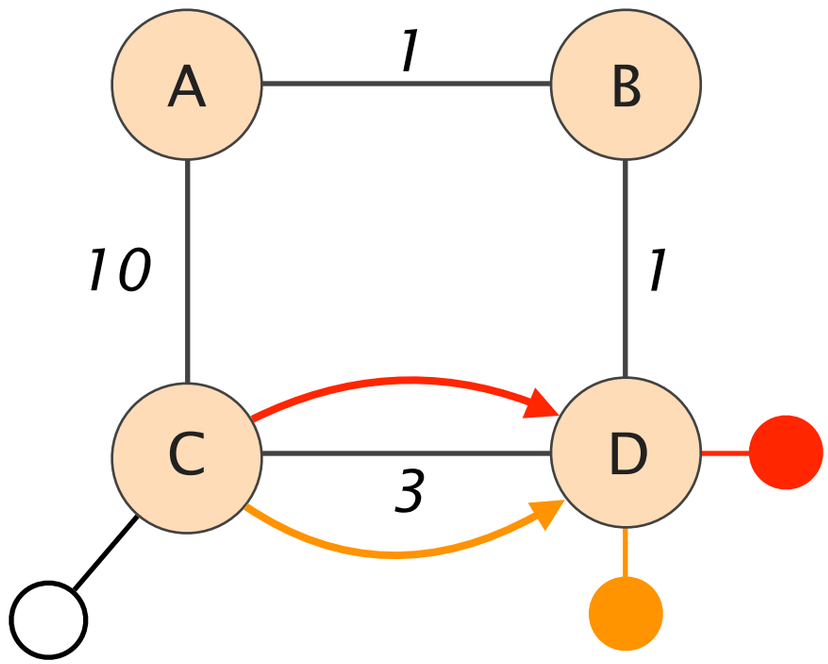


desired

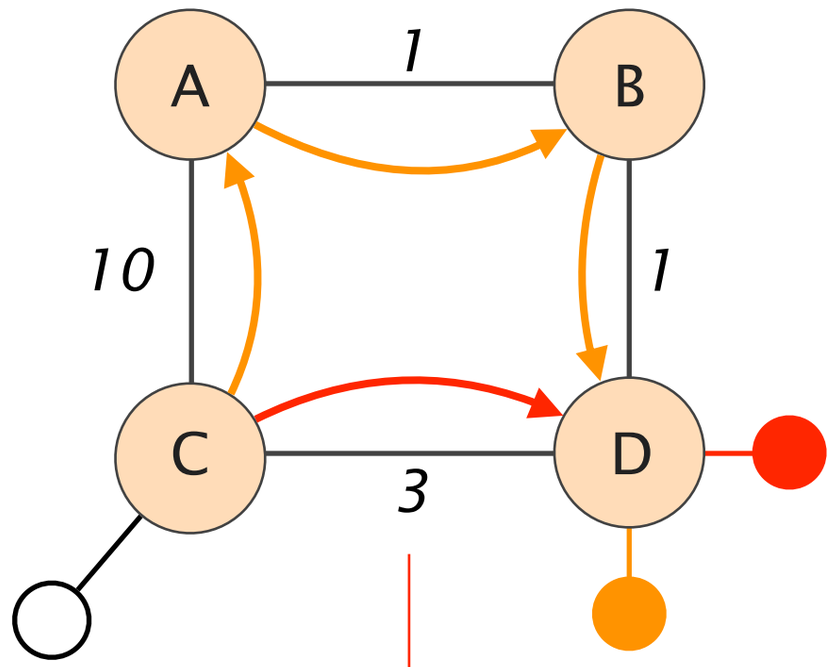


Moving only the orange flow to A is impossible
with an IGP as both destinations are connected to D

initial

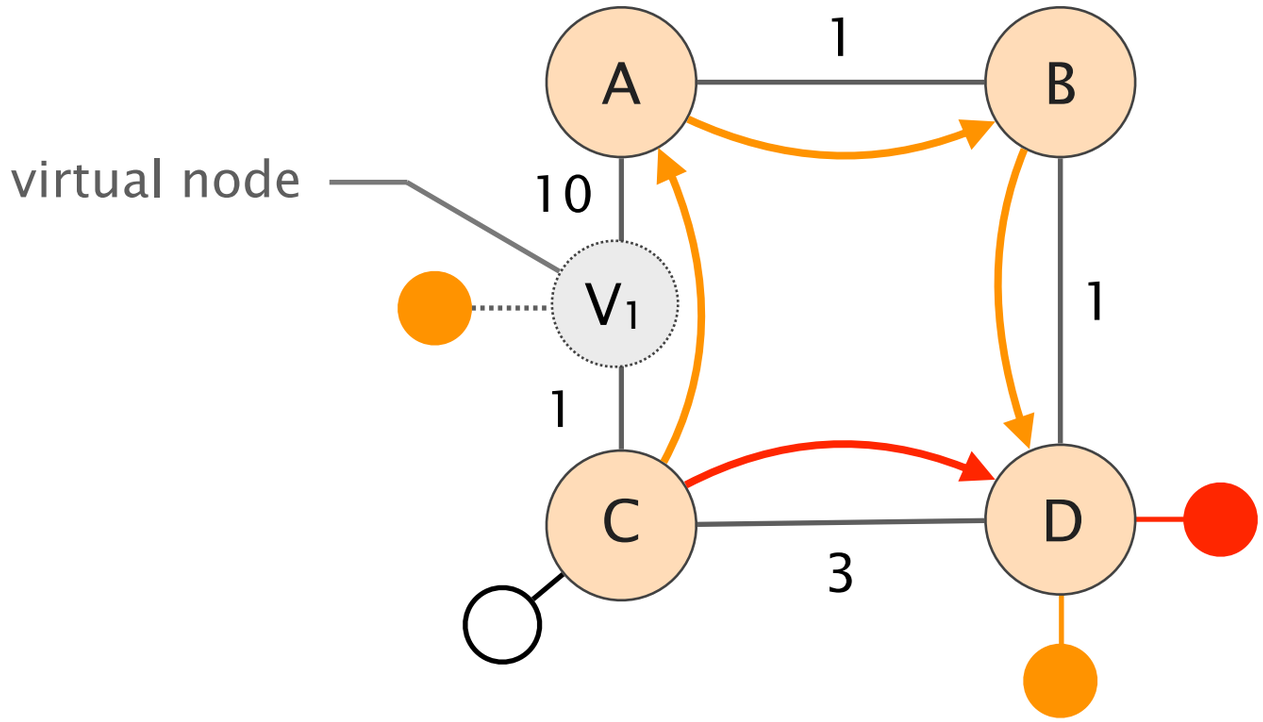


desired



impossible to achieve by
reweighing the IGP links

SDN-controlled IGP can move the orange flow by adding a virtual node announcing the orange destination



Traffic sent to V_1 by C is physically sent to A

A SDN-enabled IGP is powerful

Theorem

A SDN-enabled IGP can make the routers use any set of non-contradictory paths

A SDN-enabled IGP is powerful

Theorem

A SDN-enabled IGP can make the routers use any set of non-contradictory paths

A SDN-enabled IGP is powerful

Theorem

A SDN-enabled IGP can make the routers use any set of non-contradictory paths

- any path is loop-free

  (*e.g.,* [s1, a, b, a, d] is not possible)

- paths are consistent

  (*e.g.* [s1, a, b, d] and [s2, b, a, d] are inconsistent)

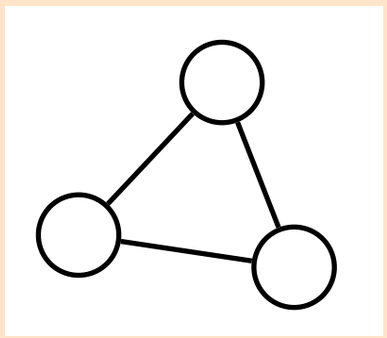
Given a physical topology and a set of path requirements,
a linear program computes an optimized virtual topology

physical topology

paths requirements

Integer Linear Program

virtual topology



+

[RA, RB, RC, RD]

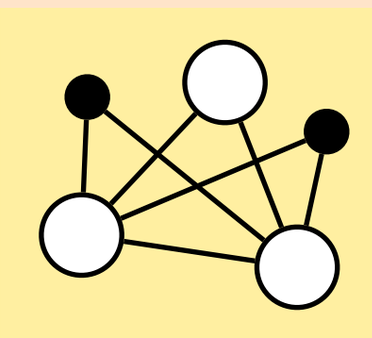
ECMP(
  [RX, RY, RZ],
  [RX, RW, RZ])

[RI, RJ, RK], backup
[RI, RL, RK]

⇒

Optimizer

⇒



minimize topology size

SDN-controlled routing enables to realize parts of the SDN promises today, on an existing network

Facilitate SDN deployment

SDN controller can program routers *and* SDN switches

Simplify controller implementation

most of the heavy work is still done by the routers

Maintain operators' mental model

good old protocols running, easier troubleshooting

Enabling SDN in old school networks with Software-Controlled Routing Protocols

Laurent Vanbever

www.vanbever.eu

ONS research track

March, 4 2014