



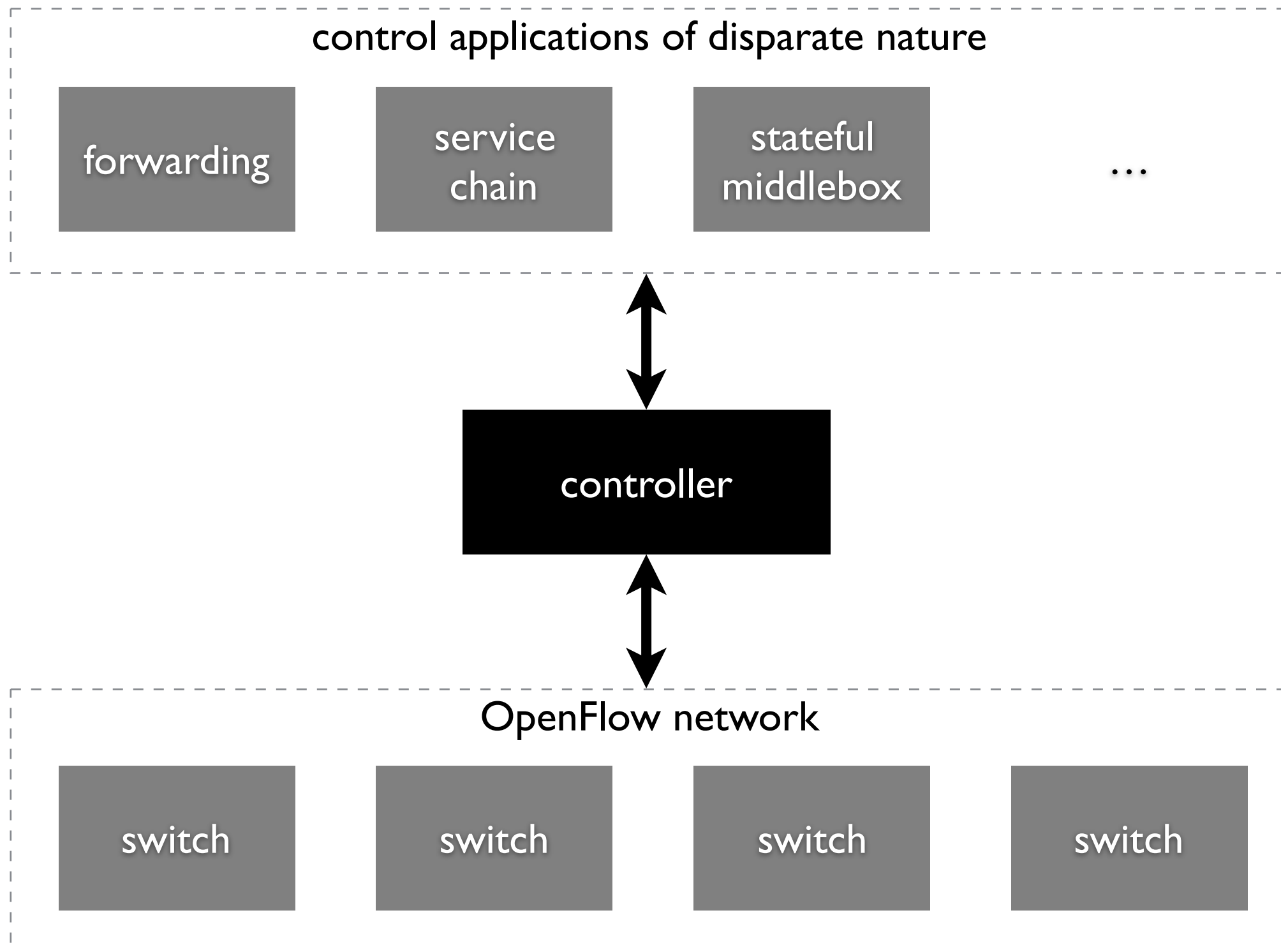
Ravel: a database-defined network

Anduo Wang* Xueyuan Mei† Jason Croft†
Matthew Caesar† Brighten Godfrey†

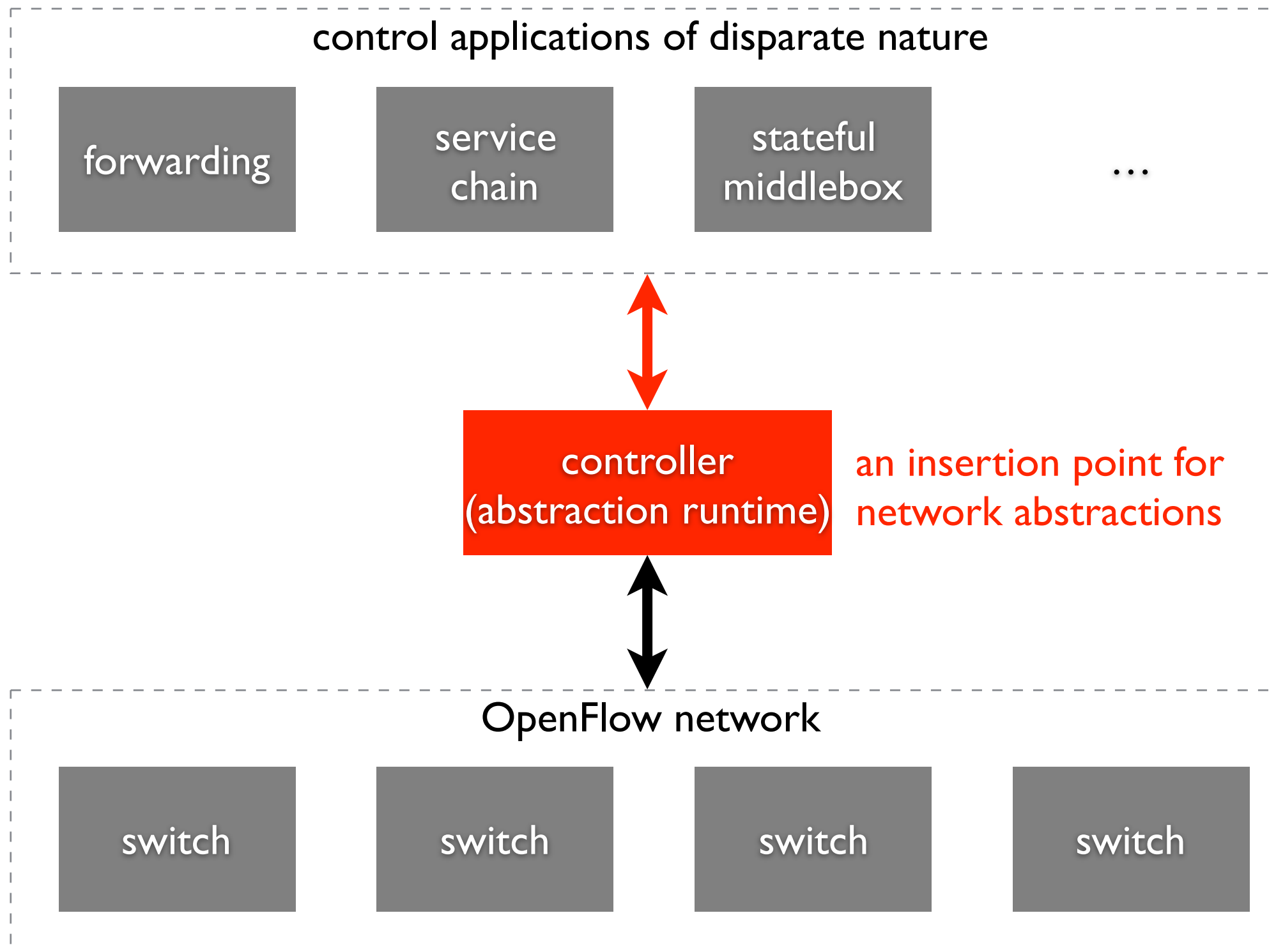
**Temple University*

†University of Illinois Urbana-Champaign

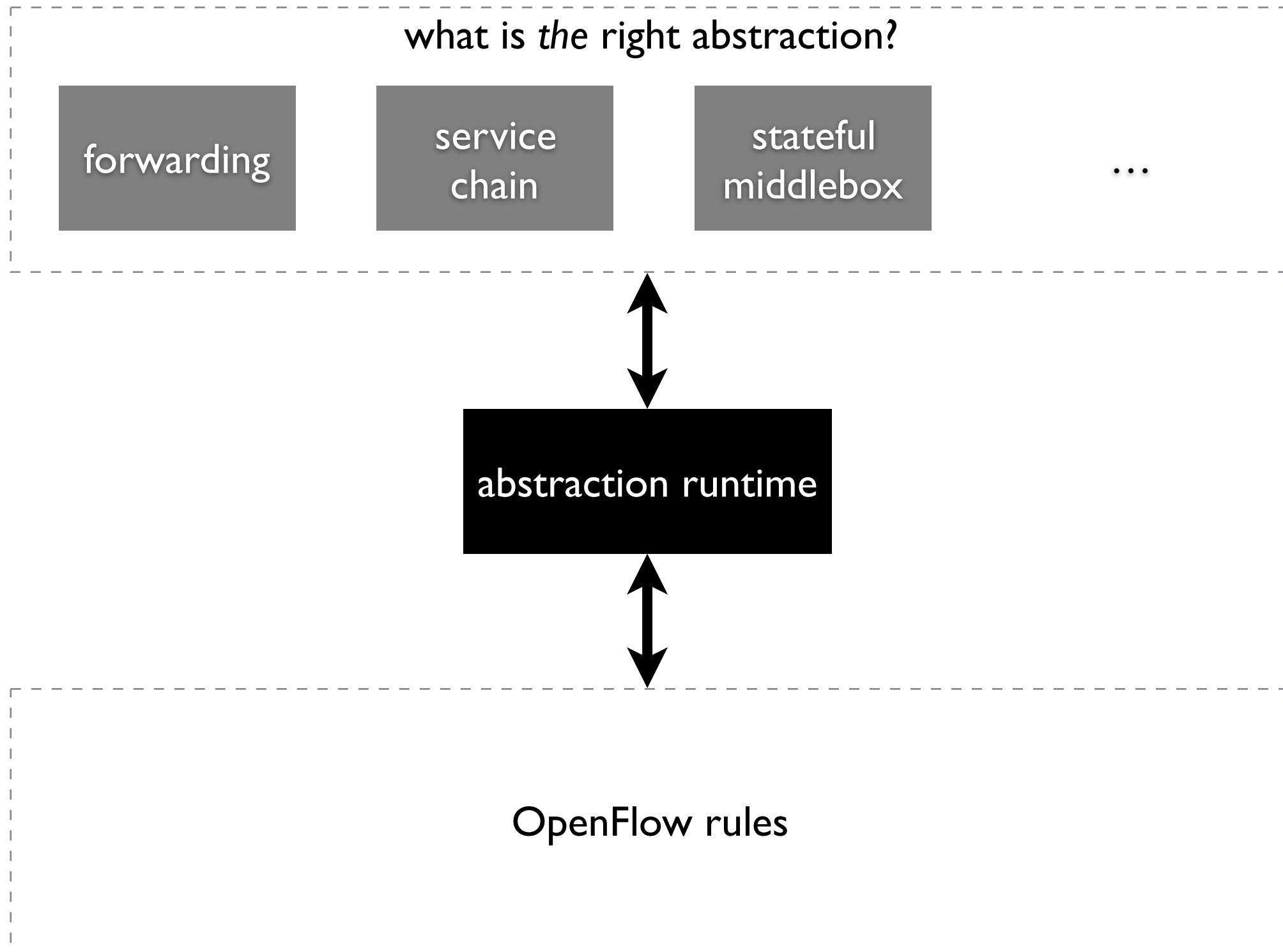
software-defined network



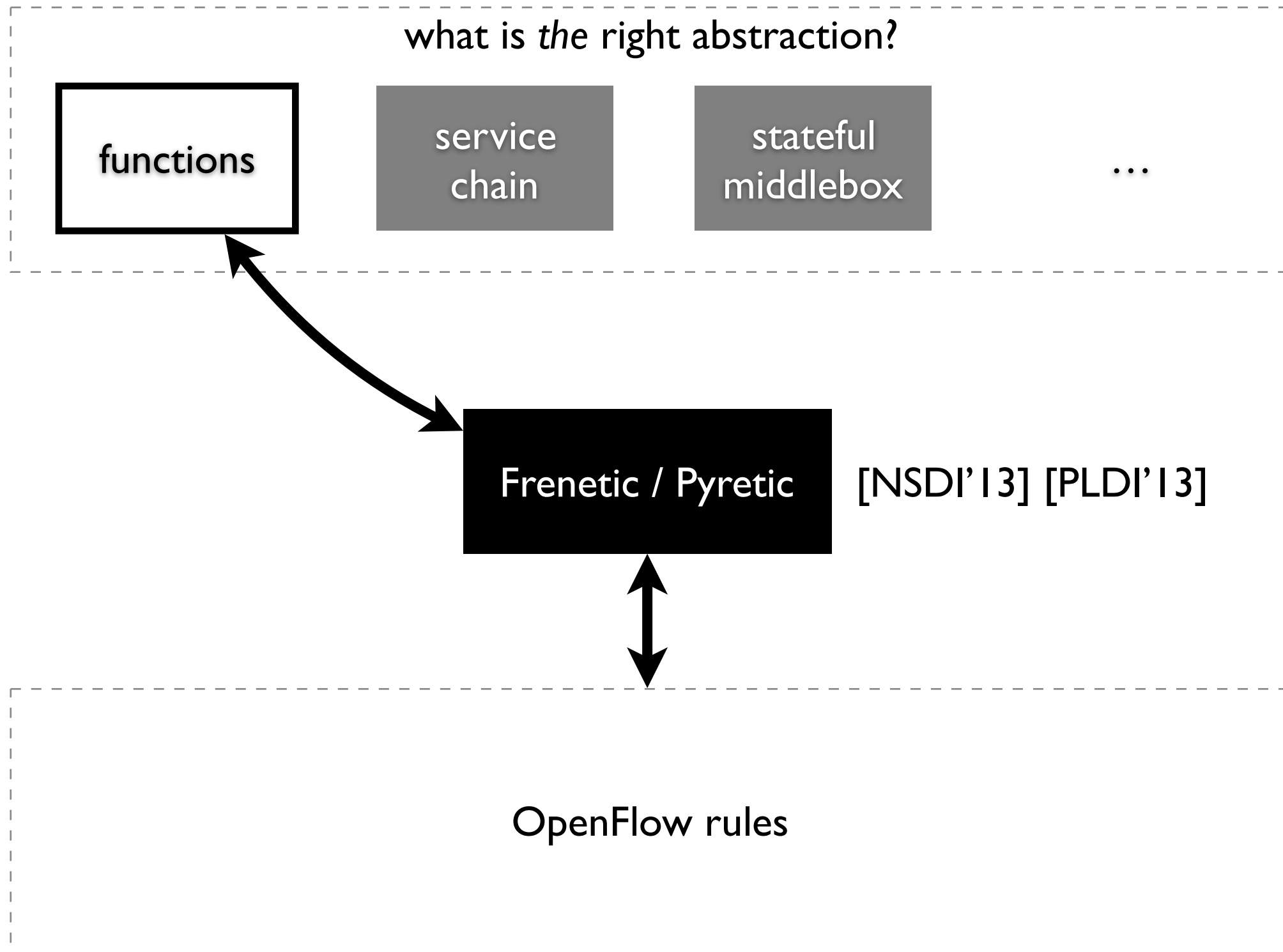
software-defined network



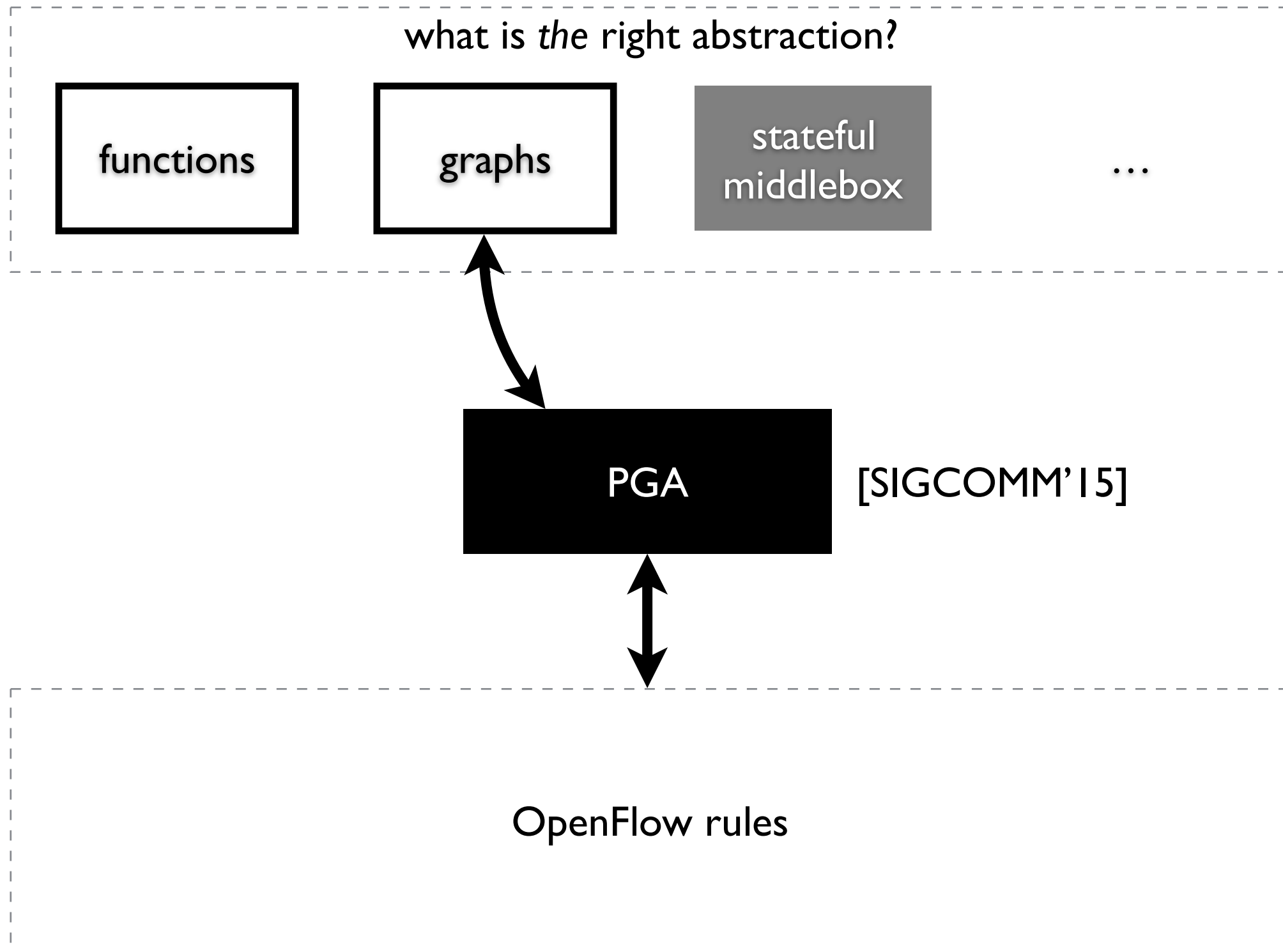
abstractions



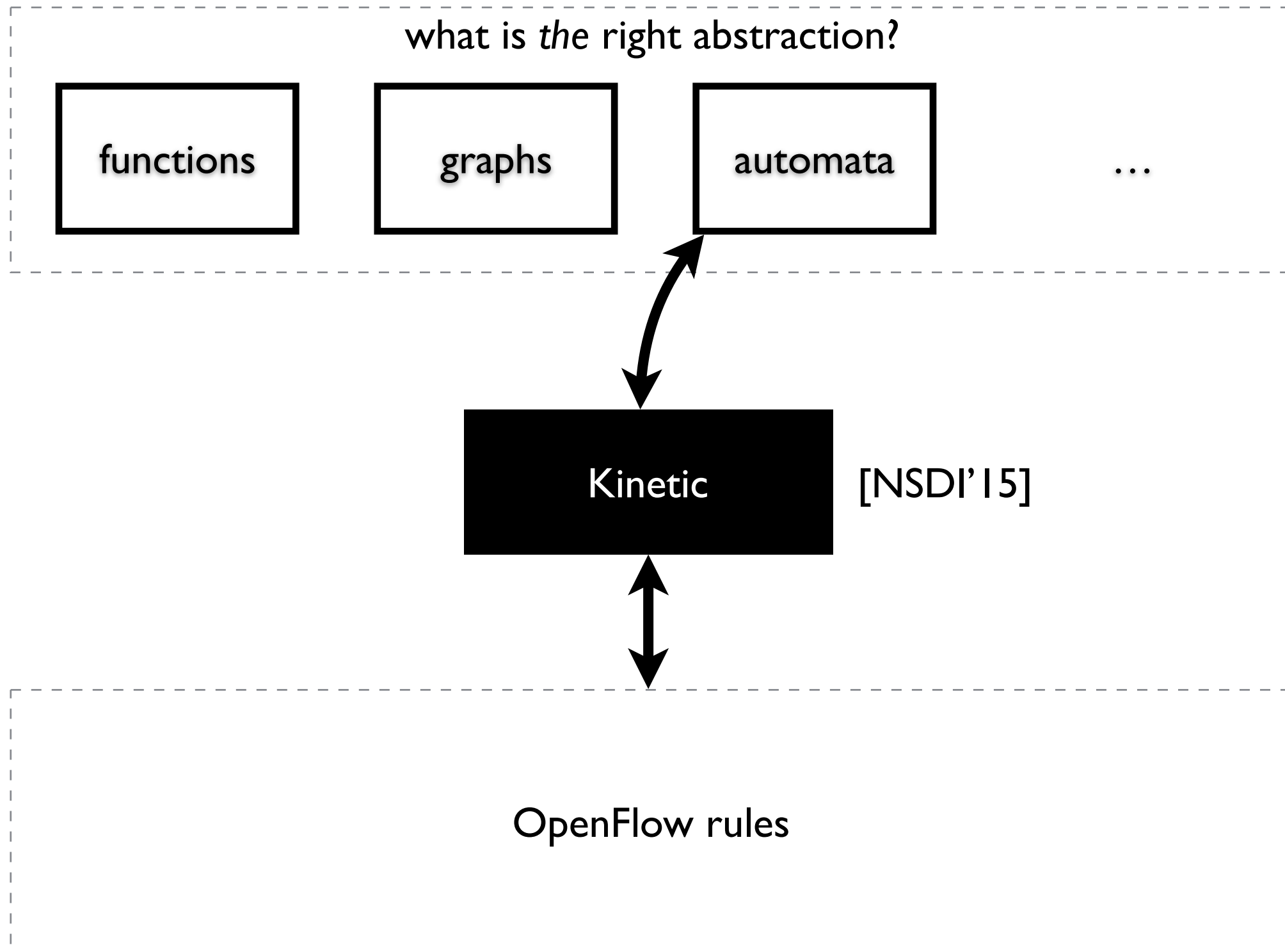
abstractions



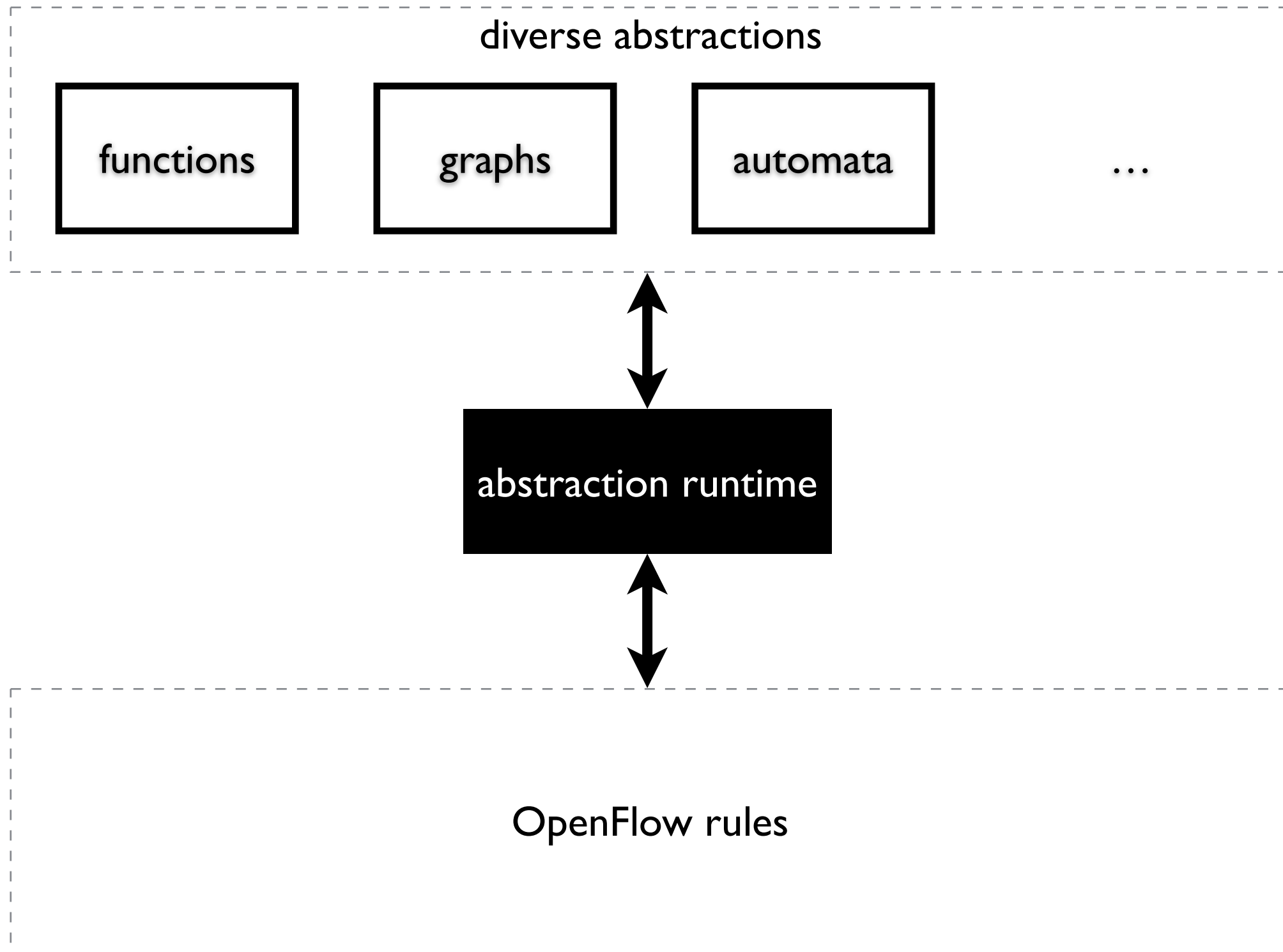
abstractions



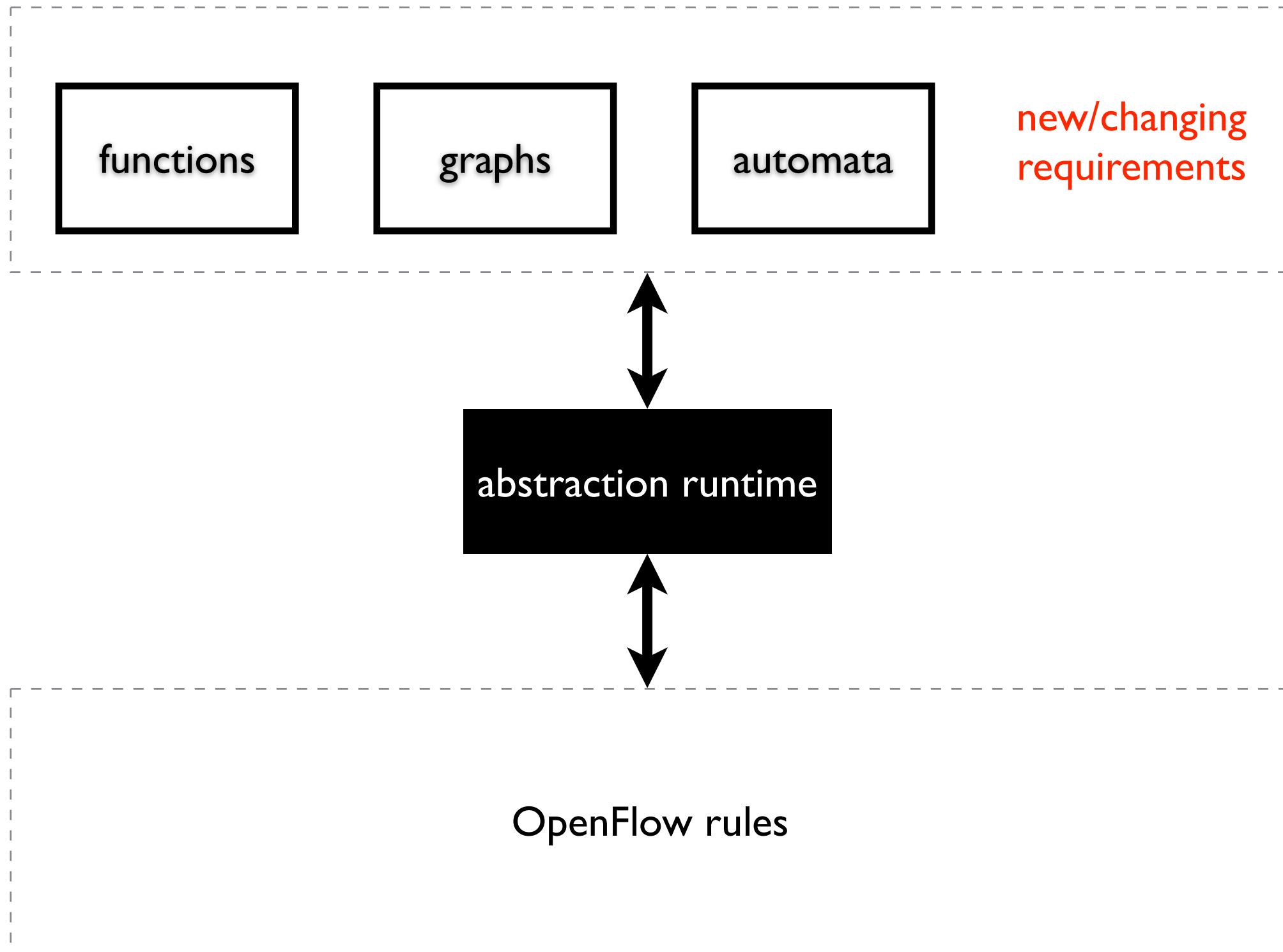
abstractions



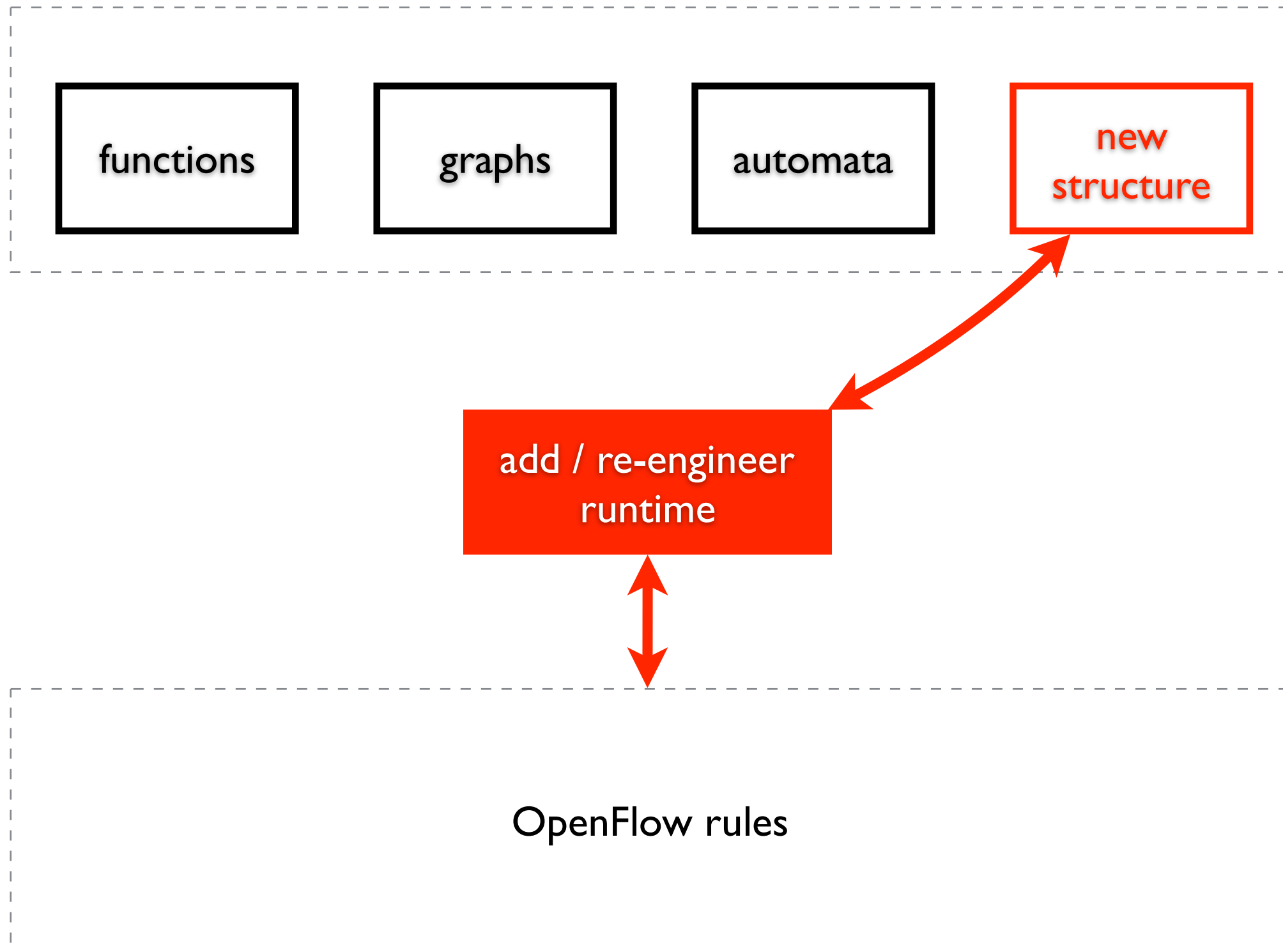
abstractions



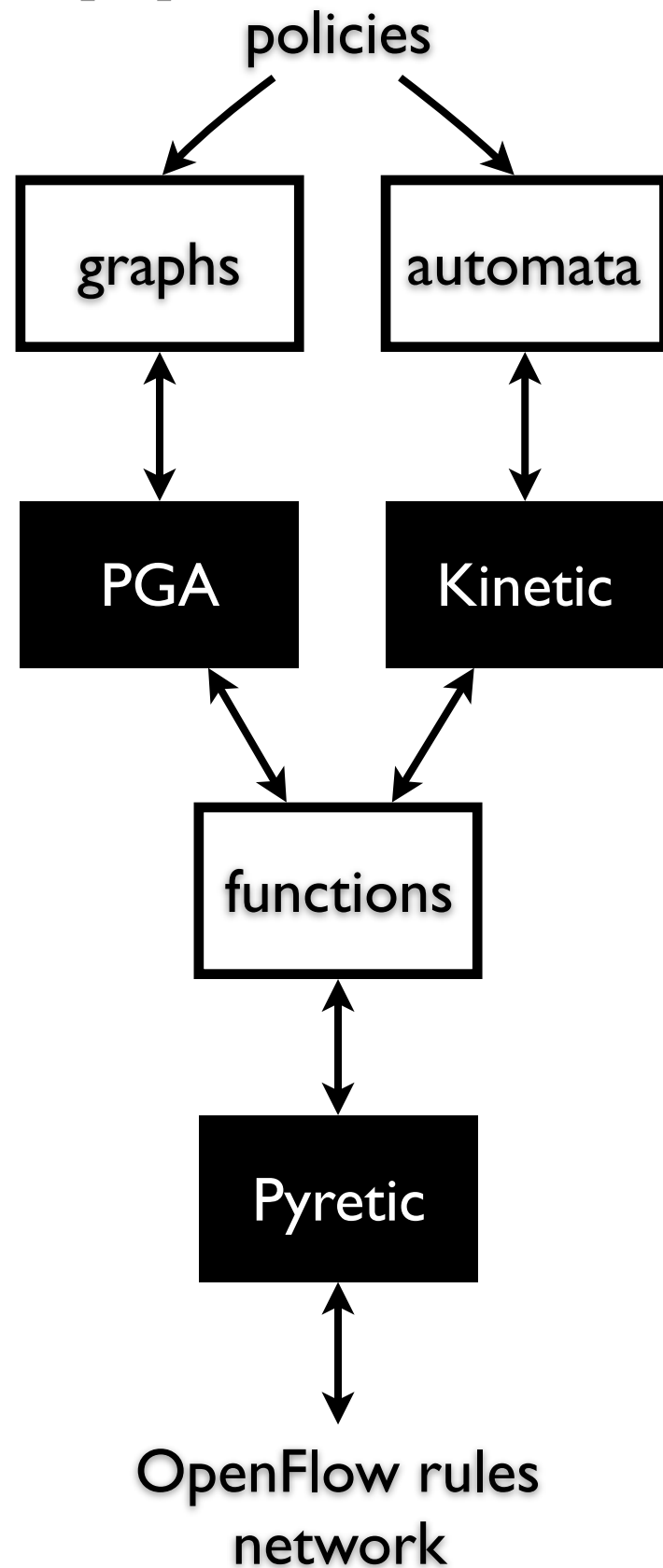
but network keeps evolving



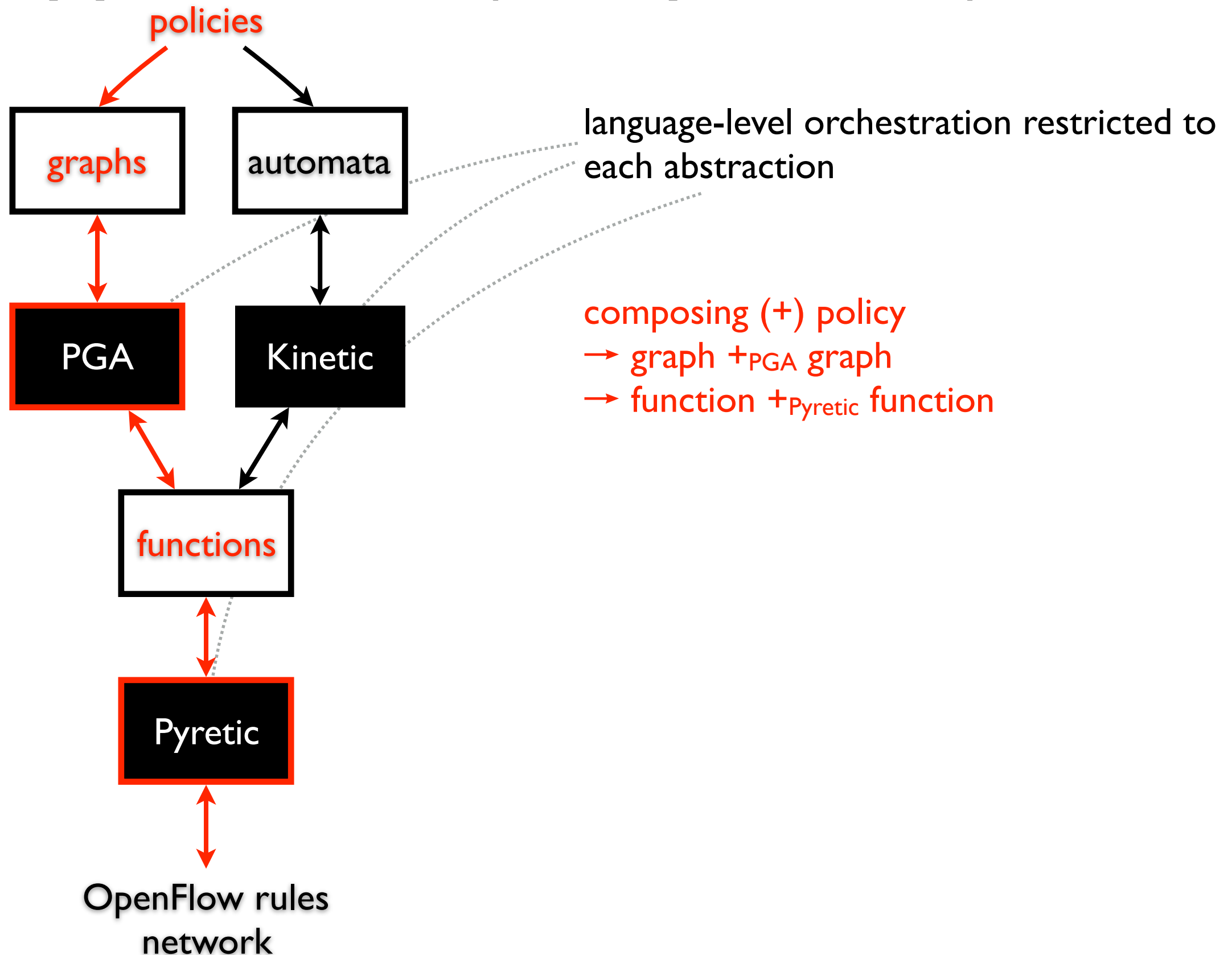
but network keeps evolving



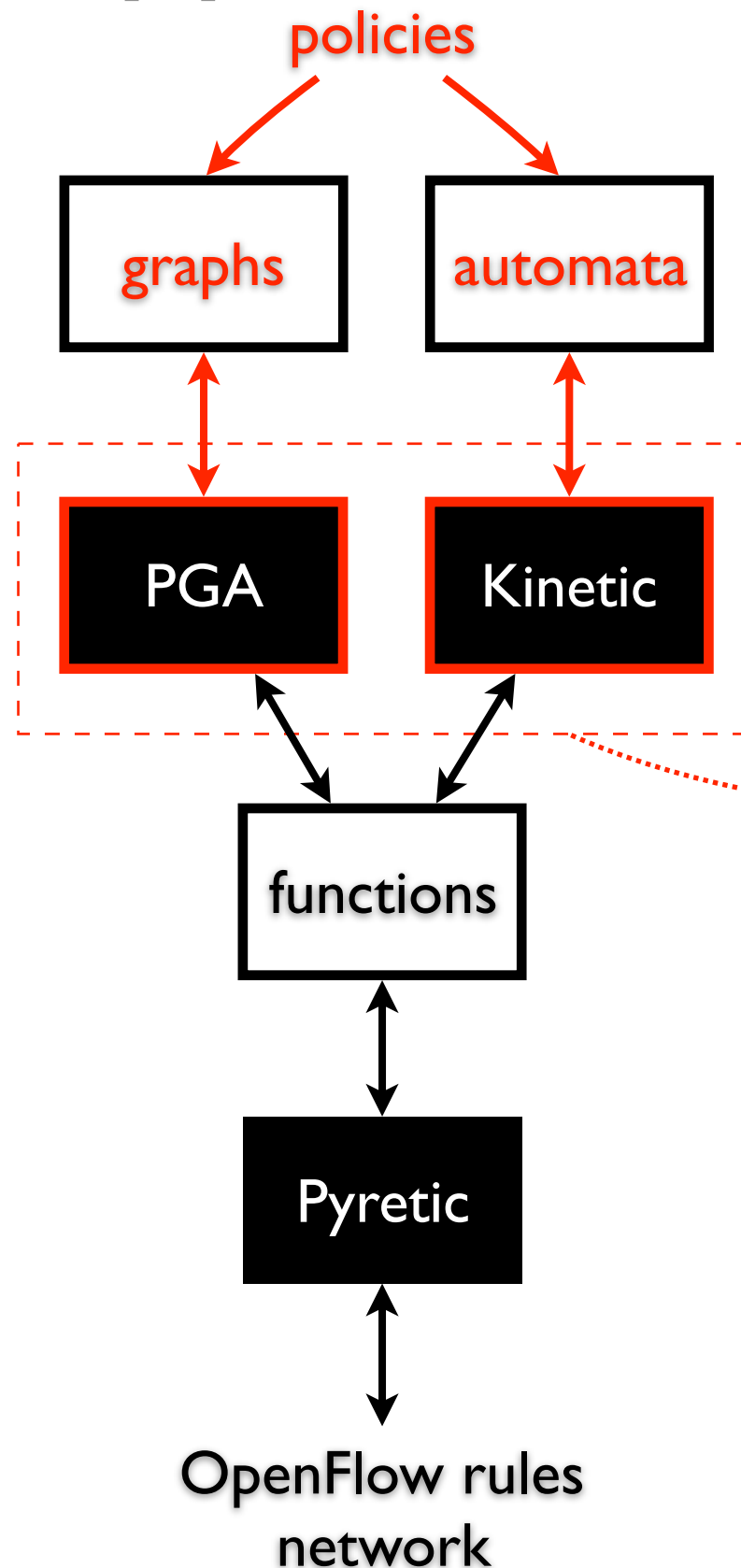
and applications (components) interact



and applications (components) interact



and applications (components) interact

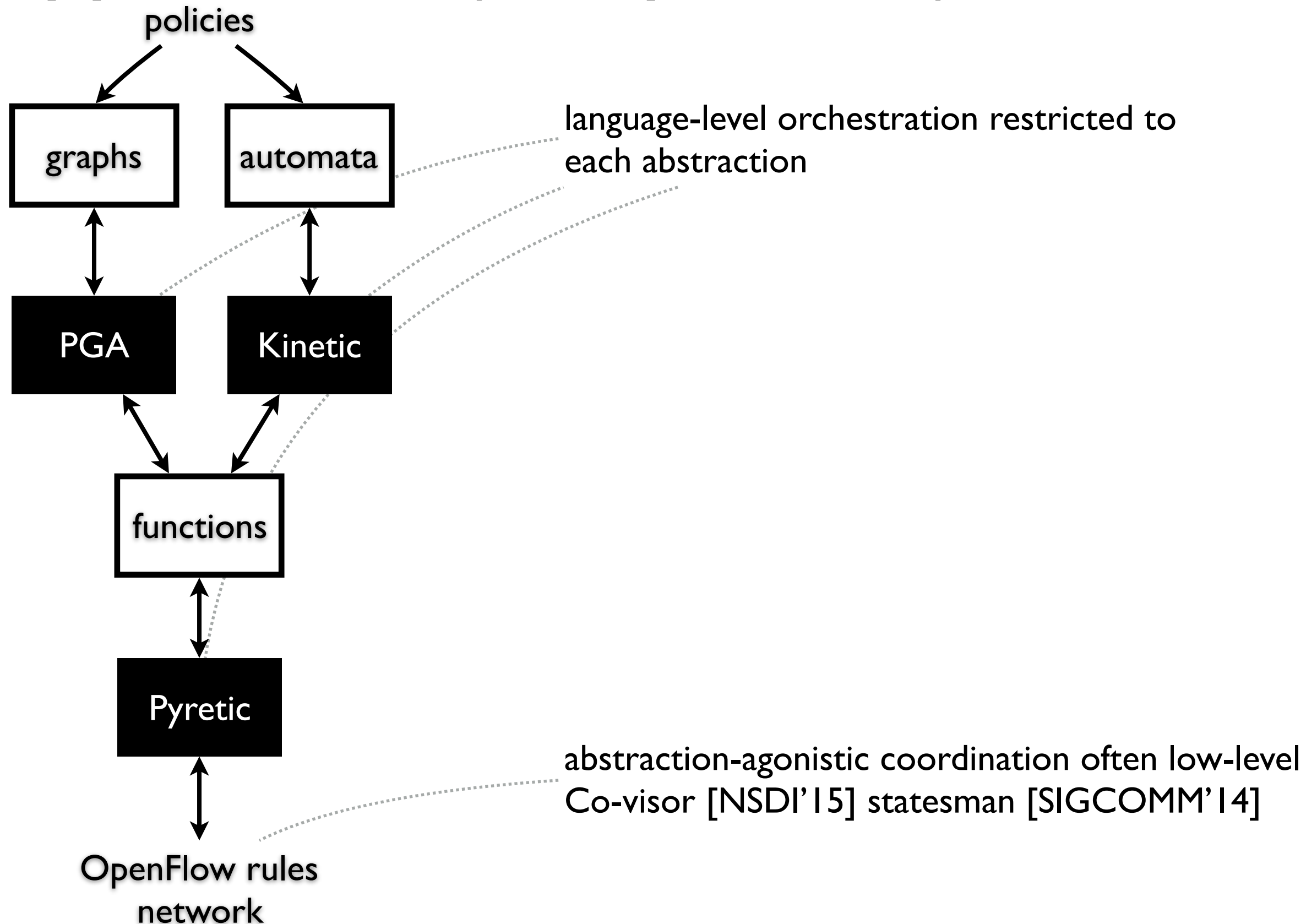


language-level orchestration restricted to each abstraction

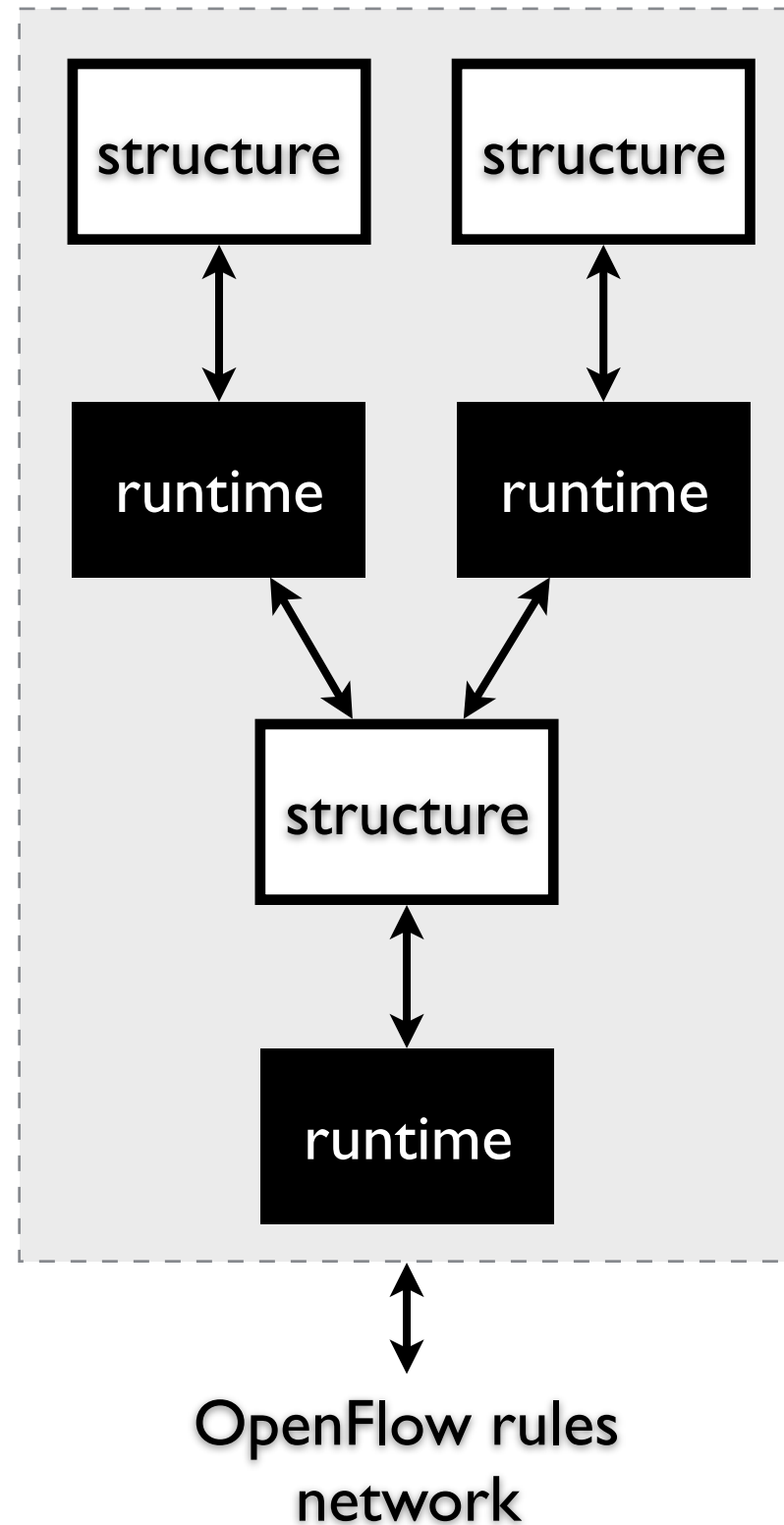
composing (+) policy
→ graph +? automata

how to integrate the runtime?
hard-wiring internals?

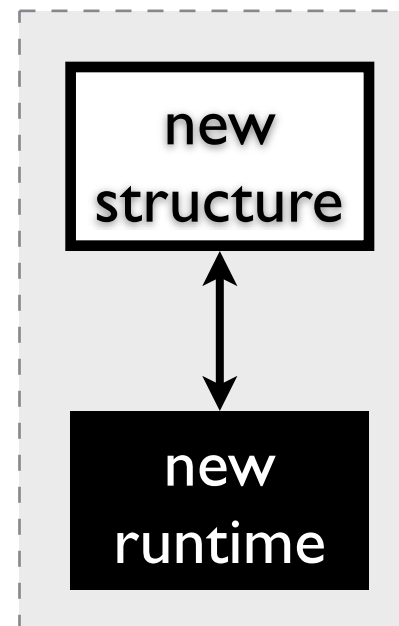
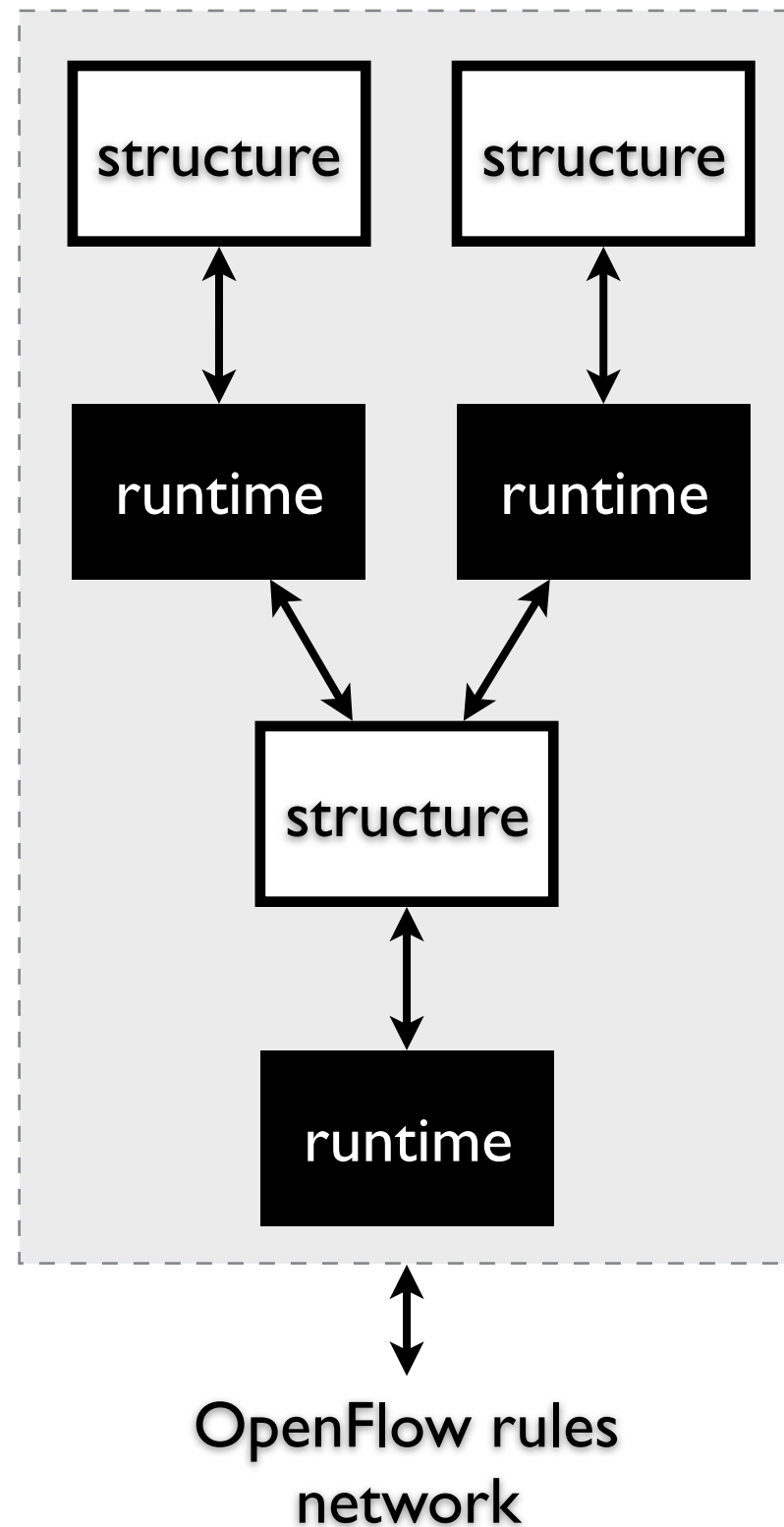
and applications (components) interact



current state of abstraction research

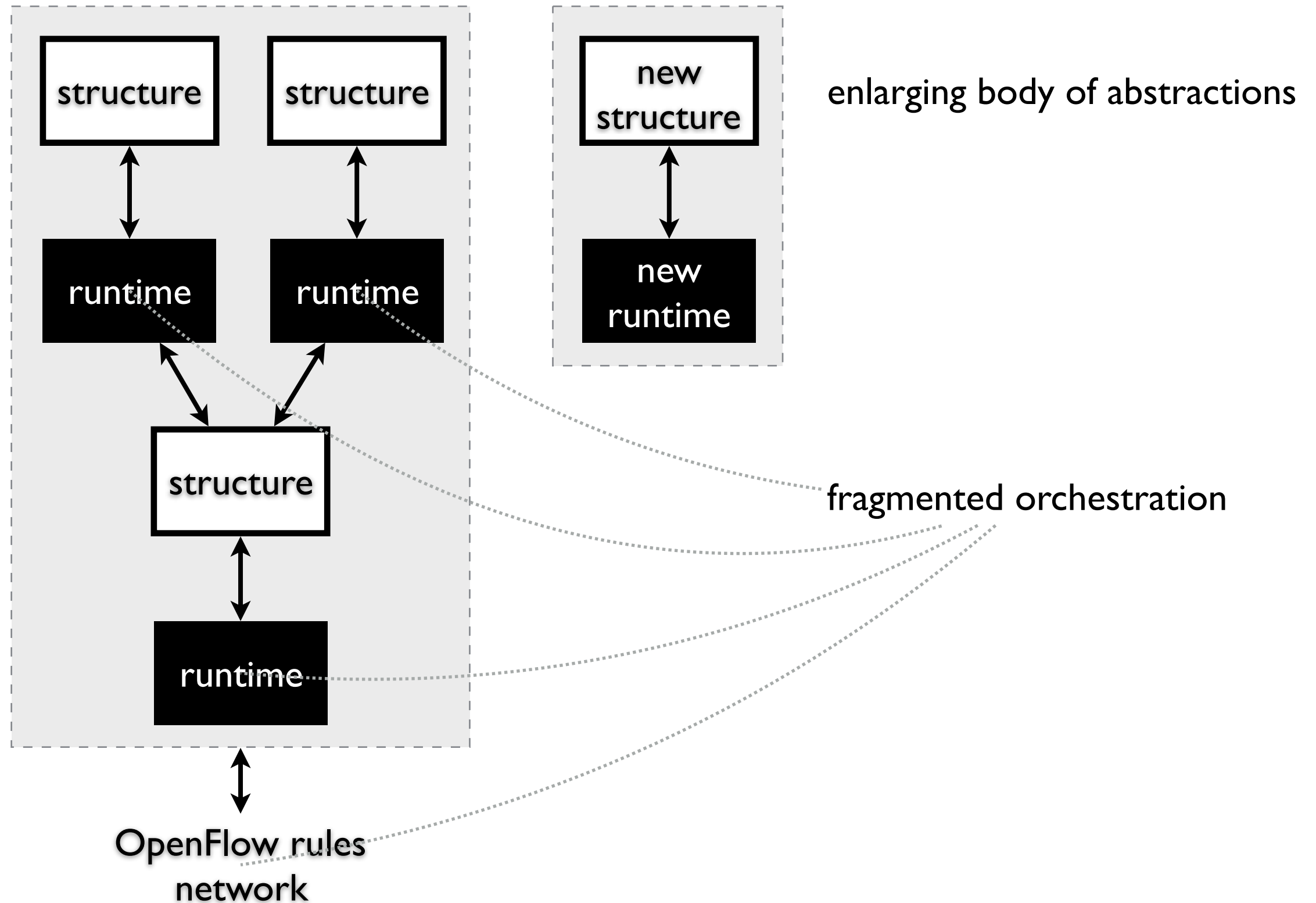


current state of abstraction research



enlarging body of abstractions

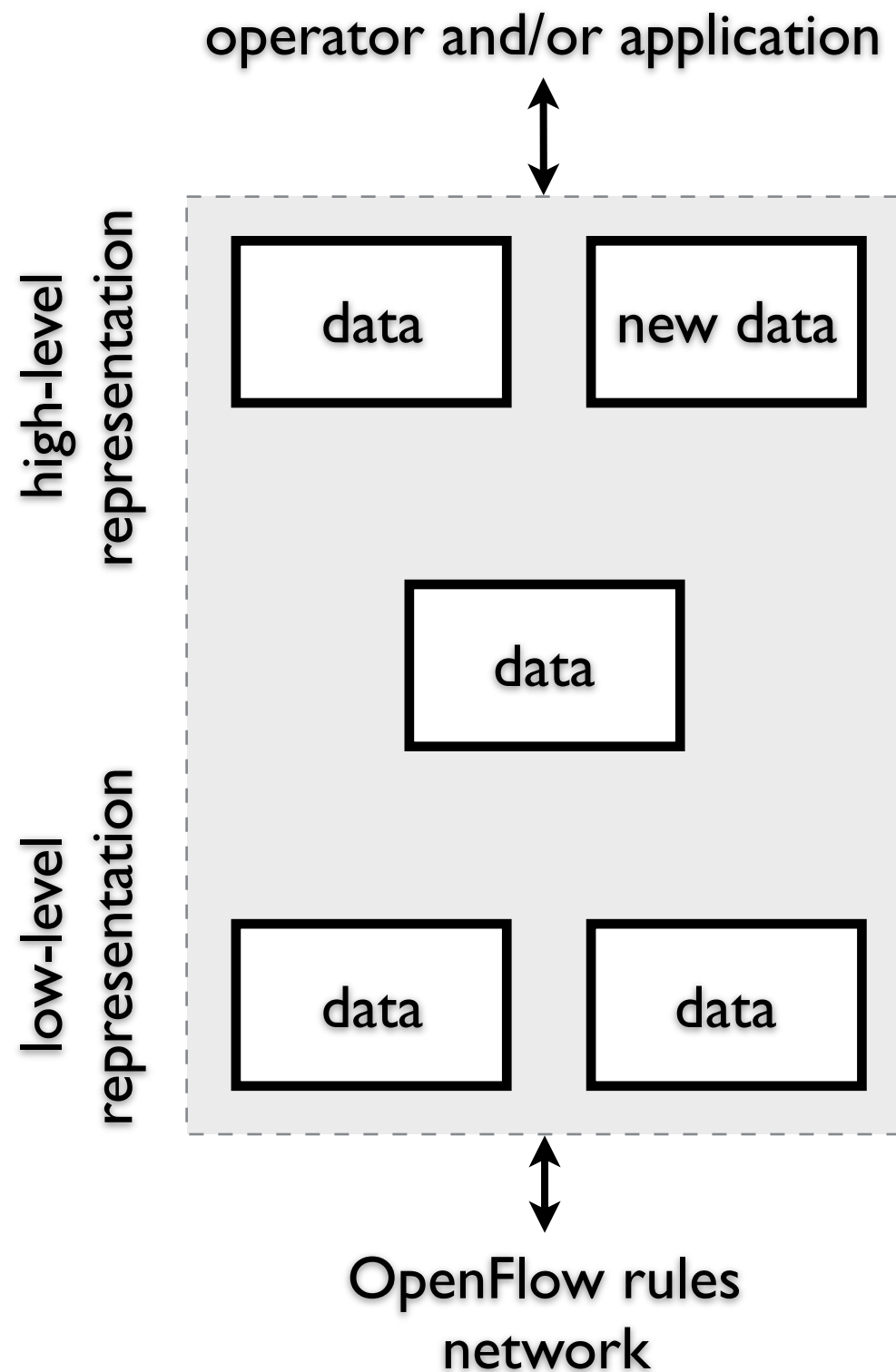
current state of abstraction research



our perspective

SDN control revolves around data representation

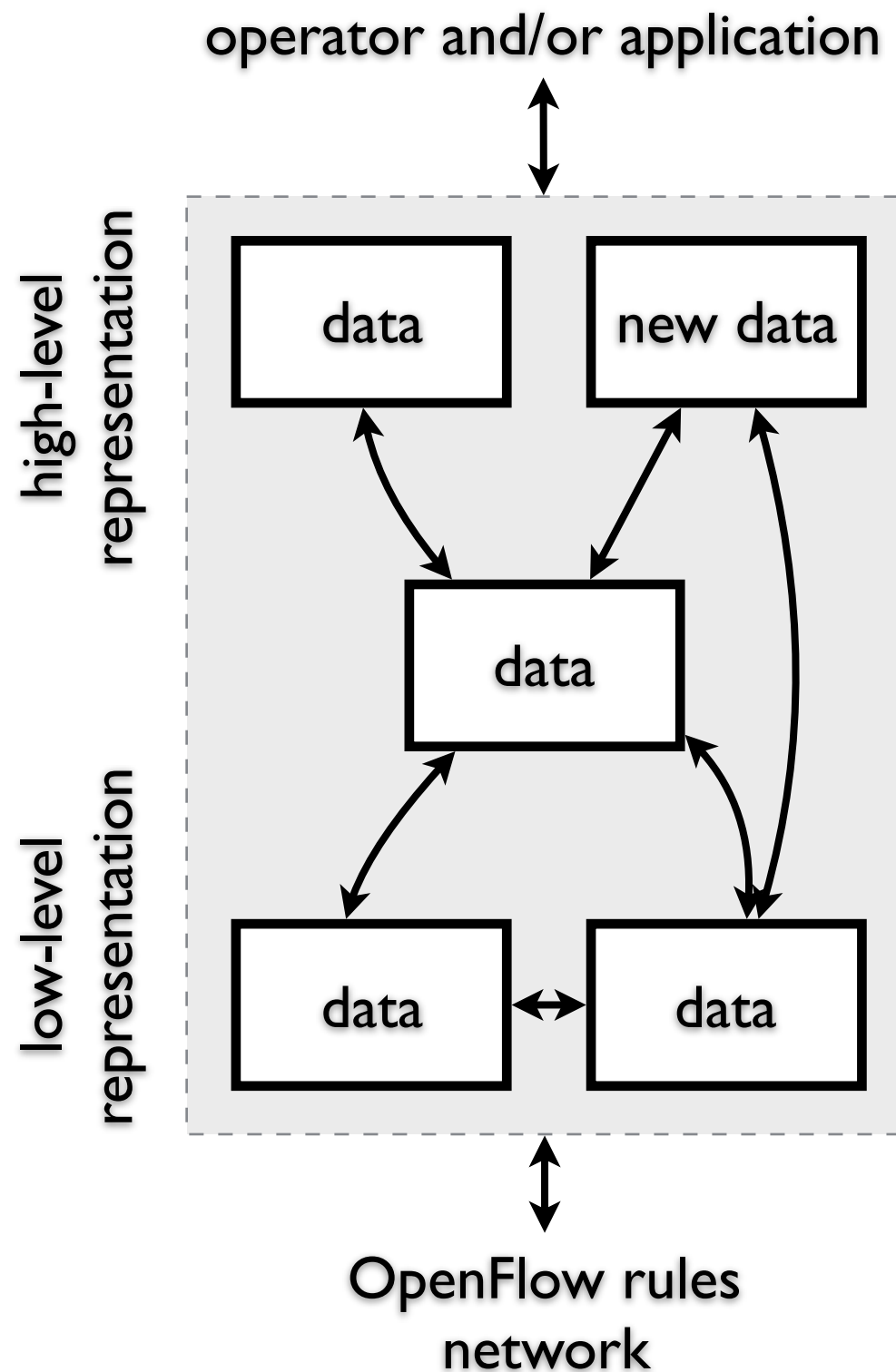
- discard specialized, pre-compiled, fixed structures
- adopt a *plain data representation*



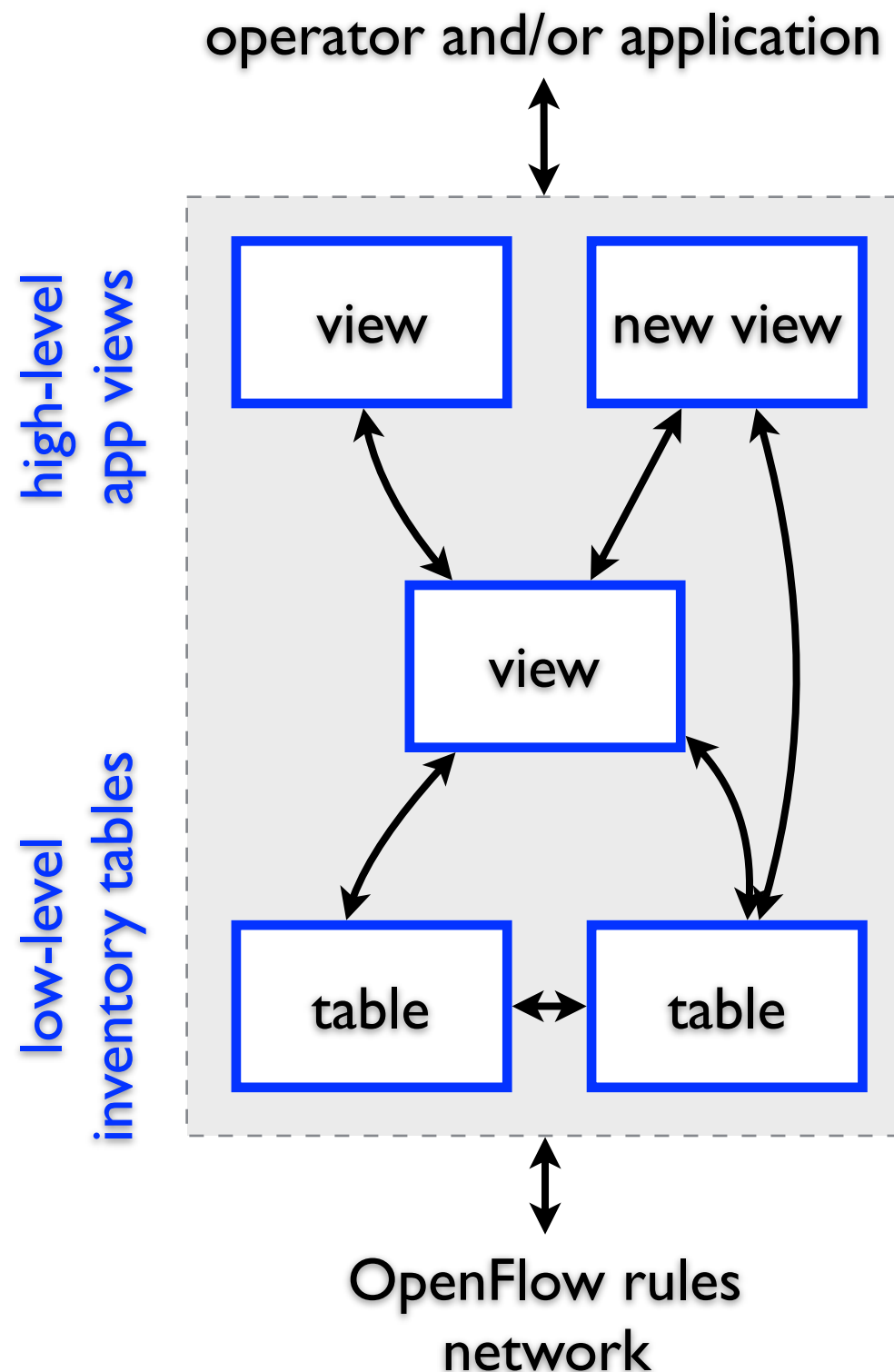
our perspective

SDN control revolves around data representation

- discard specialized, pre-compiled, fixed structures
- adopt a *plain data representation*
- use a *universal data language*

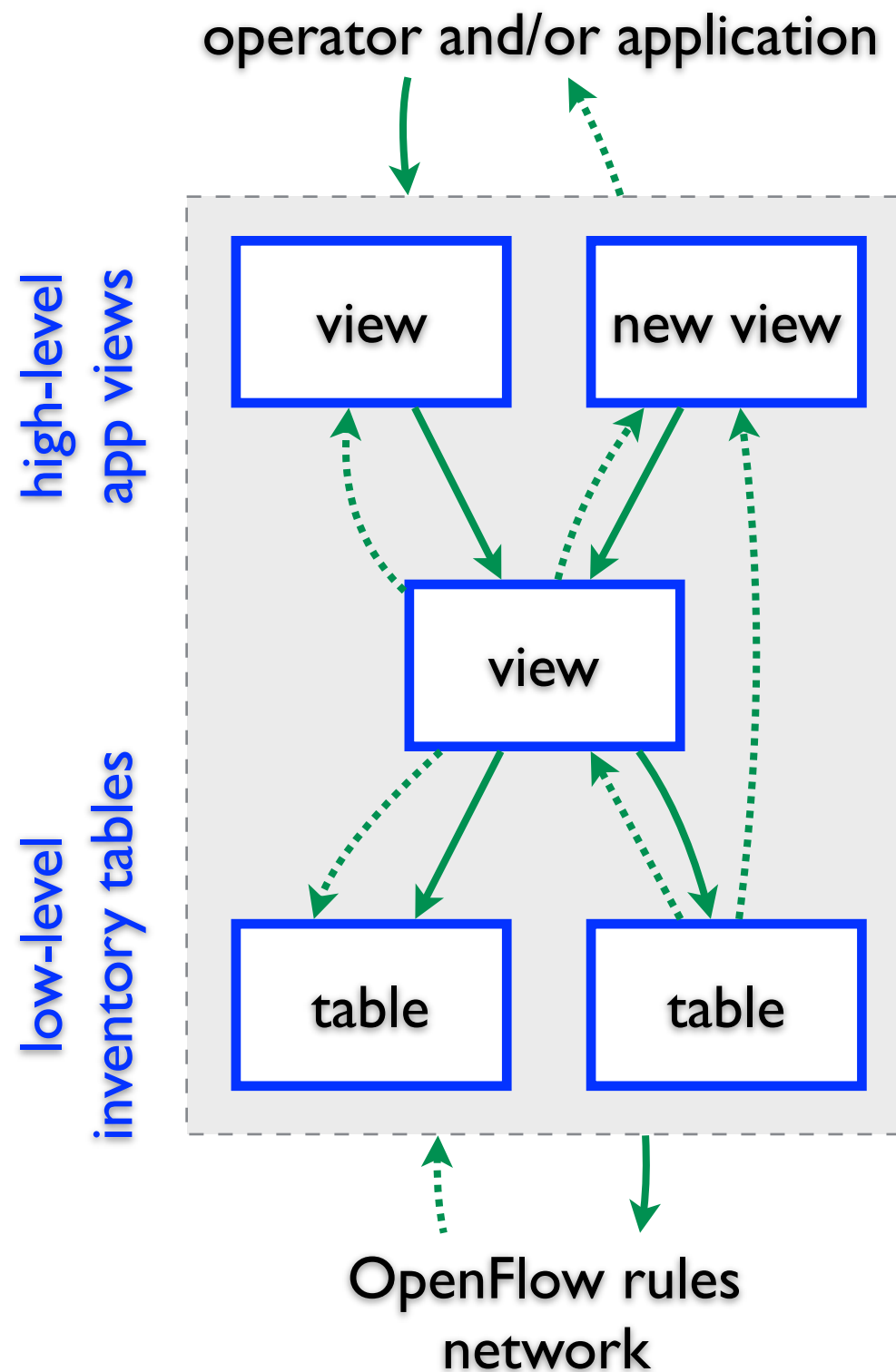


a database-defined network



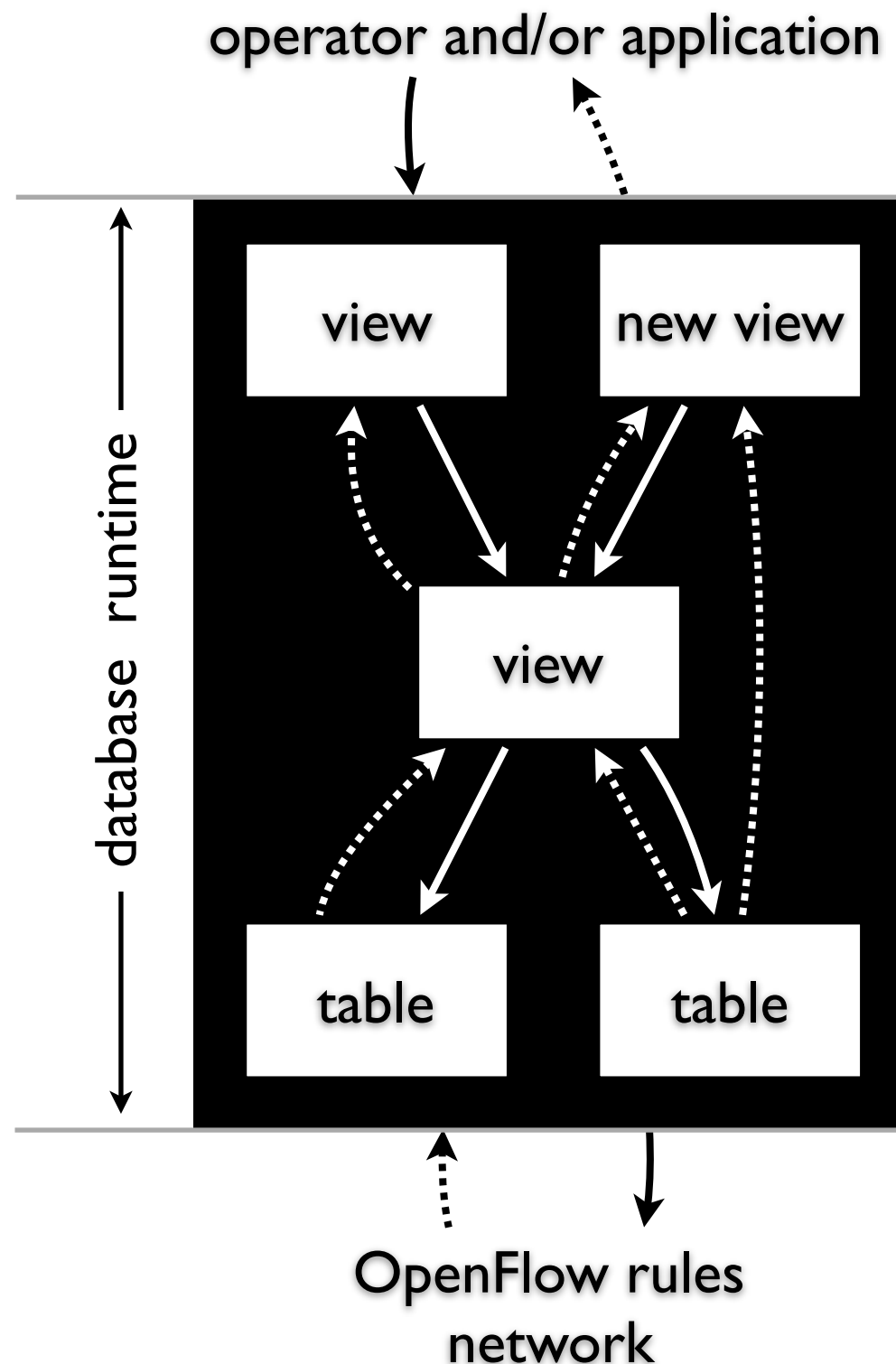
- **relation** — the plain data representation
- table — stored relation
- view — virtual relation

a database-defined network



- ─ **relation** — the plain data representation
 - ─ table — stored relation
 - ─ view — virtual relation
- ─ **SQL** — the universal data language
 - ─ query, update, trigger, rule

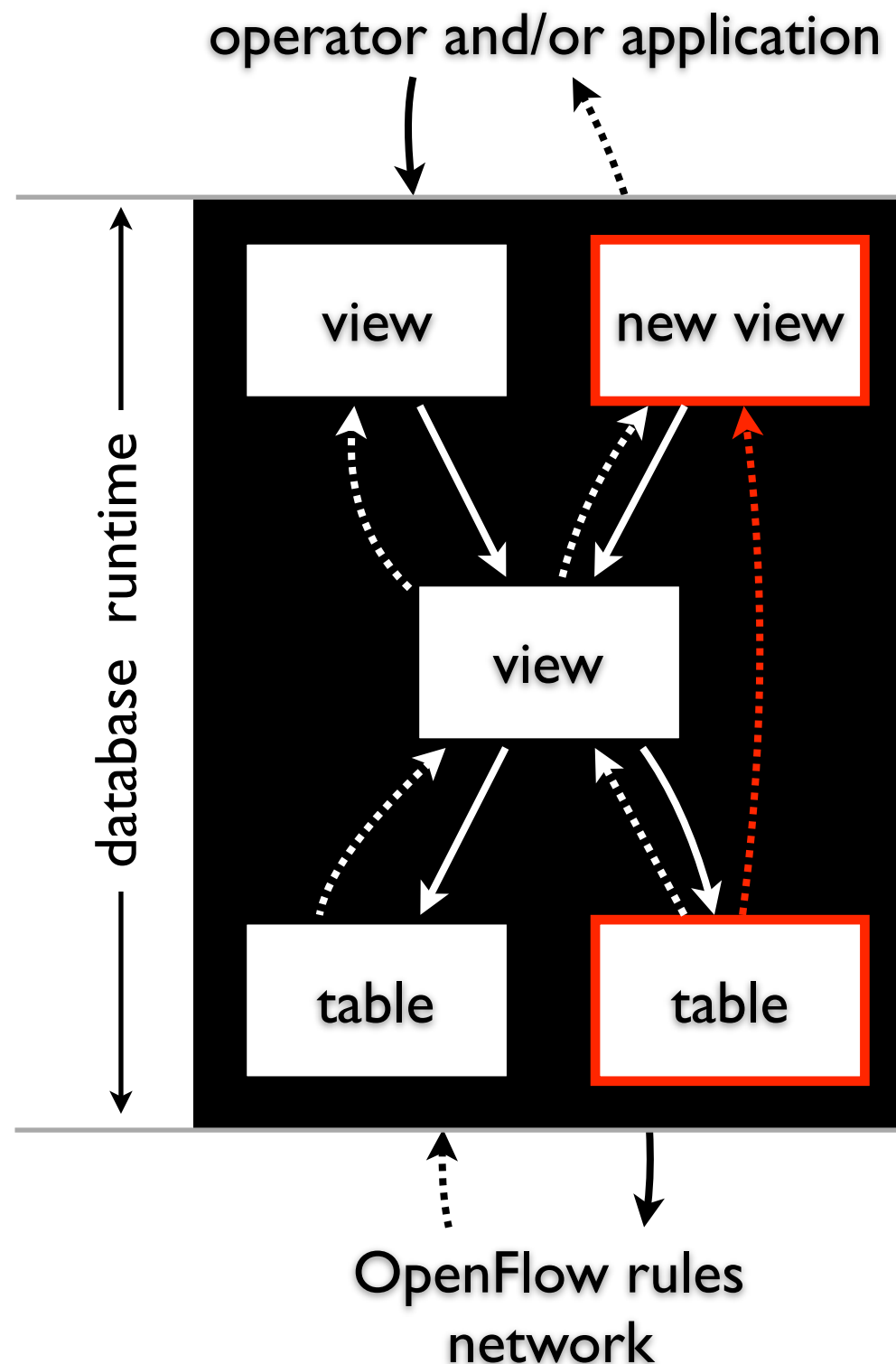
Ravel: a realization with SQL database



attractive features

- ad-hoc programmable abstraction via views
- orchestration across abstractions via view mechanism
- orchestration across applications via data mediation
- network control via SQL

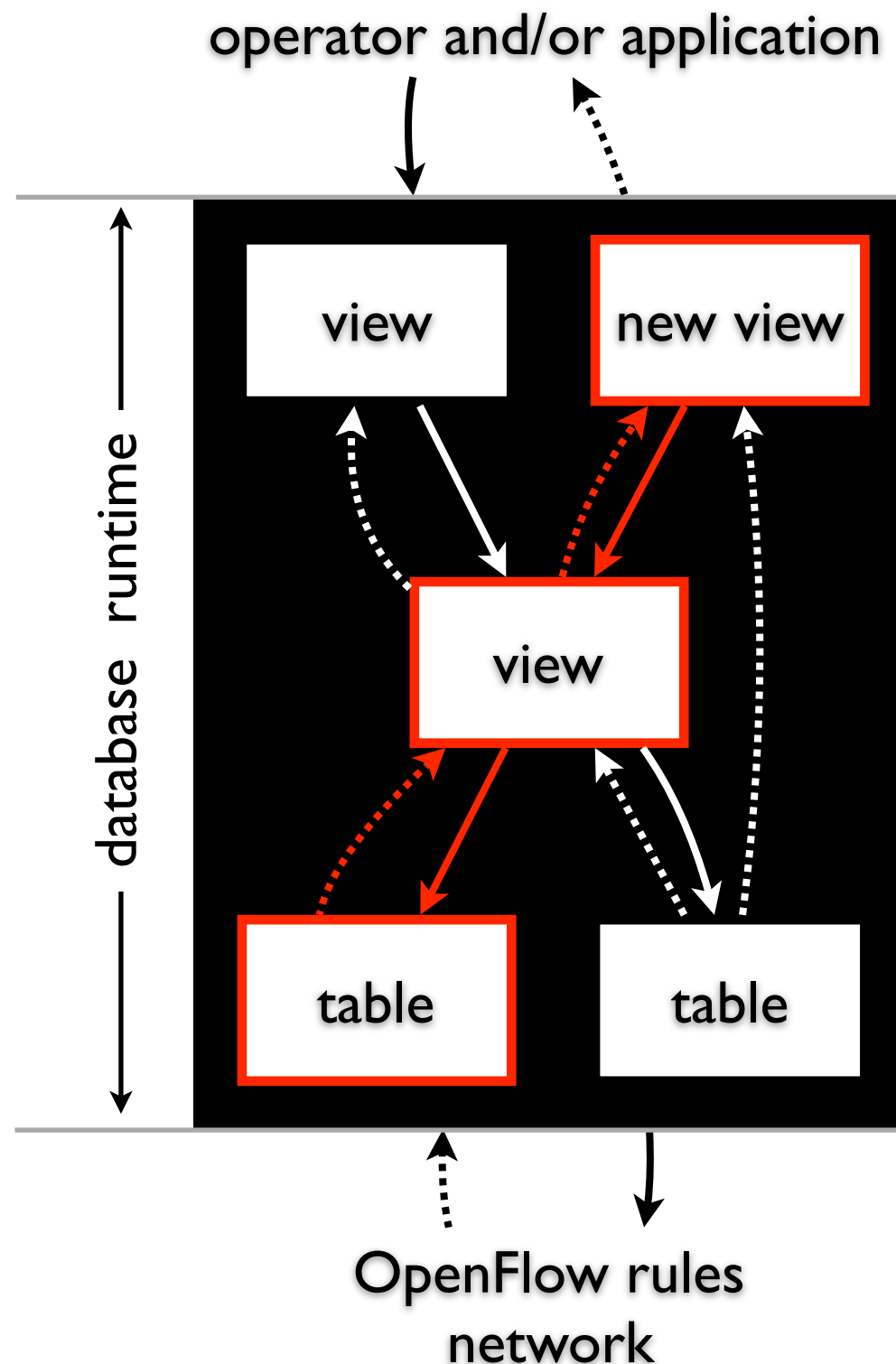
Ravel: a realization with SQL database



attractive features

- ad-hoc programmable abstraction via views
- orchestration across abstractions via view mechanism
- orchestration across applications via data mediation
- network control via SQL

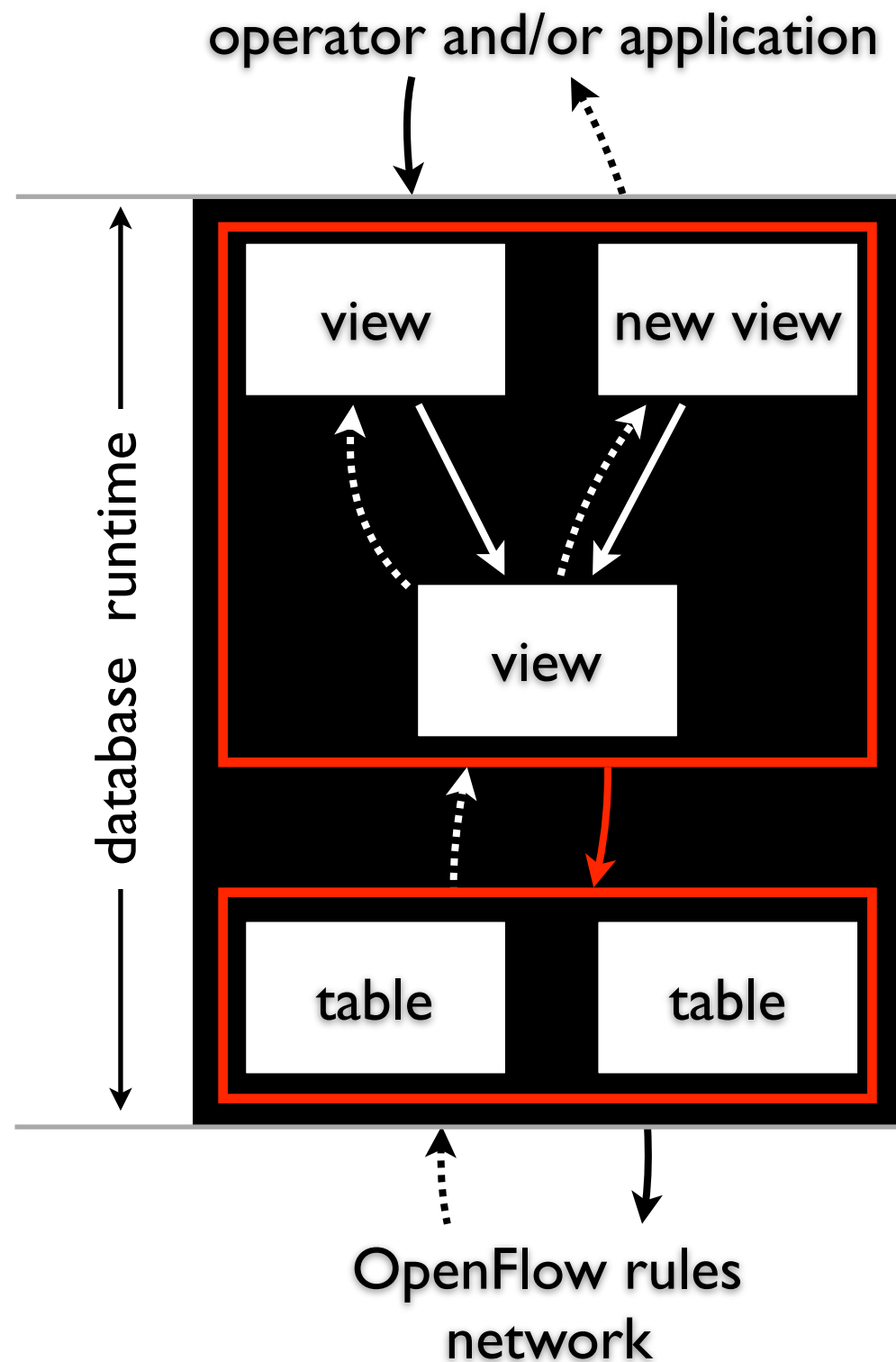
Ravel: a realization with SQL database



attractive features

- ad-hoc programmable abstraction via views
- orchestration across abstractions via view mechanism
- orchestration across applications via data mediation
- network control via SQL

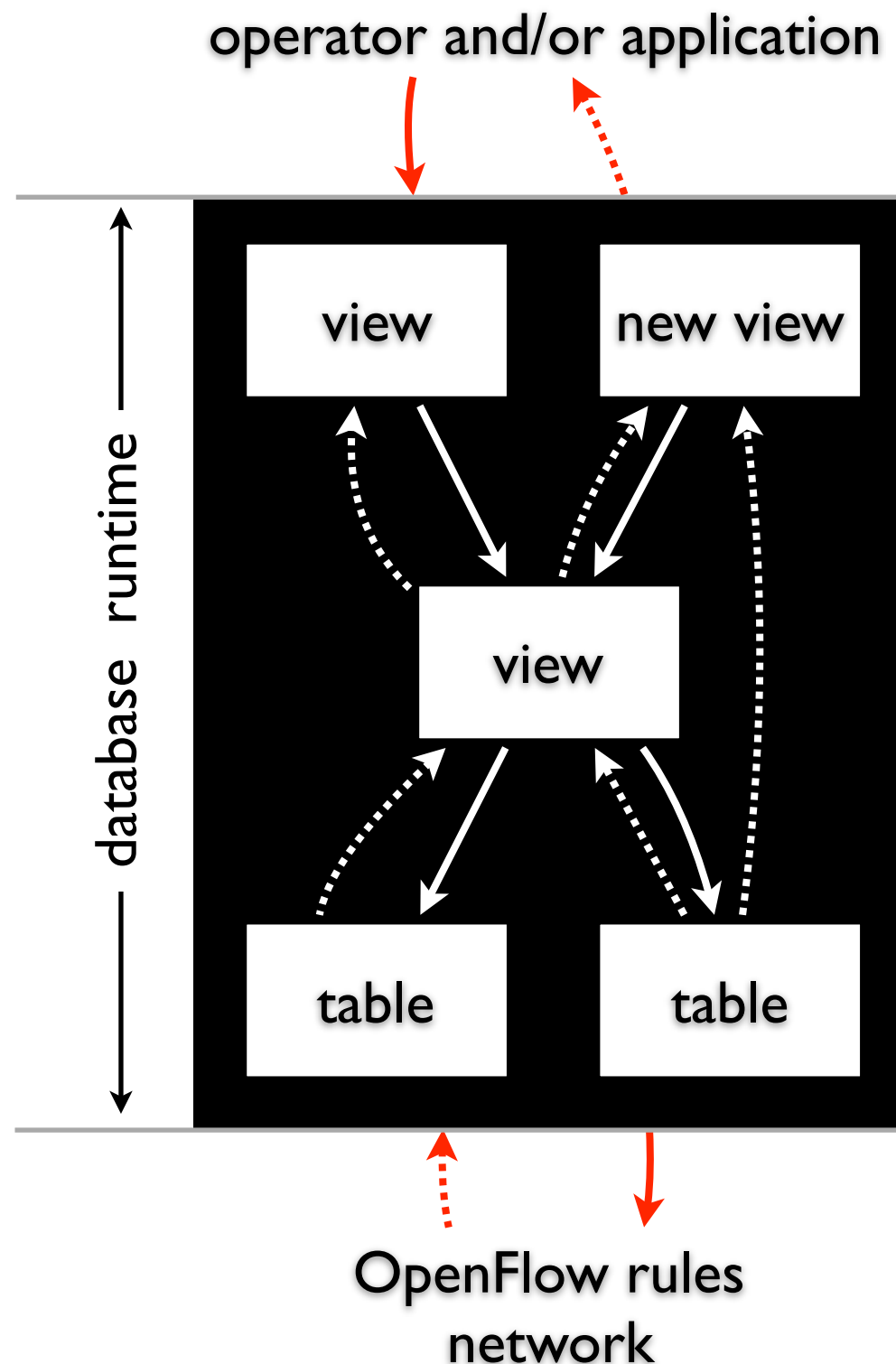
Ravel: a realization with SQL database



attractive features

- ad-hoc programmable abstraction via views
- orchestration across abstractions via view mechanism
- orchestration across applications via data mediation
- network control via SQL

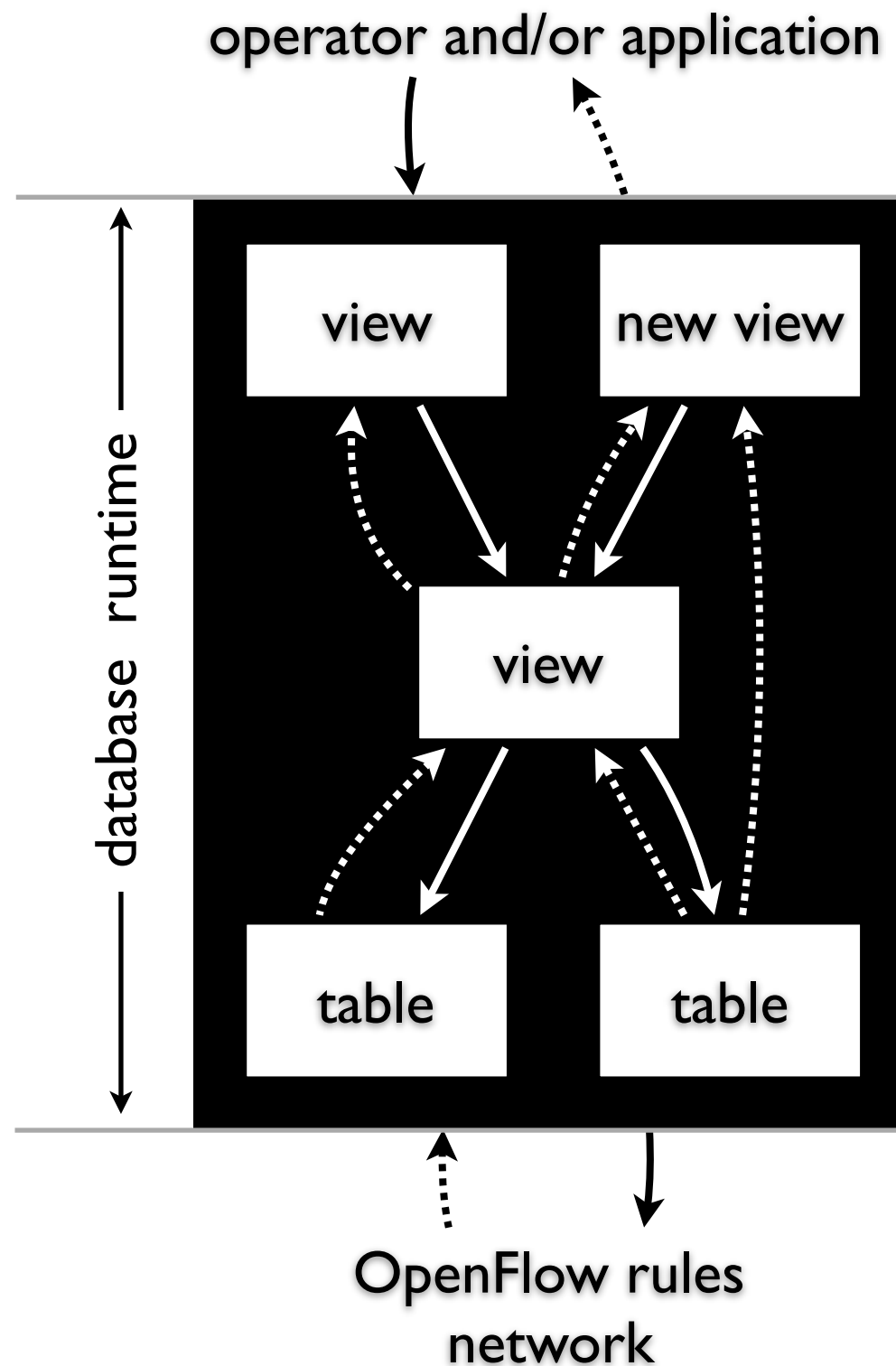
Ravel: a realization with SQL database



attractive features

- ad-hoc programmable abstraction via views
- orchestration across abstractions via view mechanism
- orchestration across applications via data mediation
- network control via SQL

Ravel: a realization with SQL database



attractive features

- abstraction
- orchestration
- SQL

abstraction: network tables

reachability matrix

fid	src	dst	vol	...
1	h_1	h_4	5	
2	h_2	h_3	9	

...

topology

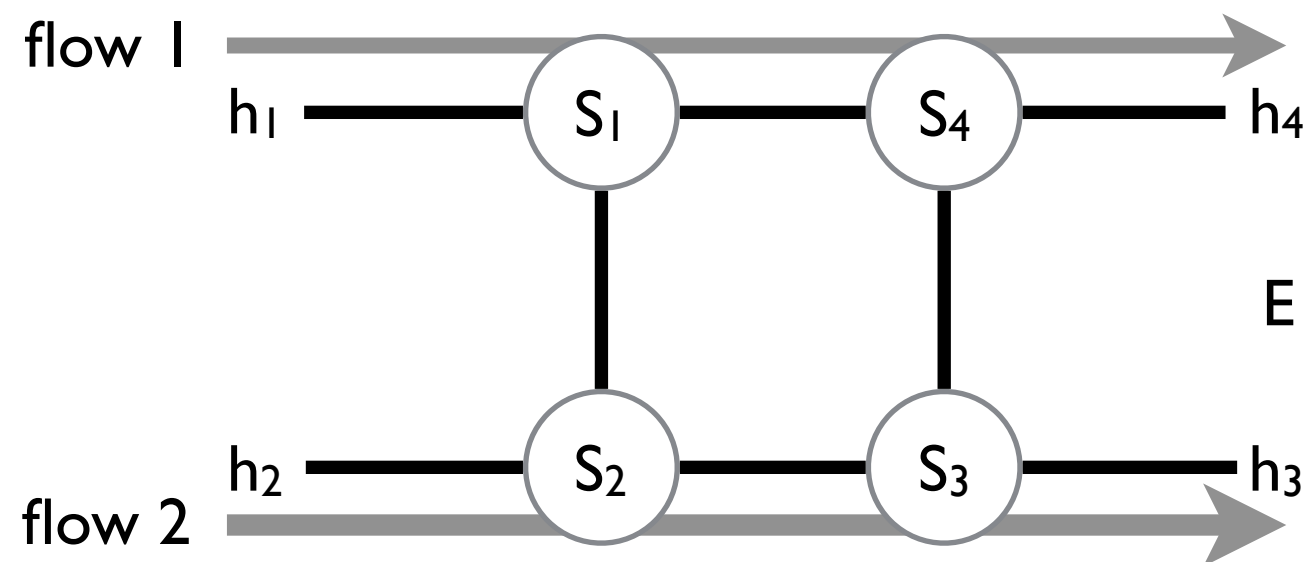
sid	nid
S_1	S_2
S_1	S_3
S_1	h_1

...

configuration

fid	sid	nid
1	S_1	S_4
1	S_4	h_4

...



abstraction: application view

firewall view: monitoring unsafe flows violating
acl policy

```
CREATE VIEW acl_violation AS (  
  SELECT fid  
  FROM rm  
  WHERE FW = 1 AND  
    (src, dst) NOT IN  
    (SELECT end1, end2 FROM acl  
      WHERE allow = 1)  
);
```

```
CREATE TABLE acl (  
  end1 integer, end2 integer, allow integer  
);
```

abstraction: application view

firewall view: monitoring unsafe flows violating
acl policy

```
CREATE VIEW acl_violation AS (  
  SELECT fid  
  FROM rm  
  WHERE FW = 1 AND  
    (src, dst) NOT IN  
    (SELECT end1, end2 FROM acl  
      WHERE allow = 1)  
);
```

```
CREATE TABLE acl (  
  end1 integer, end2 integer, allow integer  
);
```

firewall control: repairing violation

```
CREATE RULE acl_repair AS  
  ON DELETE TO acl_violation  
  DO INSTEAD  
    DELETE FROM rm WHERE fid = OLD.fid;
```

abstraction: application view

firewall view: monitoring unsafe flows violating
acl policy

```
CREATE VIEW acl_violation AS (  
  SELECT fid  
  FROM rm  
  WHERE FW = 1 AND  
    (src, dst) NOT IN  
    (SELECT end1, end2 FROM acl  
      WHERE allow = 1)  
);
```

```
CREATE TABLE acl (  
  end1 integer, end2 integer, allow integer  
);
```

firewall control: repairing violation

```
CREATE RULE acl_repair AS  
  ON DELETE TO acl_violation  
  DO INSTEAD  
    DELETE FROM rm WHERE fid = OLD.fid;
```

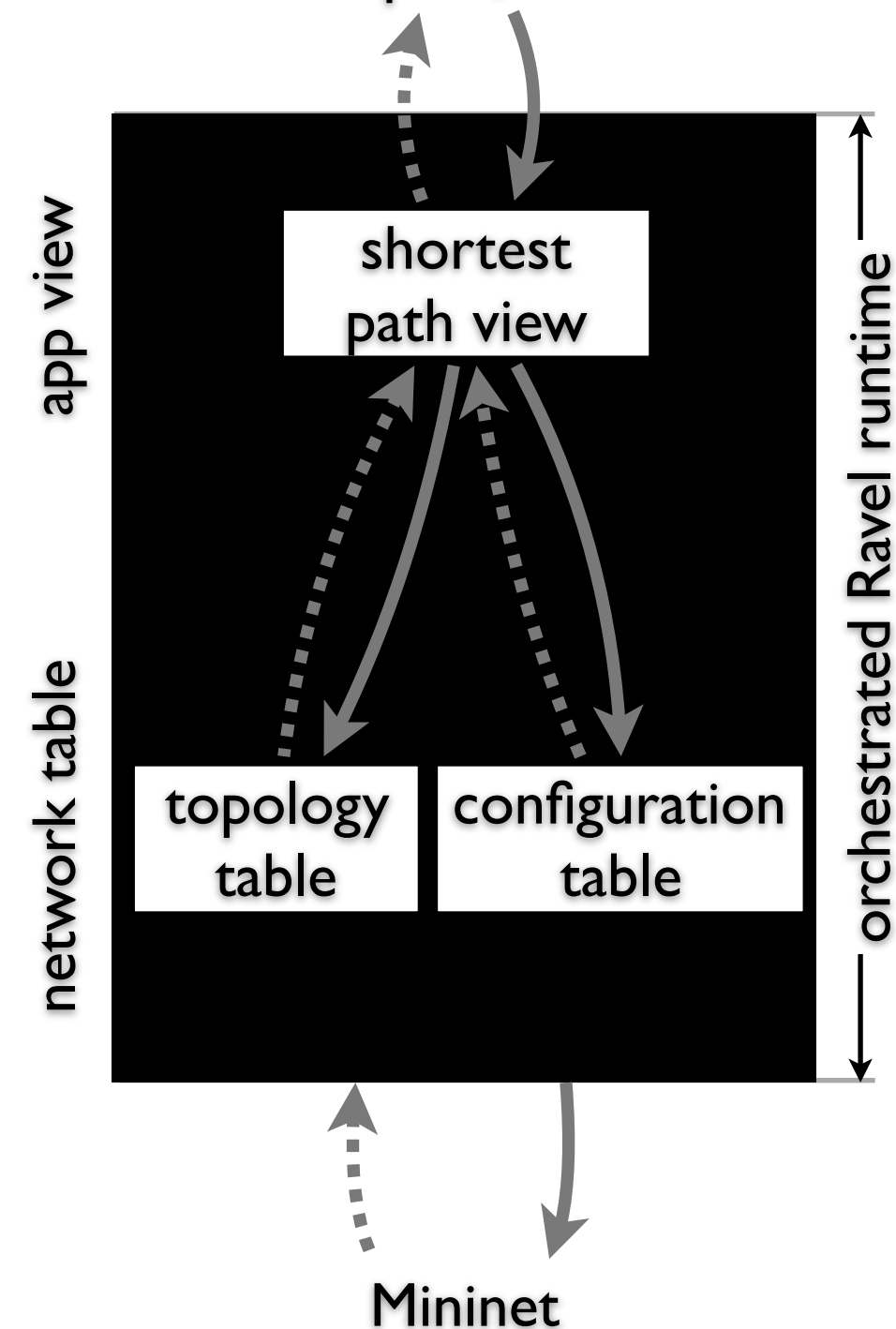
many more

- routing, stateful firewall, service chain policy between subdomains ...

orchestration across representations

routing app: check
broken path, re-route

SQL rule:
upon broken path, re-route

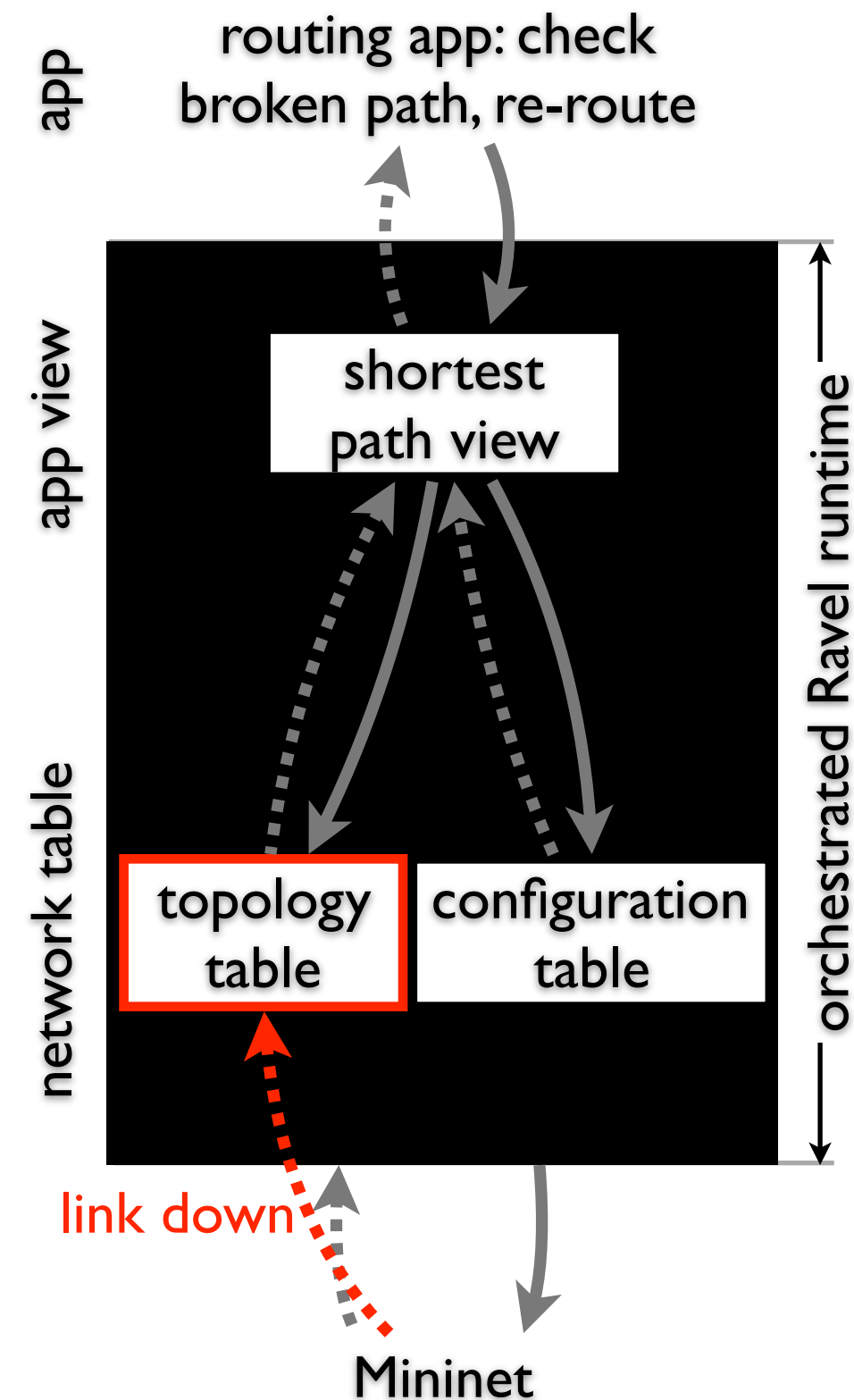


shortest path	

topology		

configuration		

orchestration across representations



SQL rule:
upon broken path, re-route

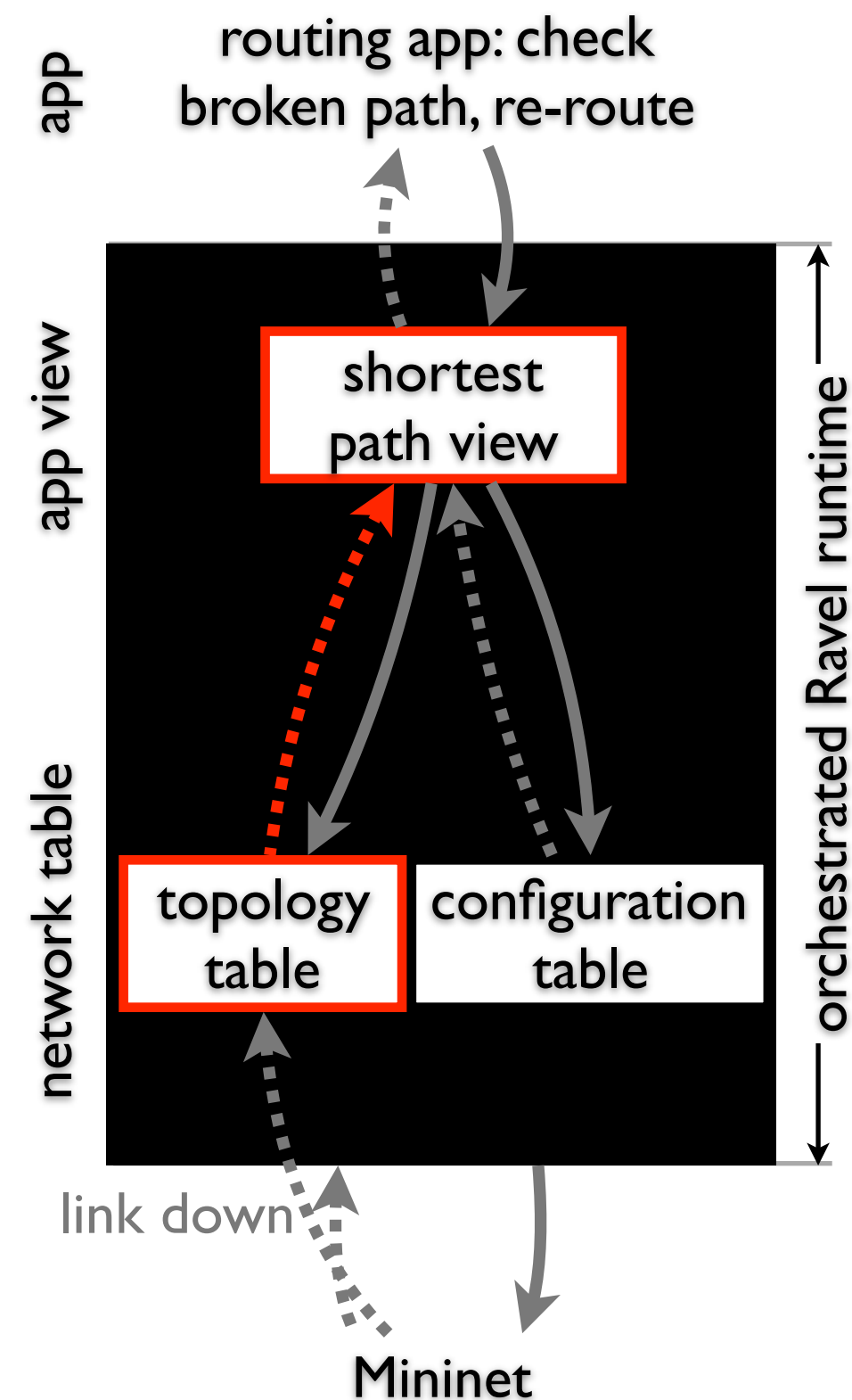
shortest path	

	topology		
	sid	nid	active
-	172	39	1
+	172	39	0

configuration		

Mininet link (172,39) down

orchestration across representations



SQL rule:
upon broken path, re-route

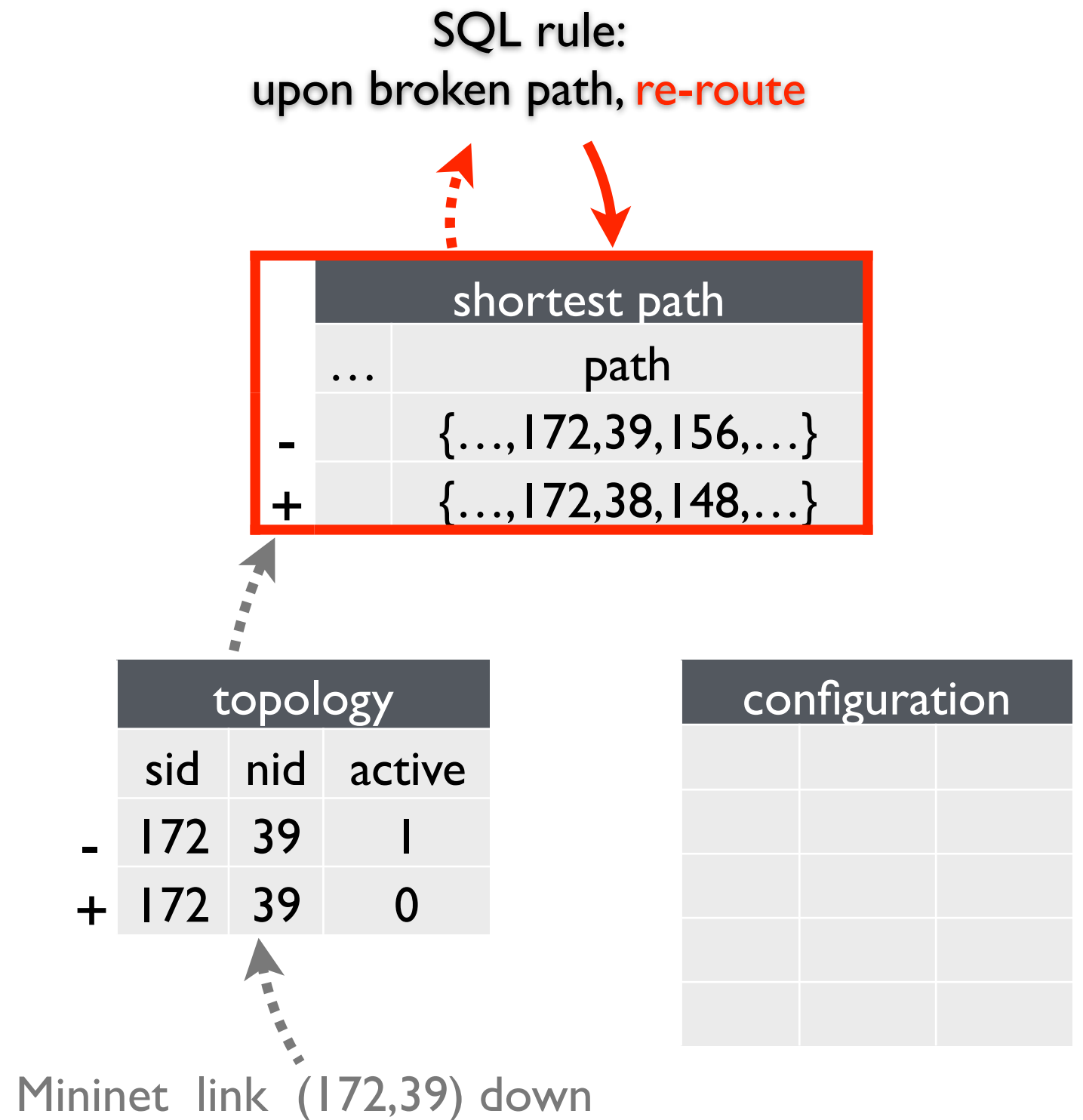
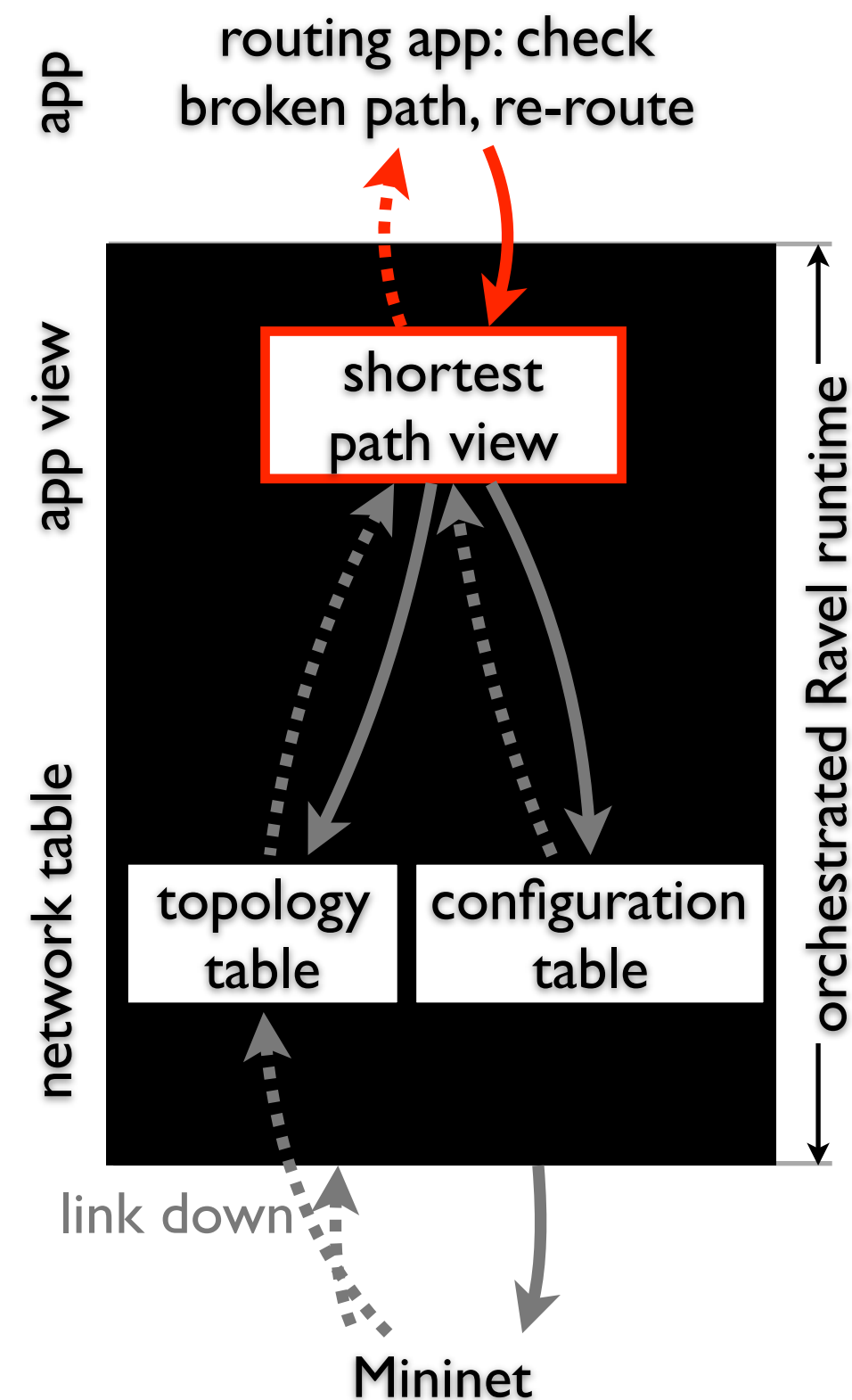
	shortest path	
	...	path
-	{..., 172, 39, 156, ...}	

	topology		
	sid	nid	active
-	172	39	1
+	172	39	0

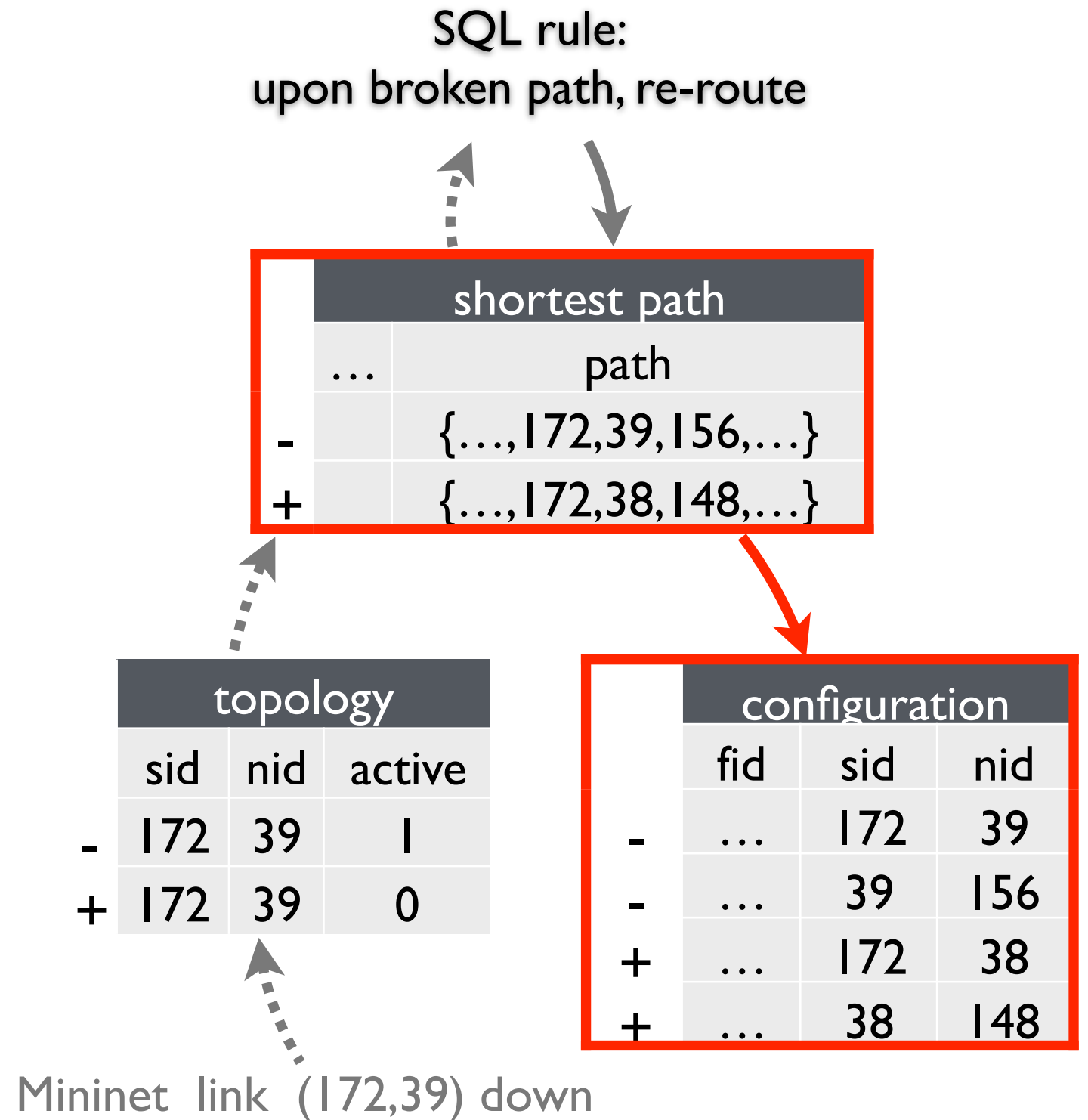
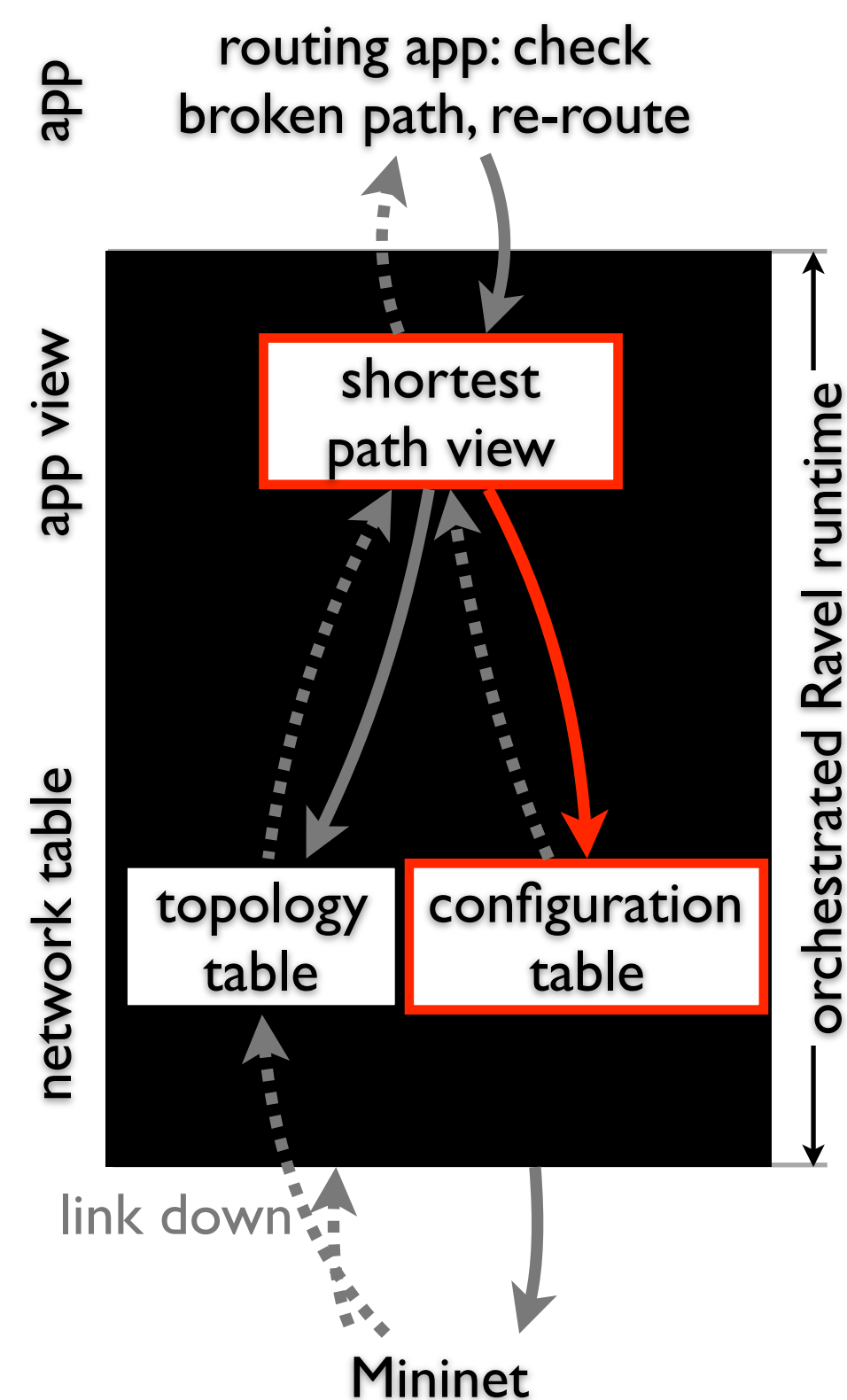
configuration		

Mininet link (172,39) down

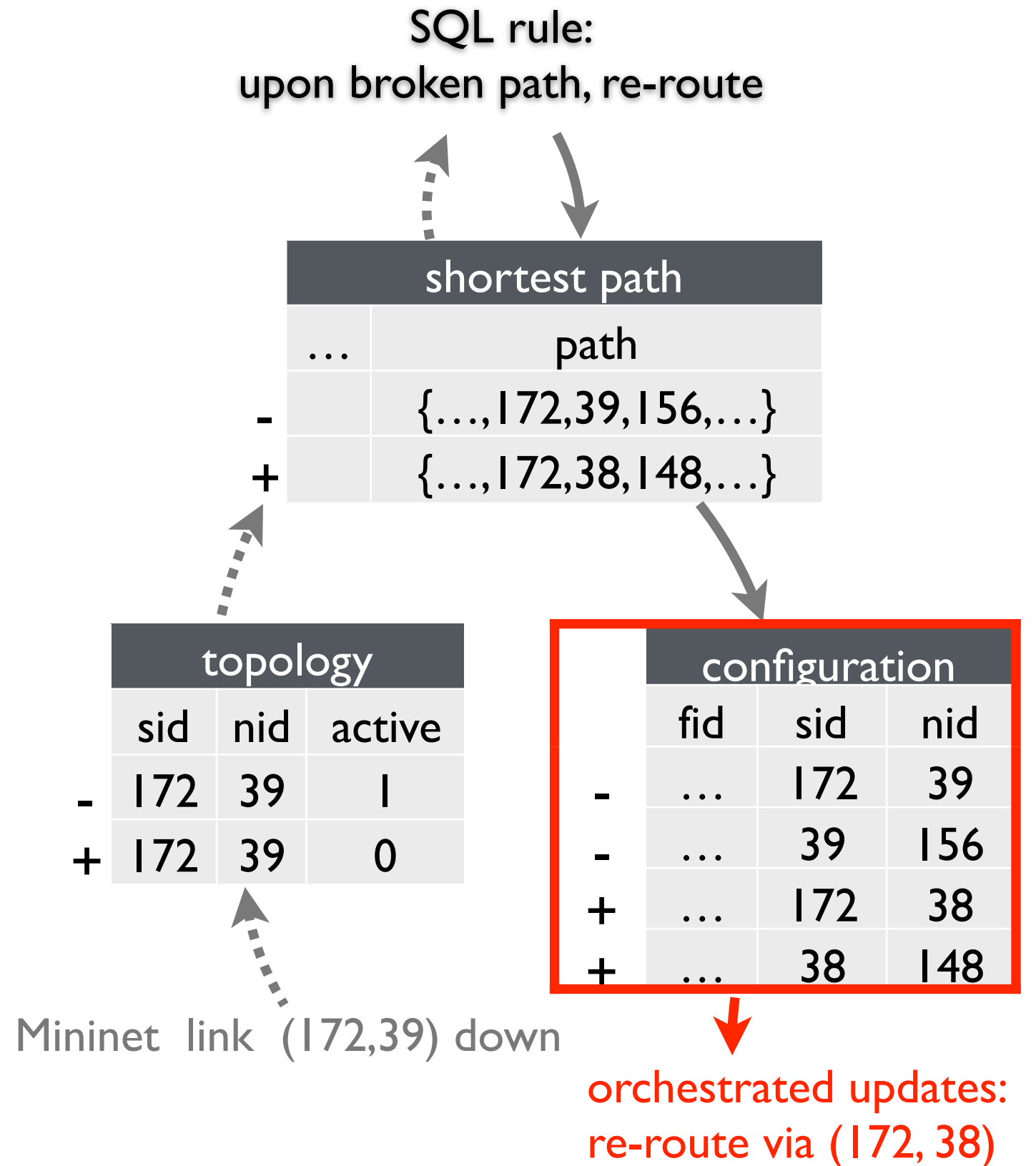
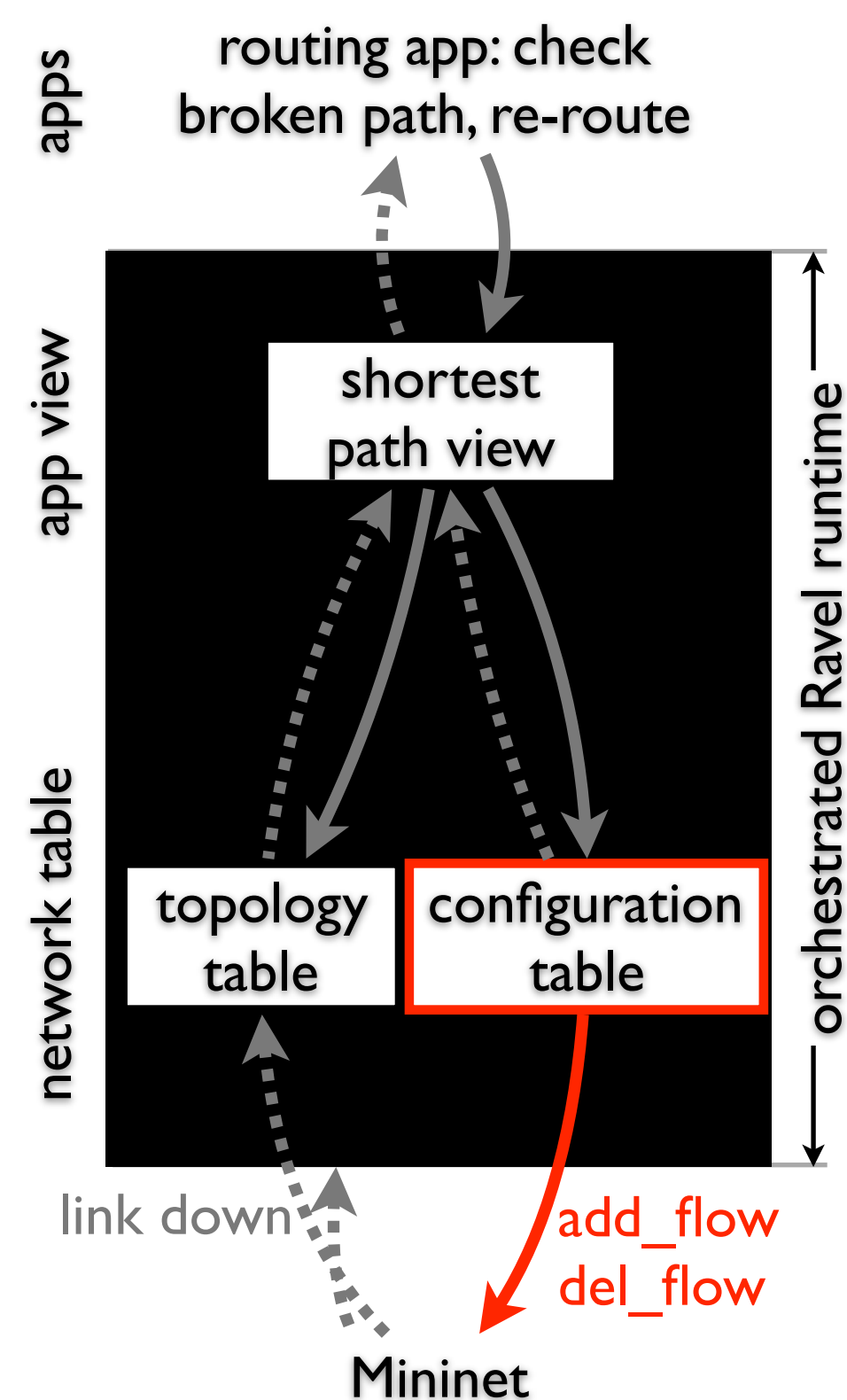
orchestration across representations



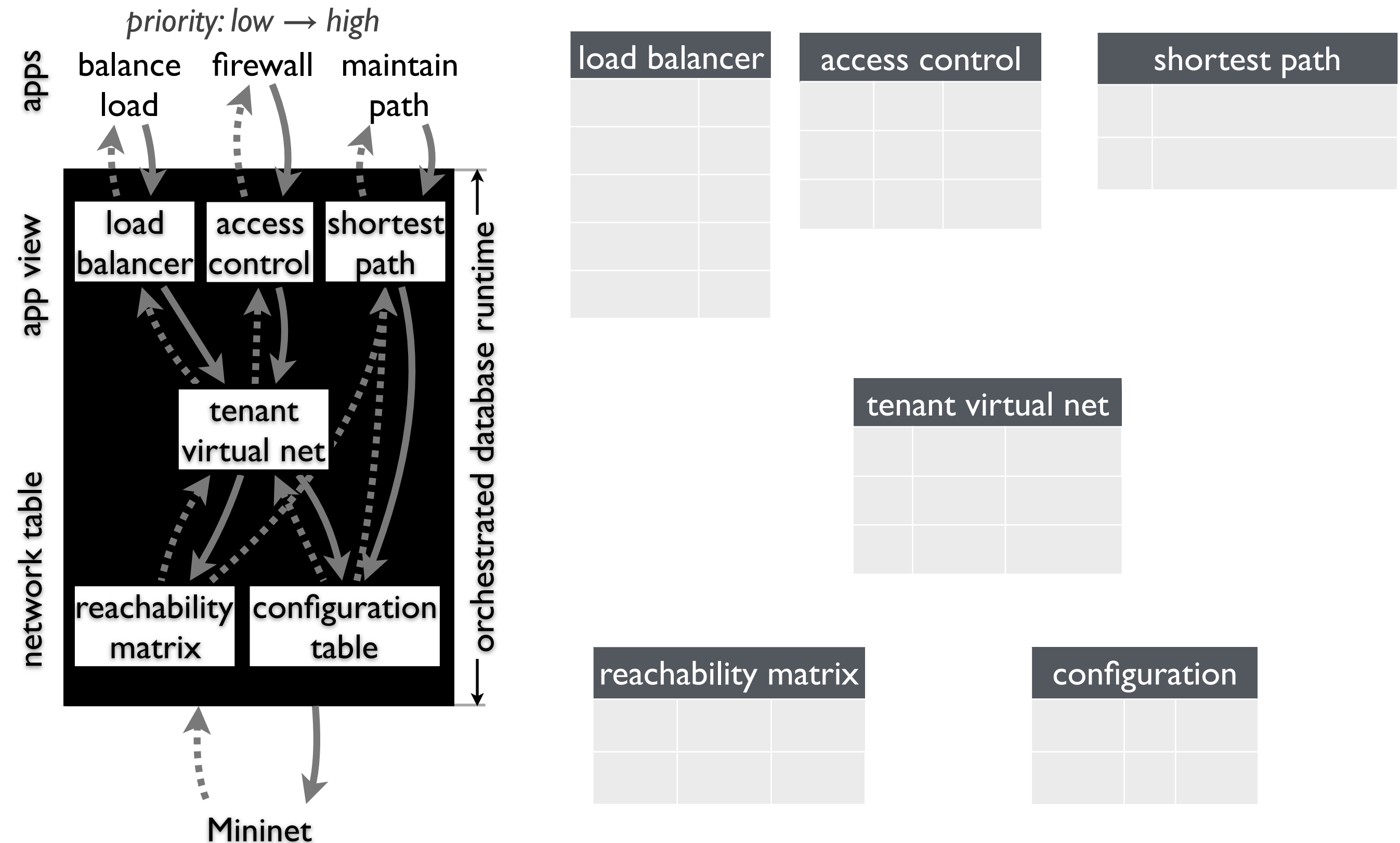
orchestration across representations



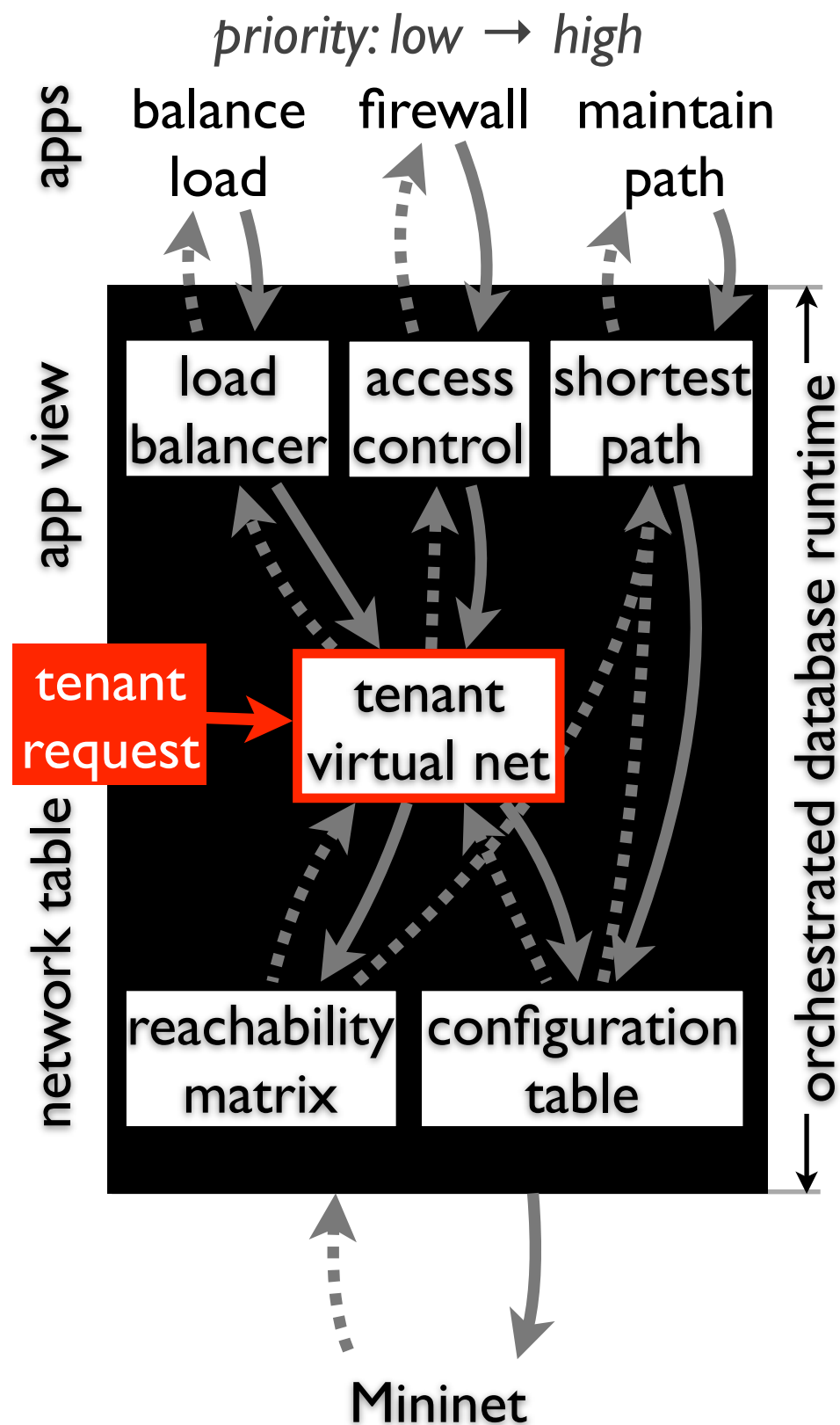
orchestration across representations



orchestration across applications



orchestration across applications



load balancer	

access control		

shortest path	

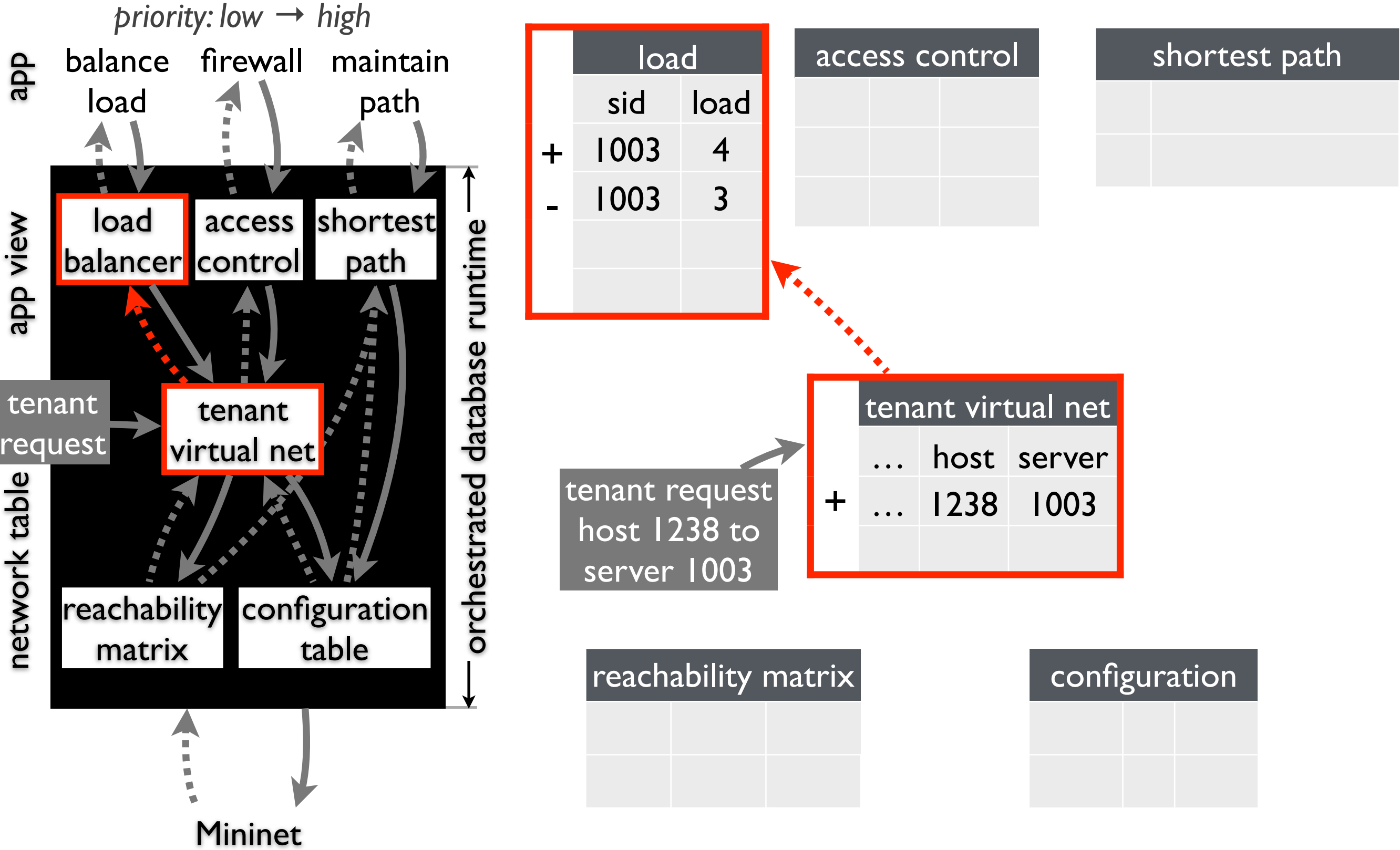
tenant request
host 1238 to
server 1003

tenant virtual net		
...	host	server
+	...	1238 1003

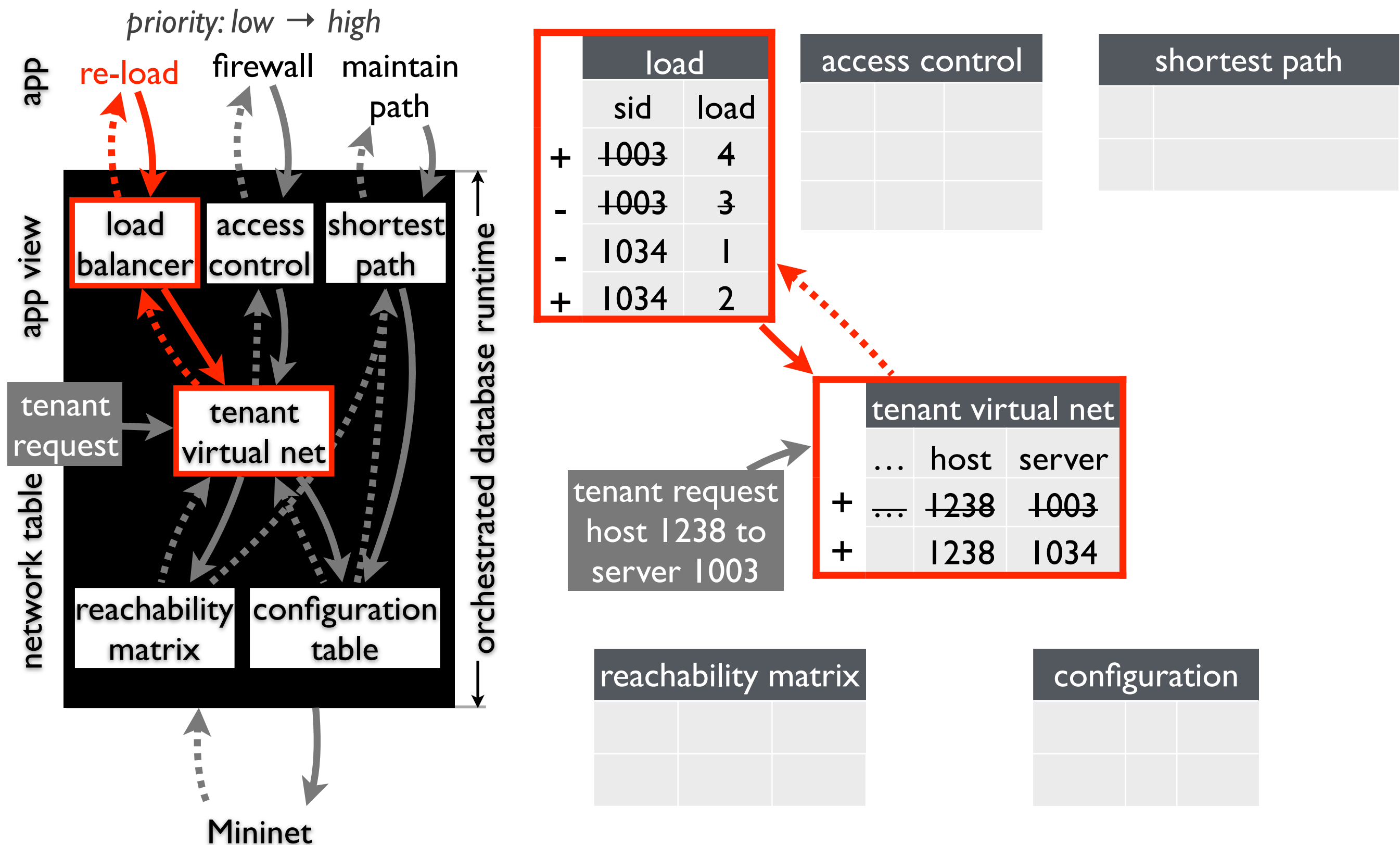
reachability matrix		

configuration		

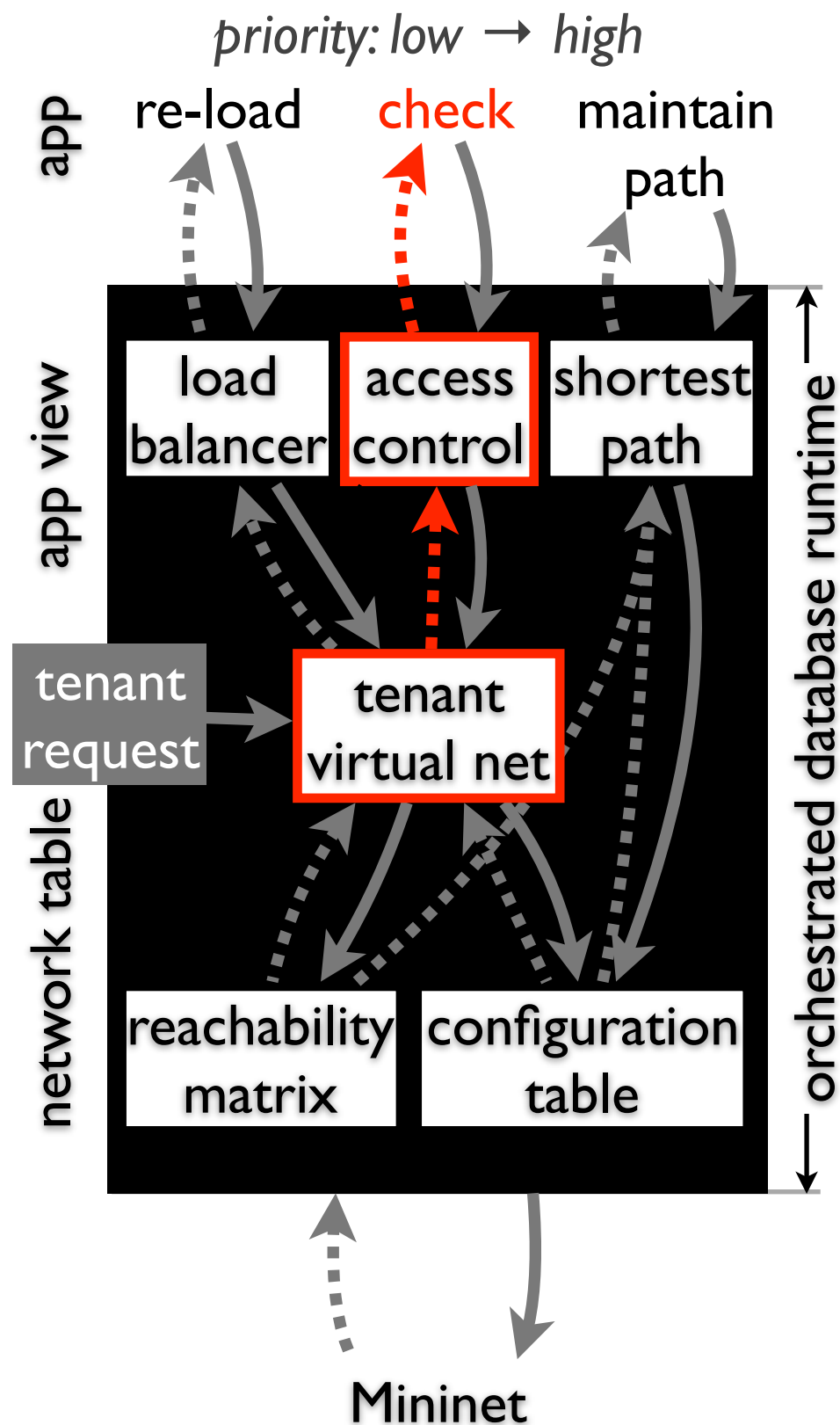
orchestration across applications



orchestration across applications



orchestration across applications



	load	
	sid	load
+	1003	4
-	1003	3
-	1034	1
+	1034	2

access control		
src	dst	allow
1238	1034	1
1238	1003	0

shortest path

tenant request
host 1238 to
server 1003

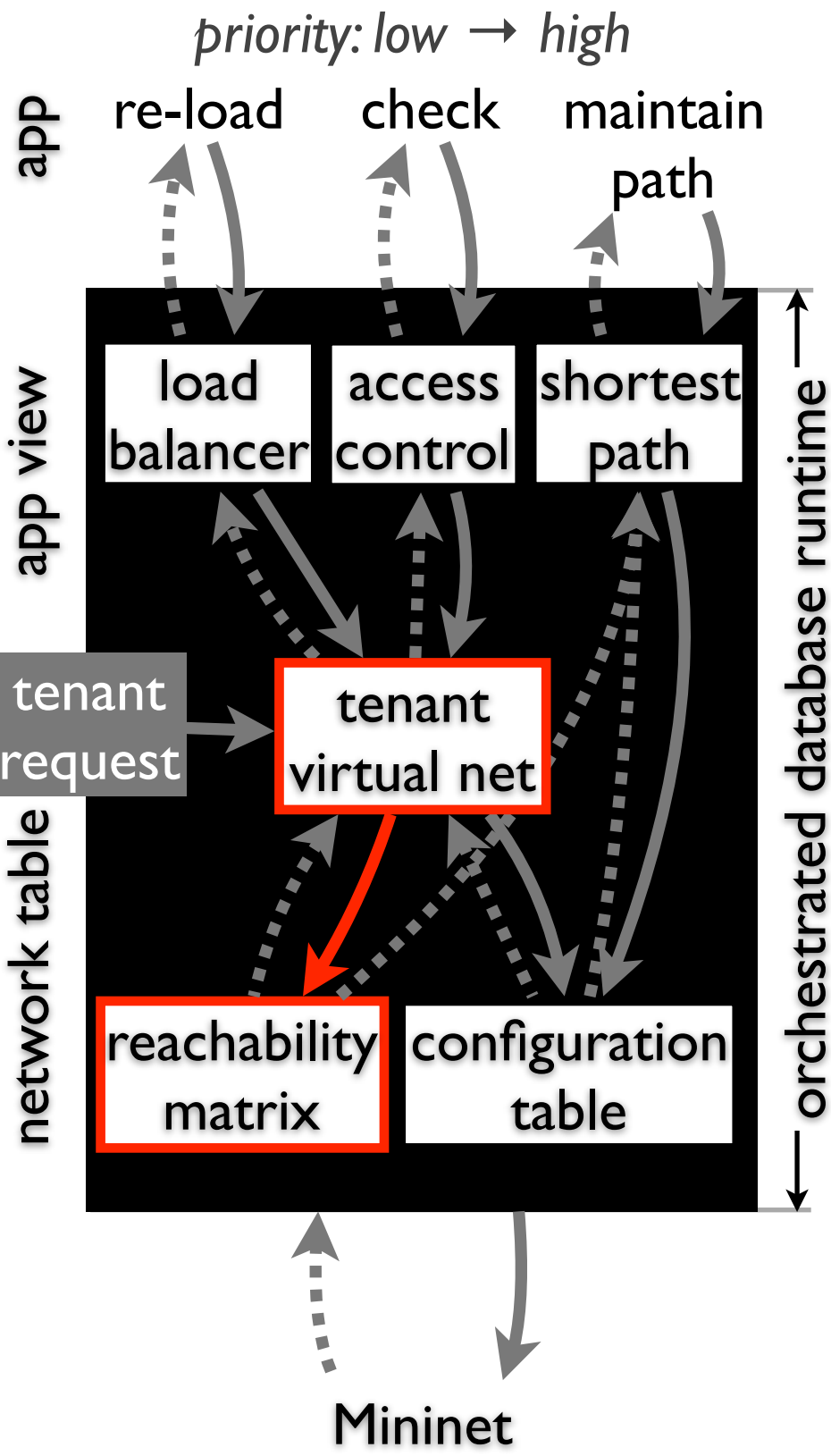
tenant virtual net			
	...	host	server
+	...	1238	1003
+		1238	1034

reachability matrix

A diagram showing a configuration table. The table has 2 rows and 3 columns. The first row is labeled 'configuration' in a dark grey header. The second row is empty. The table is represented by a grid of cells.

configuration		

orchestration across applications



load	
sid	load
+ 1003	4
- 1003	3
- 1034	1
+ 1034	2

access control		
src	dst	allow
1238	1034	1
1238	1003	0

shortest path		

tenant request host 1238 to server 1003

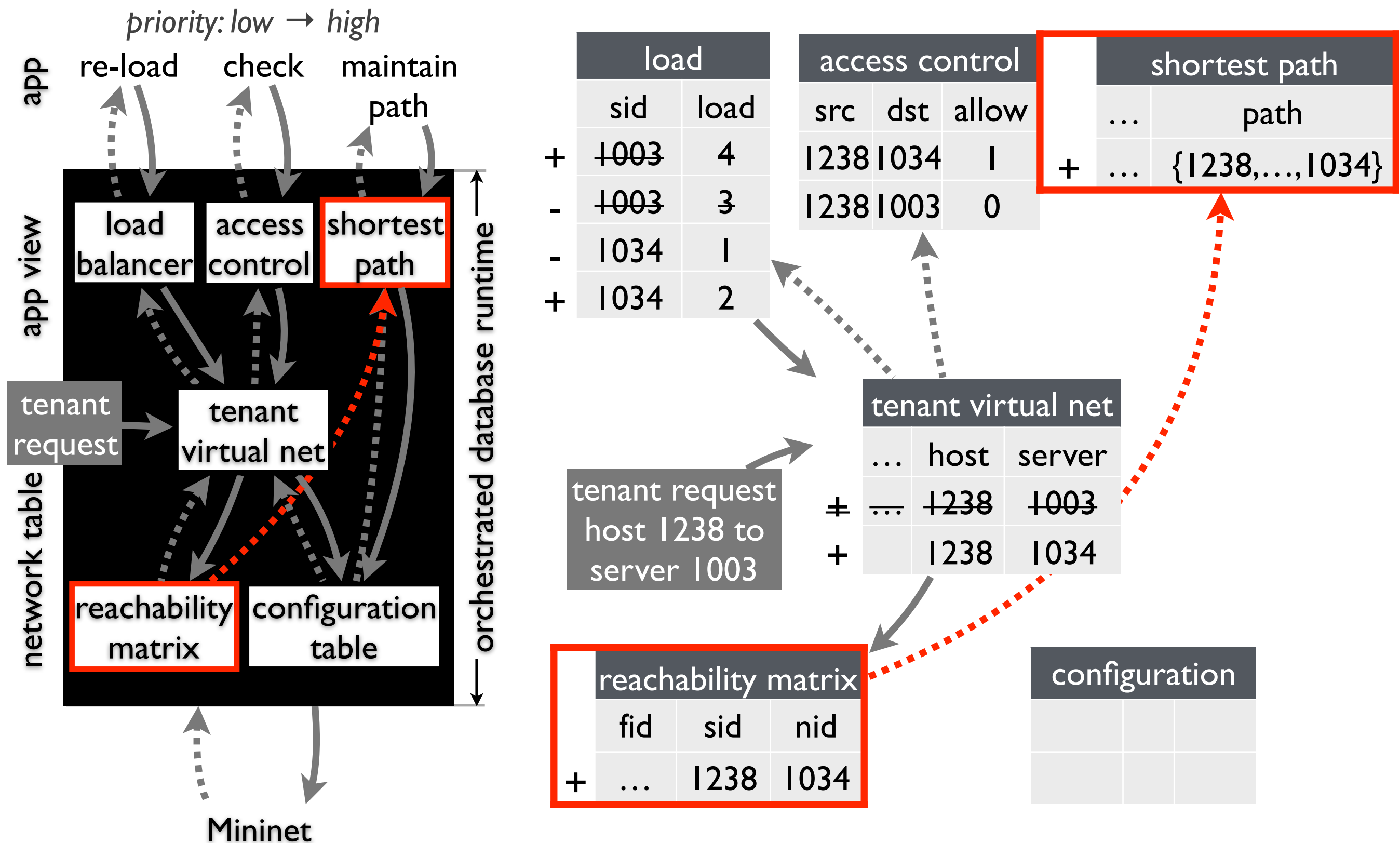
tenant virtual net			
	...	host	server
+ ...	1238	1003	
+ ...	1238	1034	

reachability matrix			
	fid	sid	nid
+ ...		1238	1034

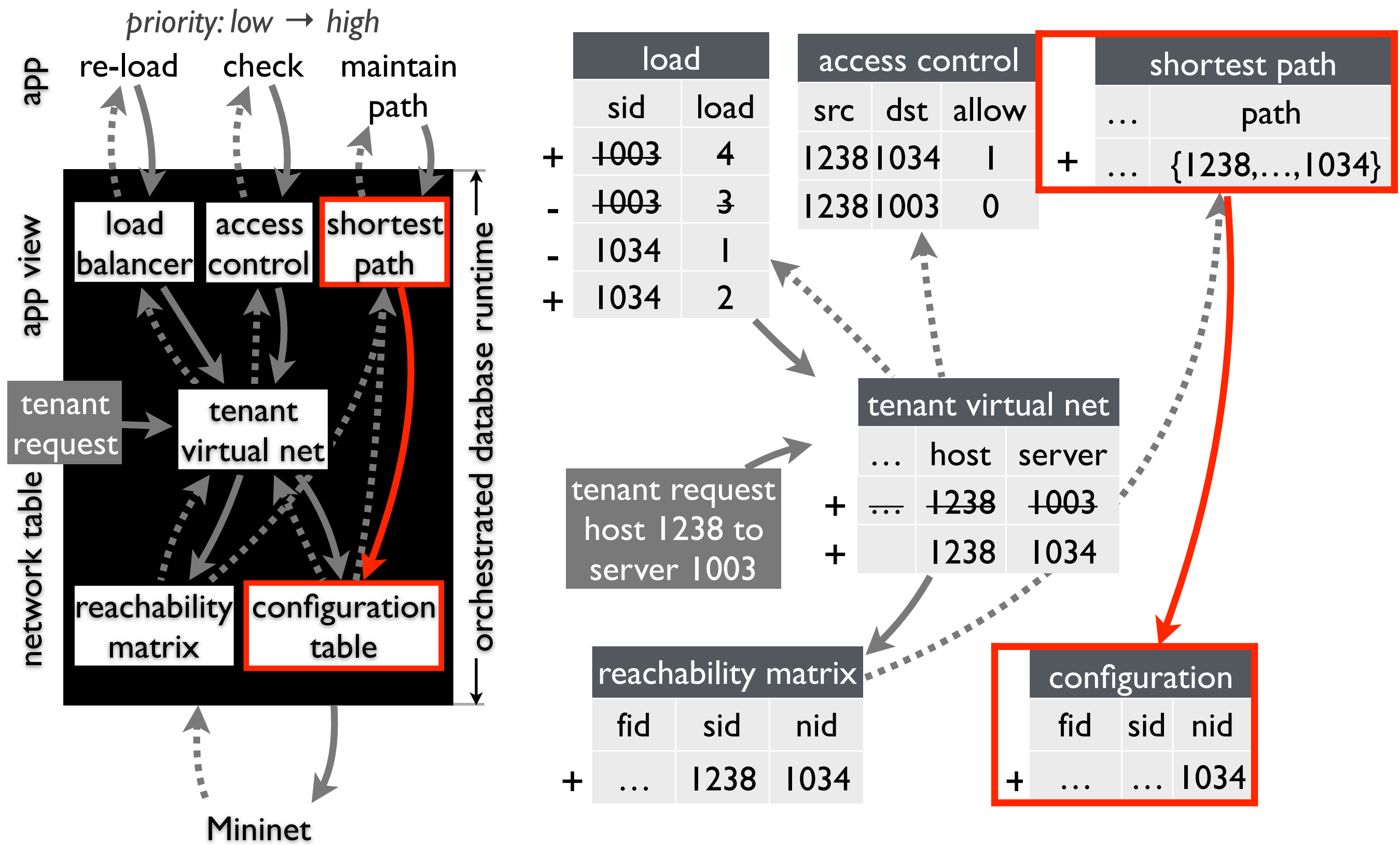
configuration		

The right side of the diagram shows data tables and request flow. It includes a 'load' table with source IDs and their loads, an 'access control' table with source-destination pairs and allow flags, and a 'shortest path' table. A 'tenant request' for host 1238 to server 1003 is shown. This request is processed by the 'tenant virtual net' table, which maps the request to specific host and server IDs. The 'reachability matrix' table then shows the mapping of the request to specific source, destination, and node IDs. The 'configuration' table is also shown. Arrows indicate the flow of data and control between these tables and the central orchestration components.

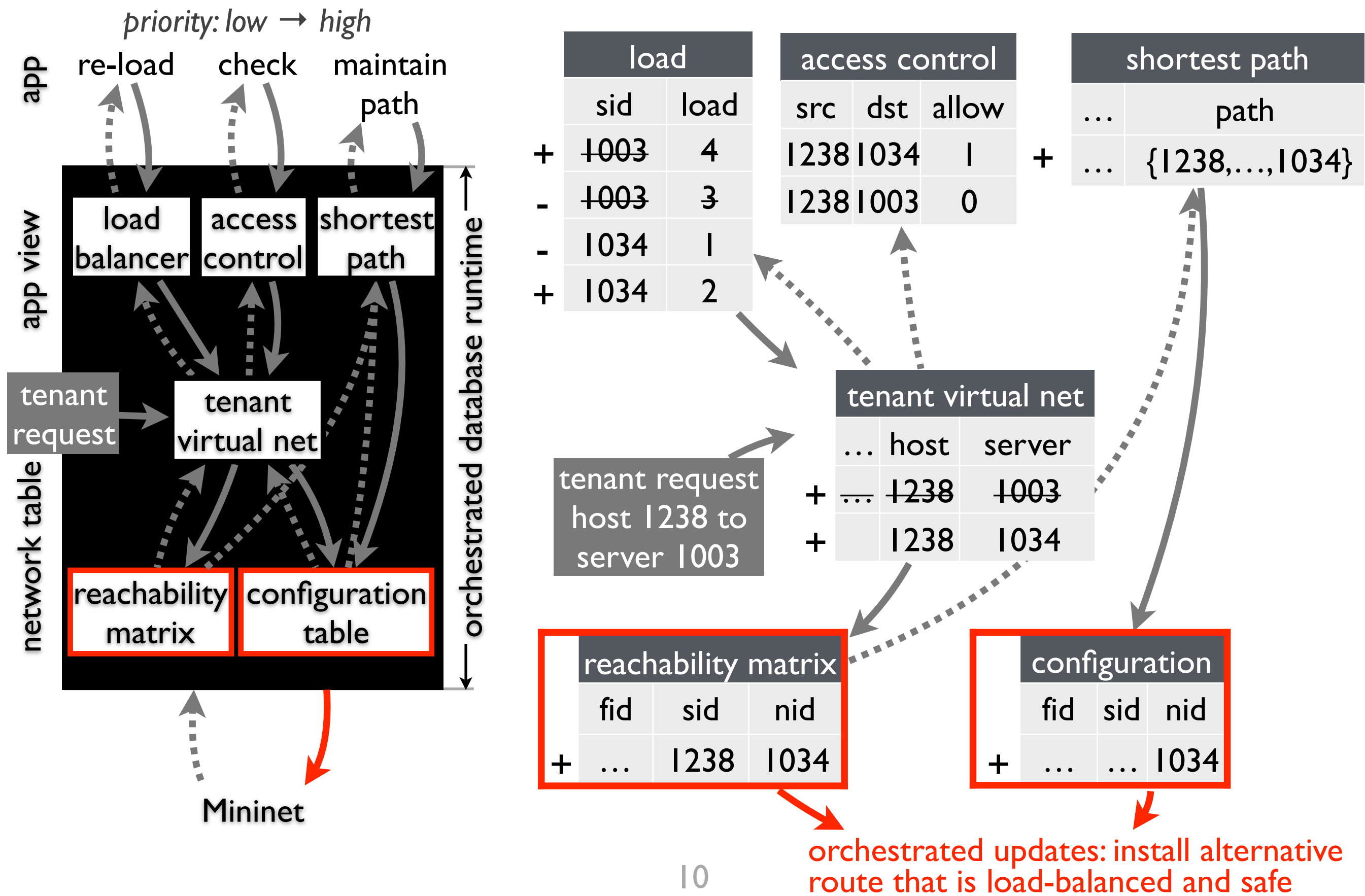
orchestration across applications



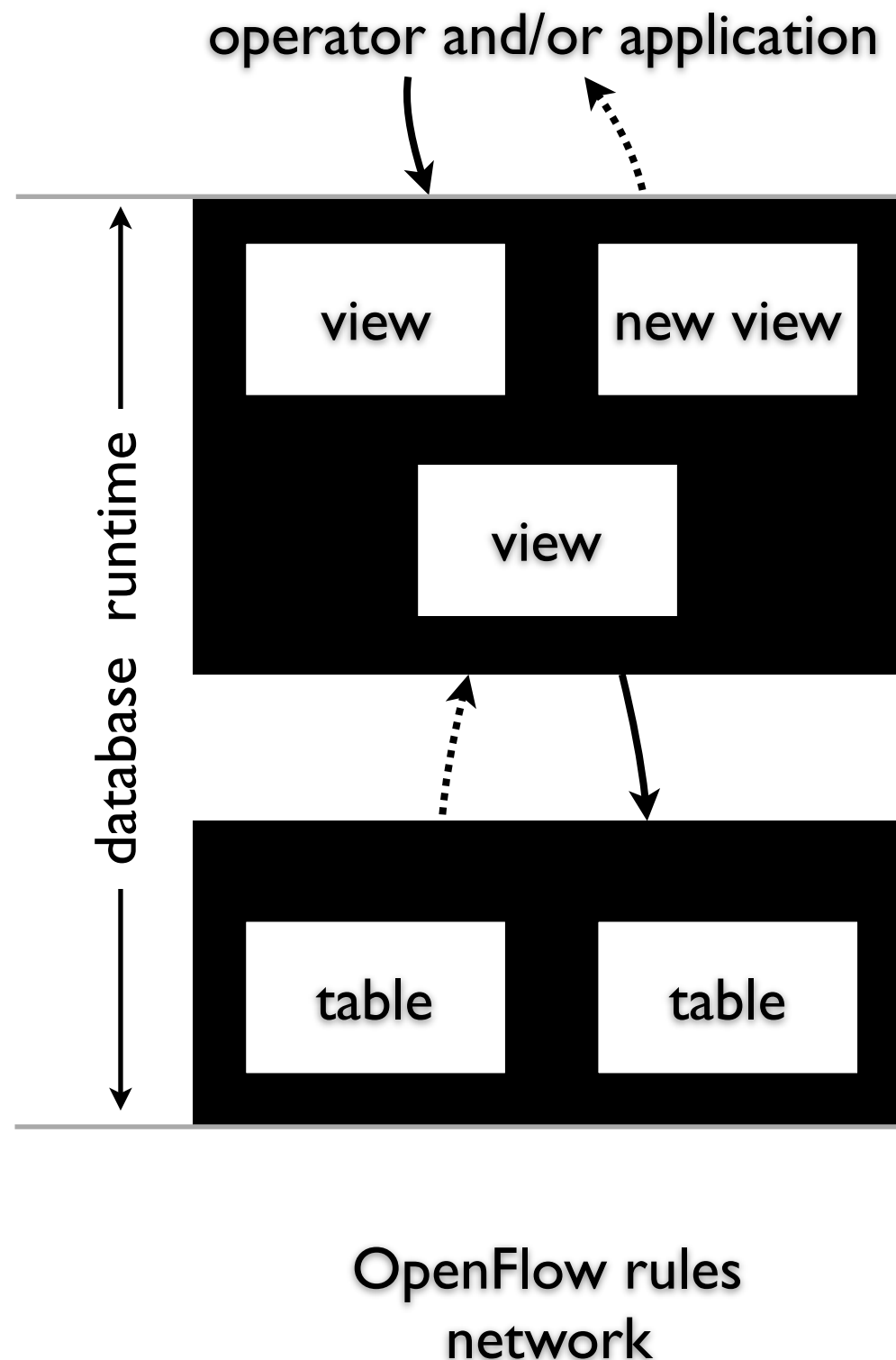
orchestration across applications



orchestration across applications



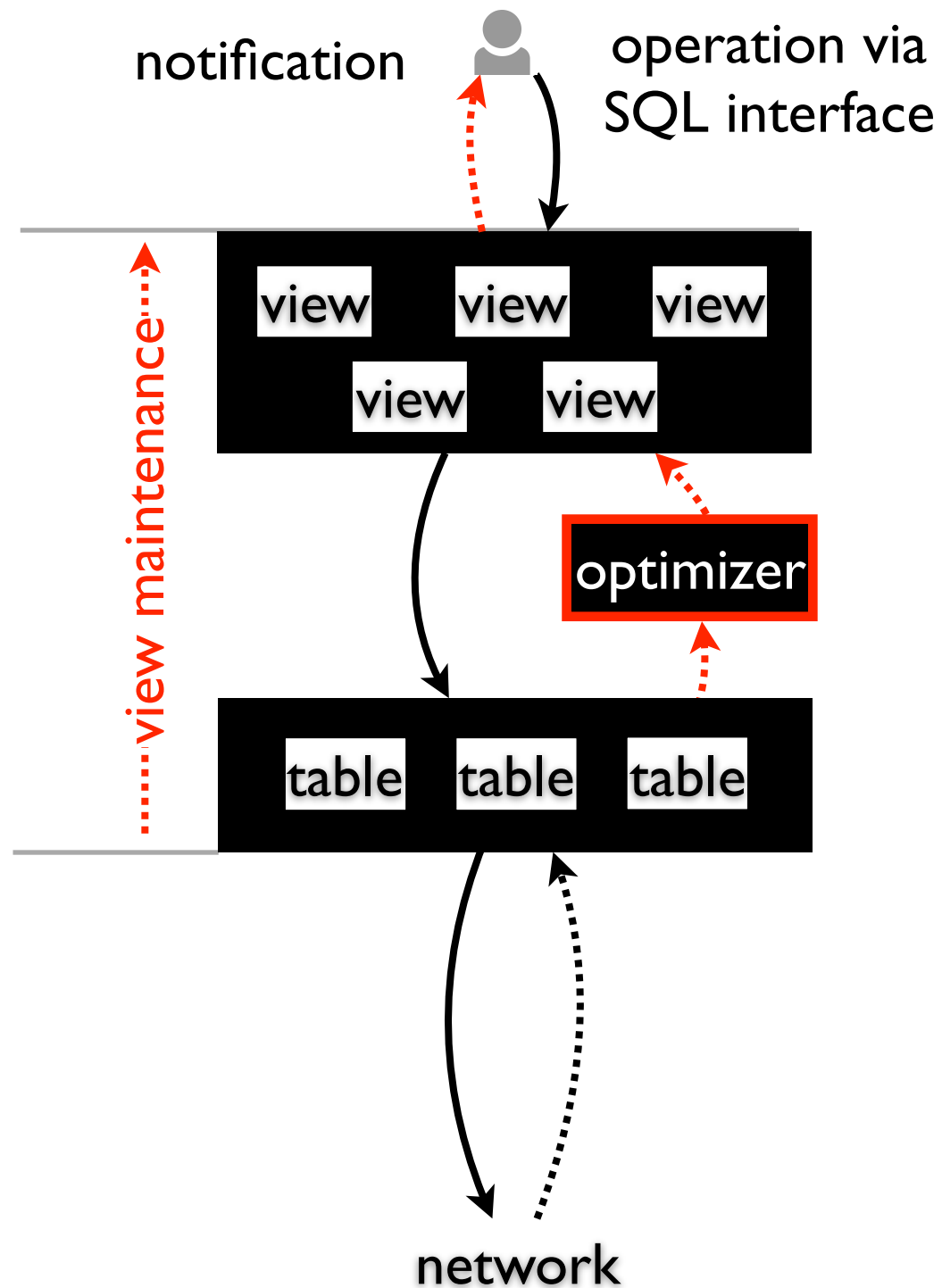
achieving *Ravel* advantages



attractive features

- ad-hoc programmable abstraction via views
- orchestration across abstractions via view mechanism
- orchestration across applications via data mediation
- network control via SQL

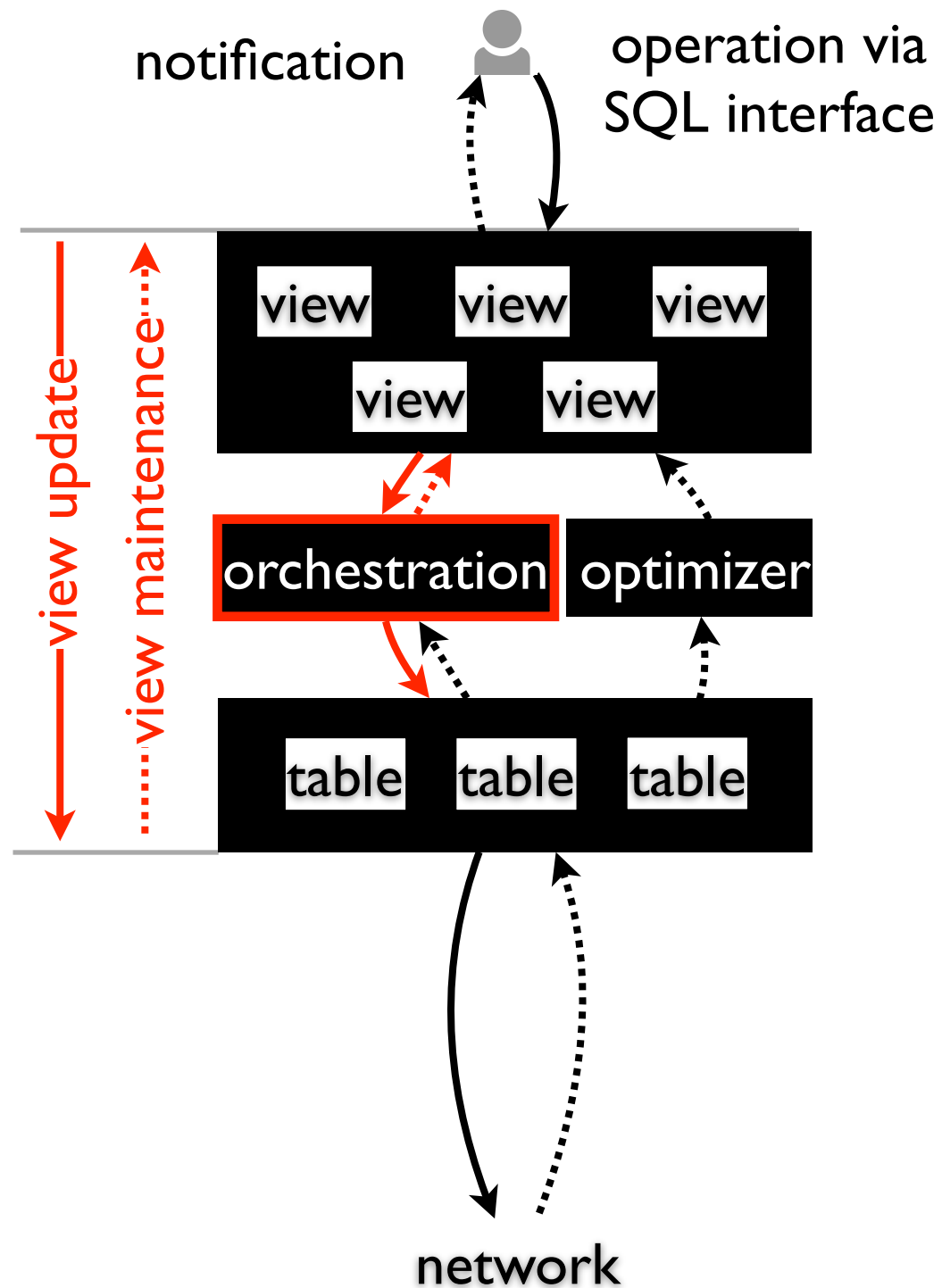
runtime



ad-hoc programmable abstraction via views

- challenge: inefficient user view
- solution: optimizer
 - materialize user view with fast maintenance algorithm
 - (one order of magnitude) faster access with small maintenance overhead (.01~10ms)

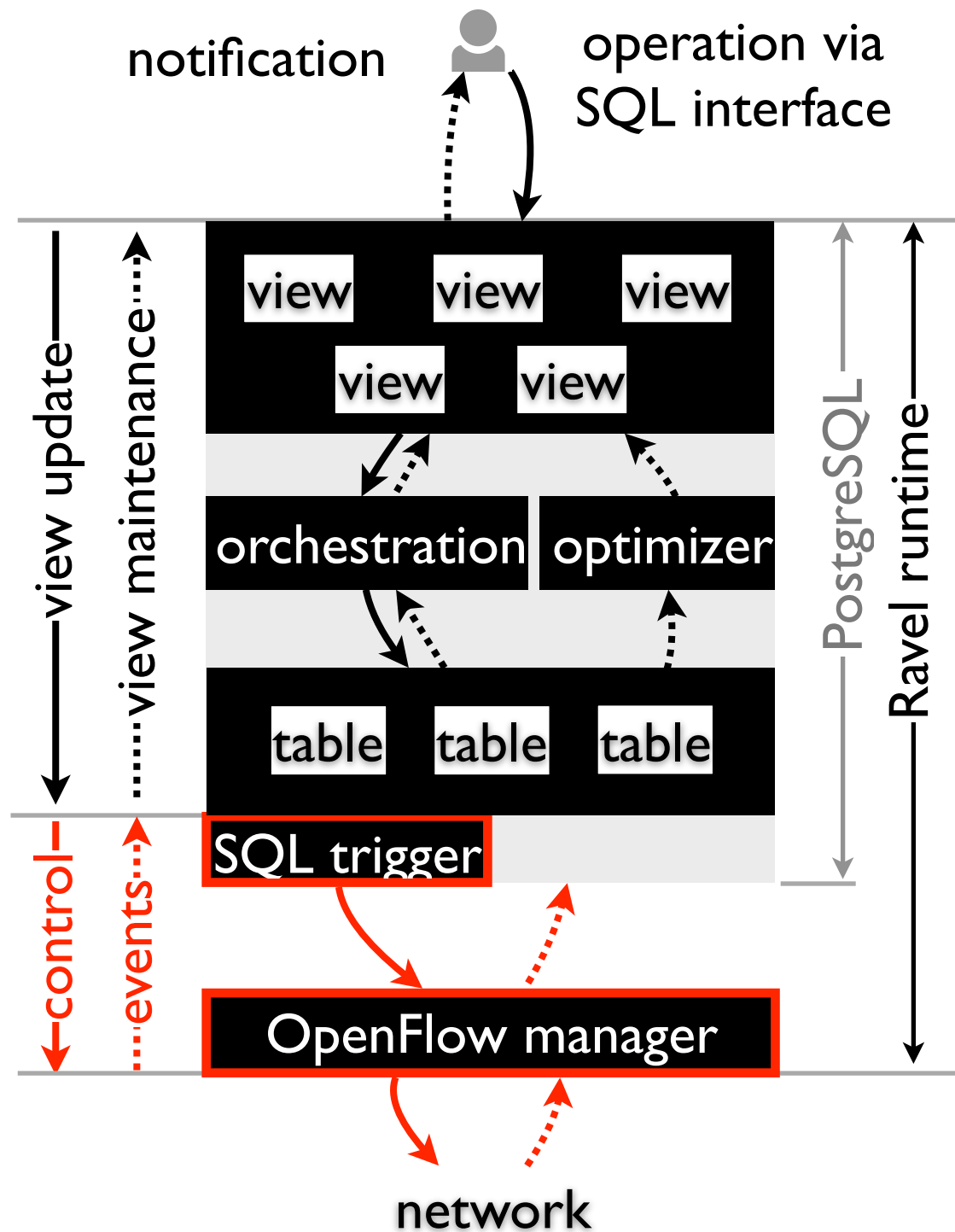
runtime



orchestration across applications

- challenge: database lacking inter-view support
- solution: mediation protocol
 - translate app priority into view updates that dynamically merge into a coherent data plane

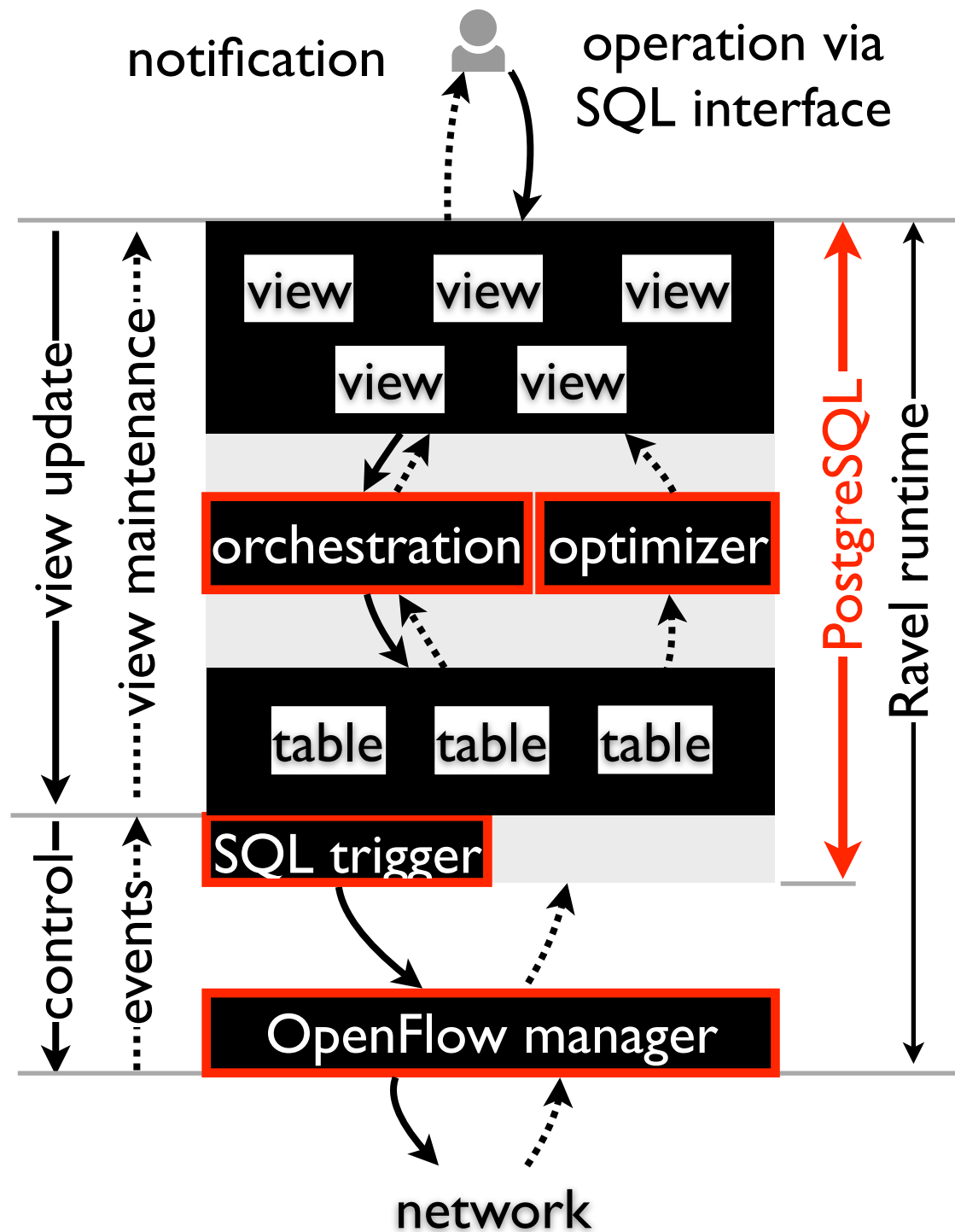
runtime



SDN control via SQL

- challenge: database lacks connection to network data plane
- solution: SQL trigger + OF manager

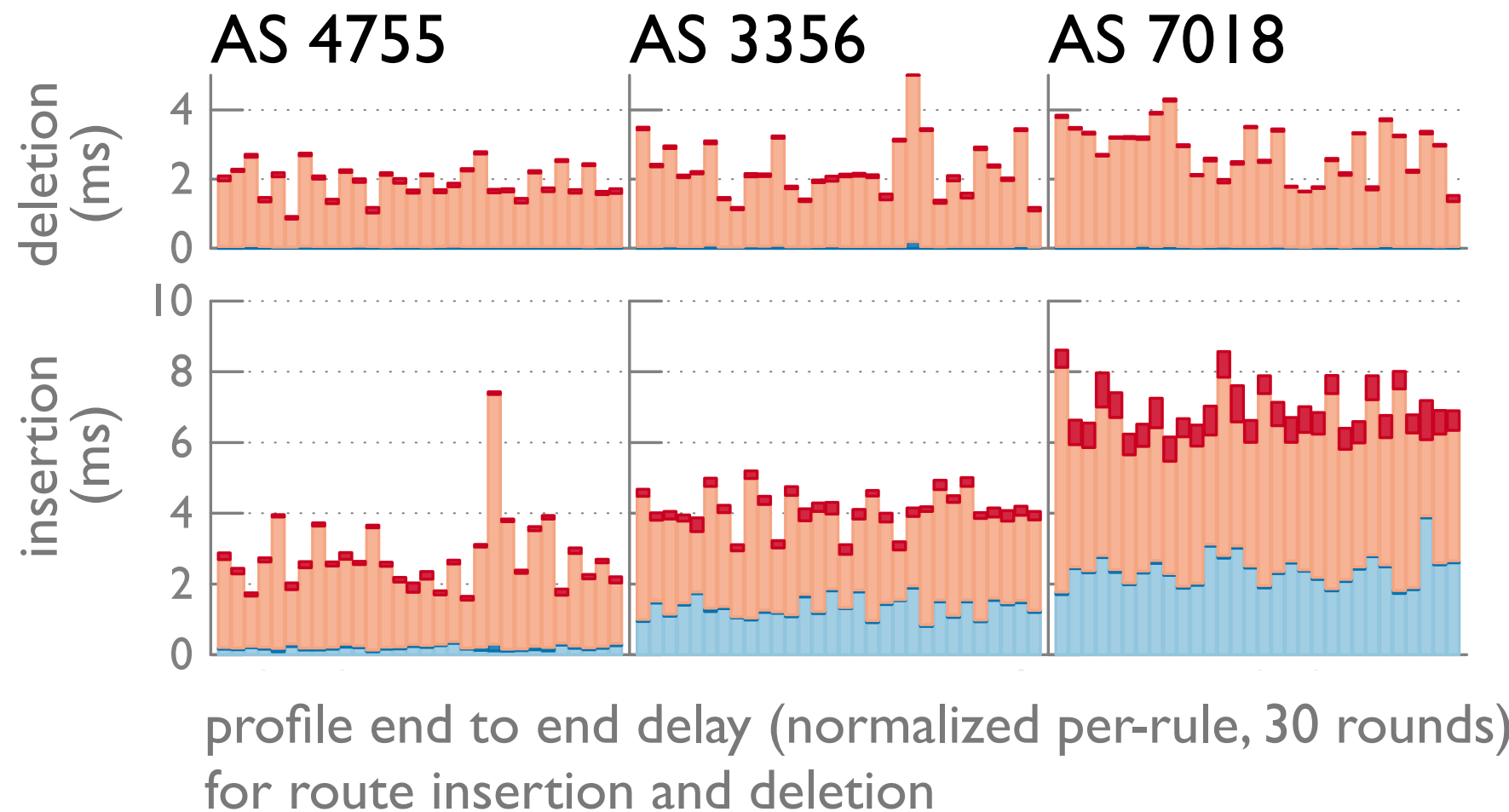
runtime



a high-performance runtime





- PostgreSQL
- orchestration
- optimizer
- SQL trigger and OF manager

evaluation

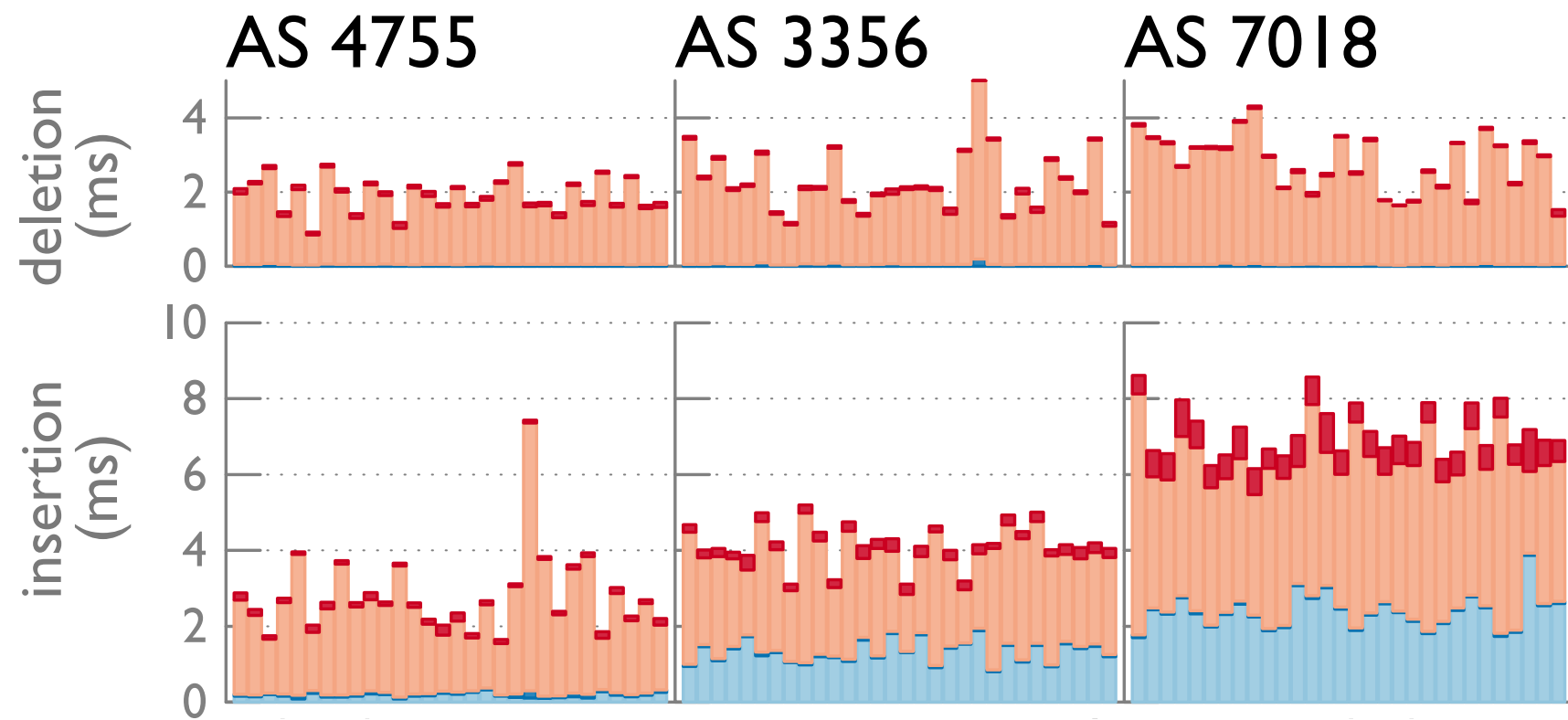


Rocketfuel ISP topology

AS#	nodes	links
4755	142	258
3356	1772	13640
7018	25382	11292

compute path 
lookup ports 
write to table 
trigger/rule 

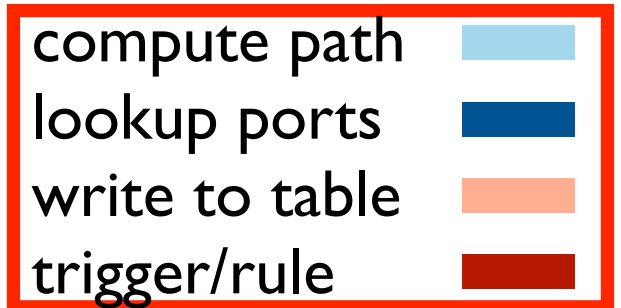
evaluation



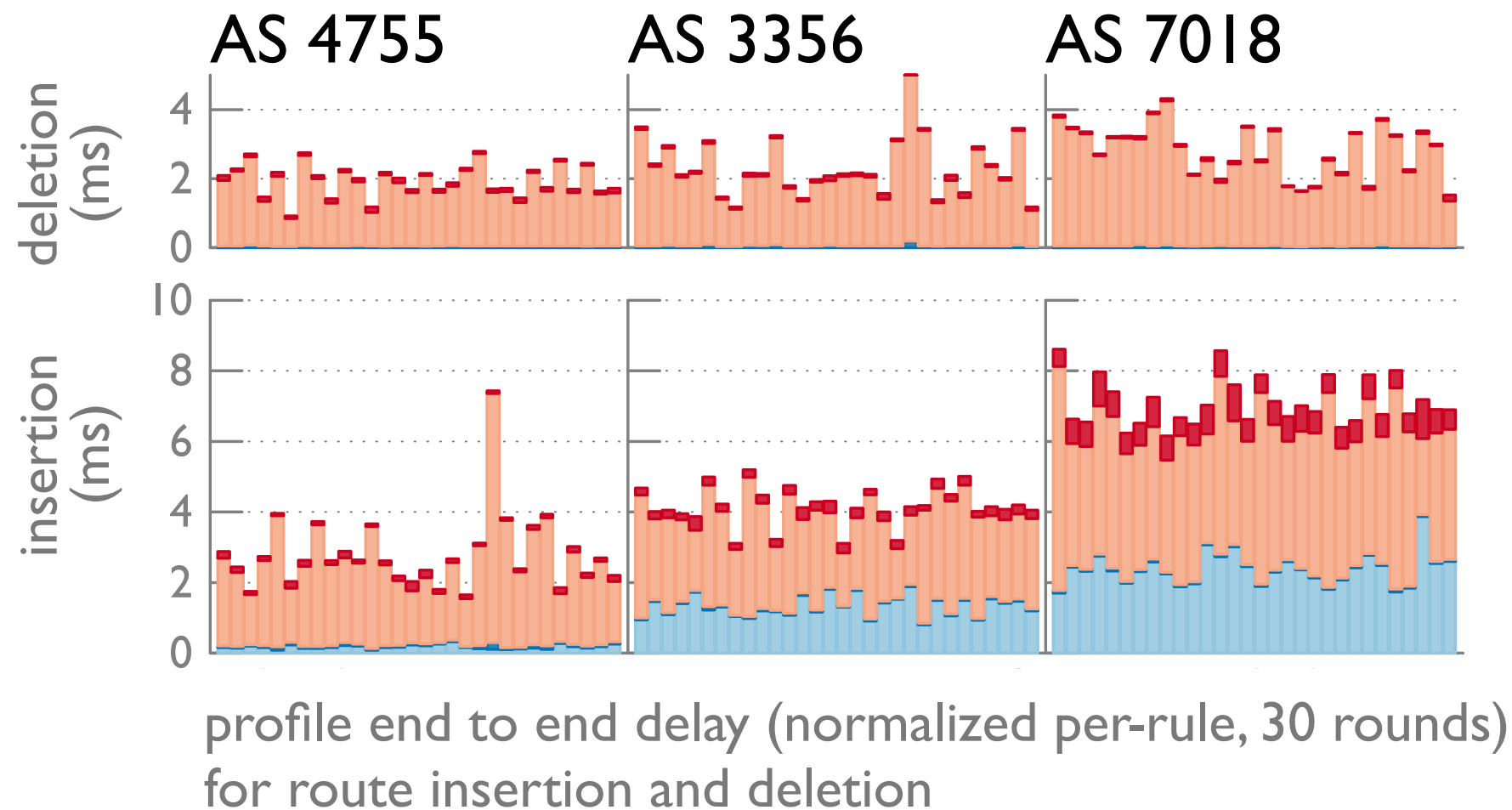
profile end to end delay (normalized per-rule, 30 rounds)
for route insertion and deletion

Rocketfuel ISP topology

AS#	nodes	links
4755	142	258
3356	1772	13640
7018	25382	11292







evaluation

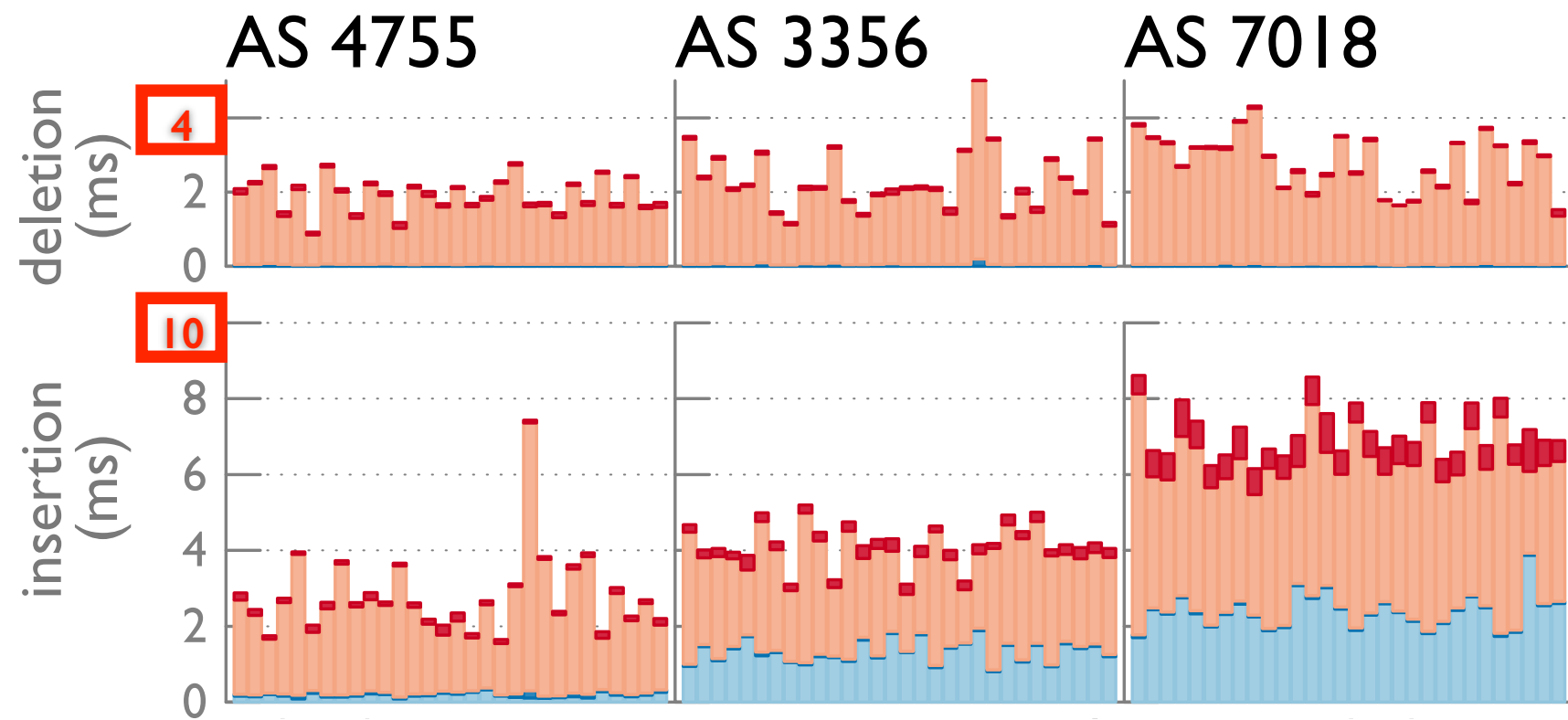


Rocketfuel ISP topology

AS#	nodes	links
4755	142	258
3356	1772	13640
7018	25382	11292

compute path 
lookup ports 
write to table 
trigger/rule 


evaluation





profile end to end delay (normalized per-rule, 30 rounds)
for route insertion and deletion


Rocketfuel ISP topology

AS#	nodes	links
4755	142	258
3356	1772	13640
7018	25382	11292

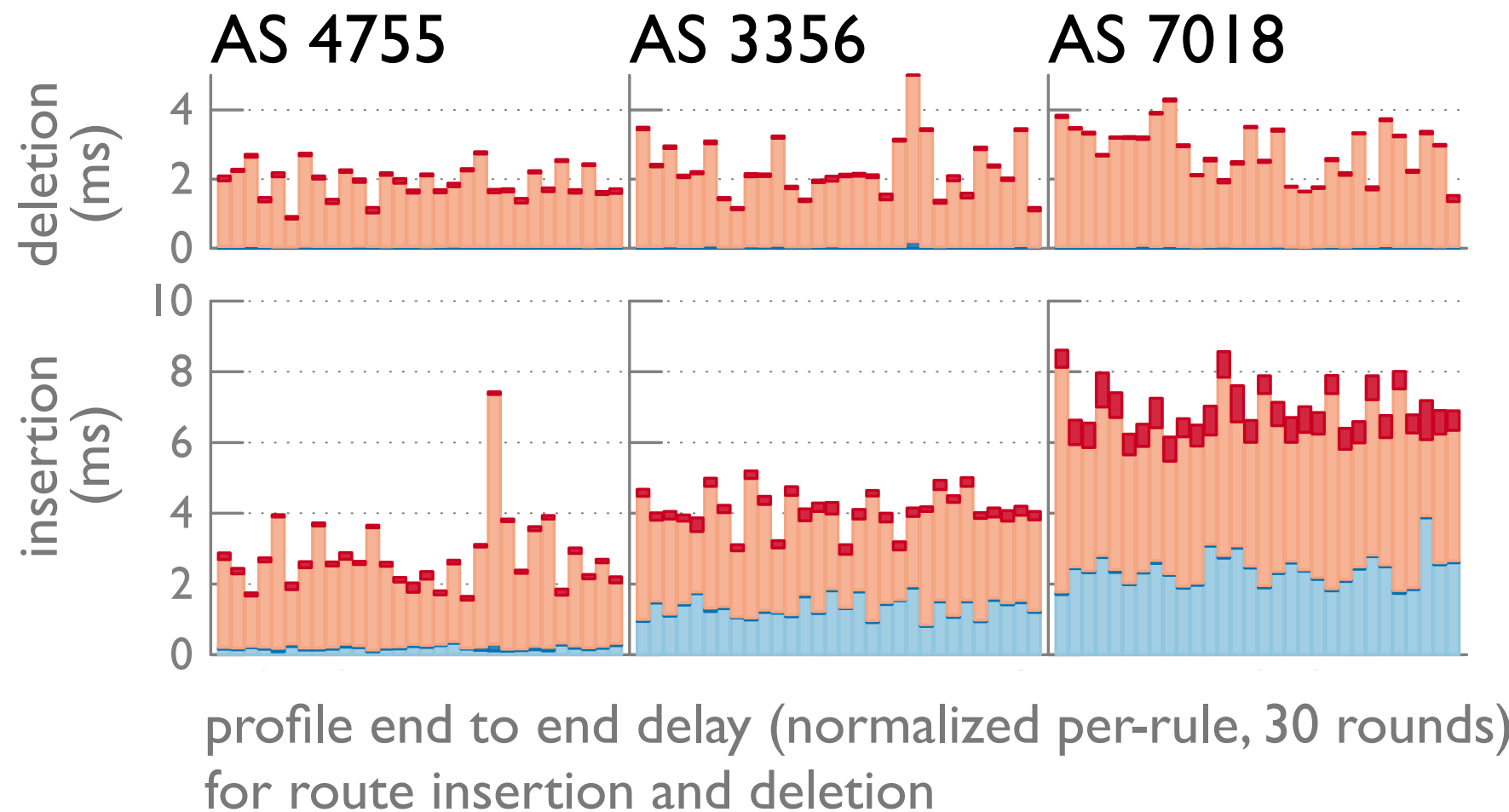
compute path 

lookup ports 

write to table 

trigger/rule 

evaluation

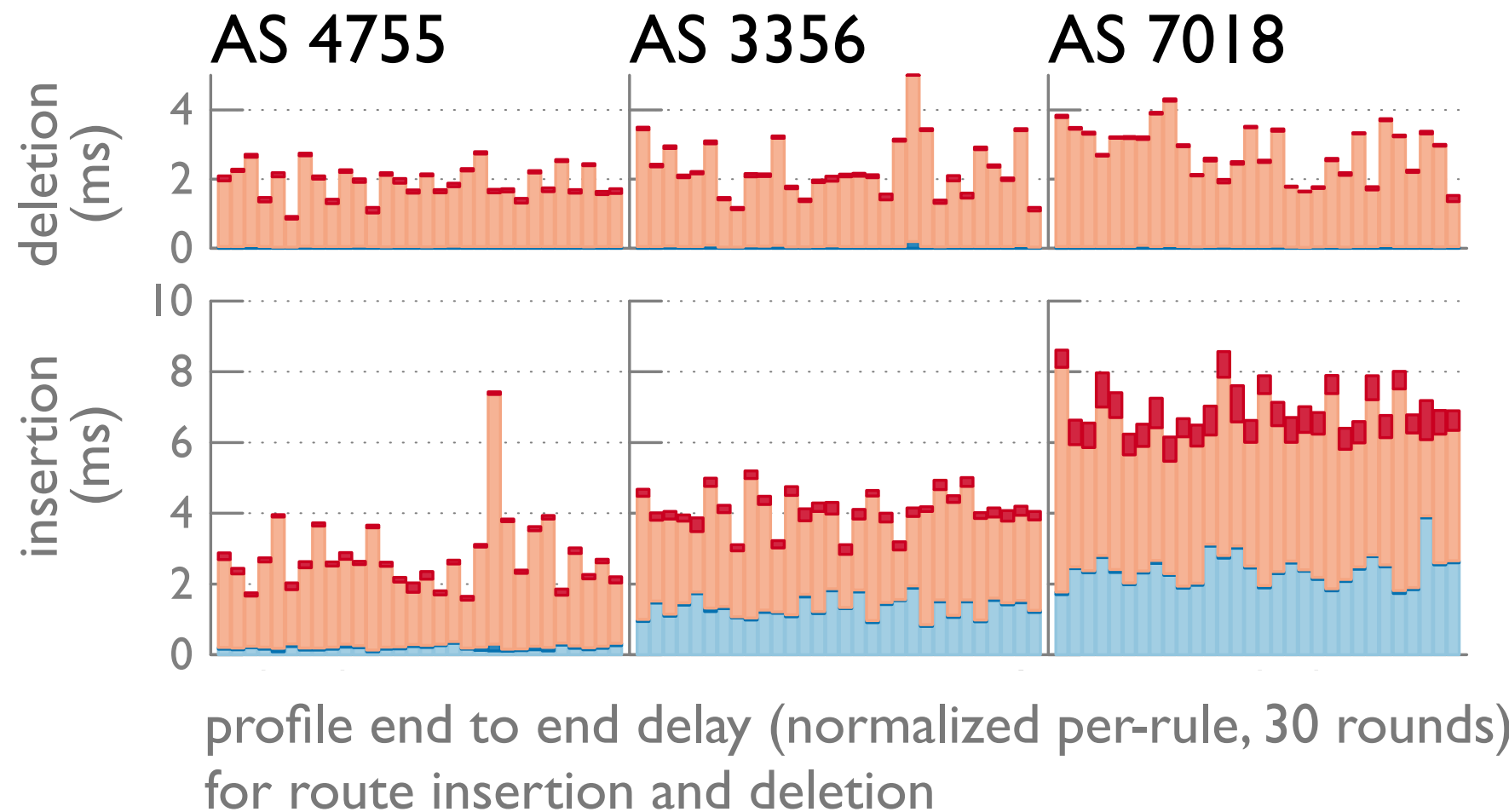


Rocketfuel ISP topology

AS#	nodes	links
4755	142	258
3356	1772	13640
7018	25382	11292

compute path
lookup ports
write to table
trigger/rule

evaluation



Rocketfuel ISP topology

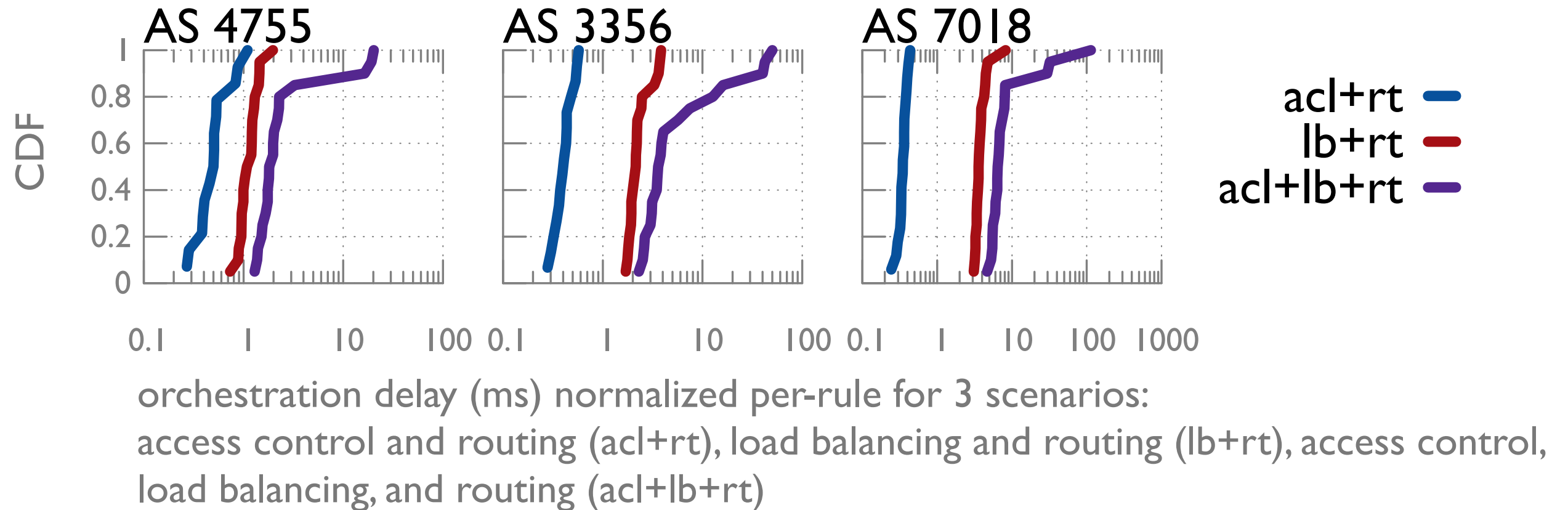
AS#	nodes	links
4755	142	258
3356	1772	13640
7018	25382	11292

compute path
lookup ports
write to table
trigger/rule

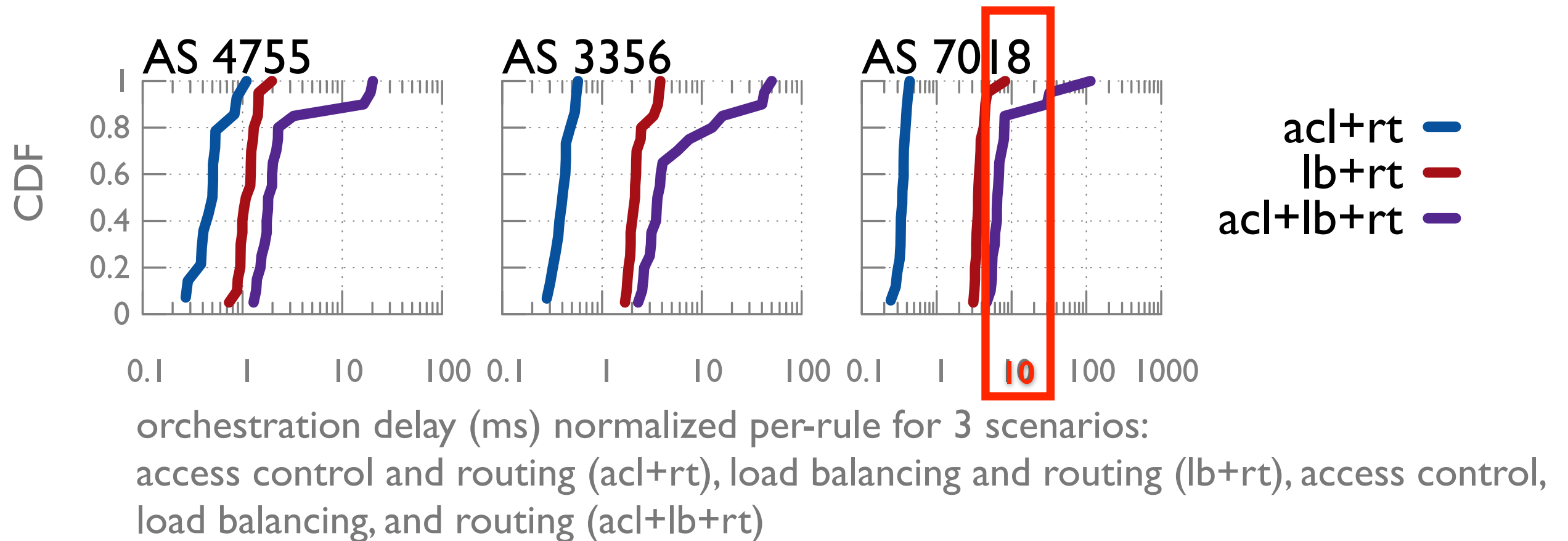
similar profile on fat-tree topology (fewer nodes, more links)

- total delay < 30ms for fat-tree with 5120 switches and 196608 links

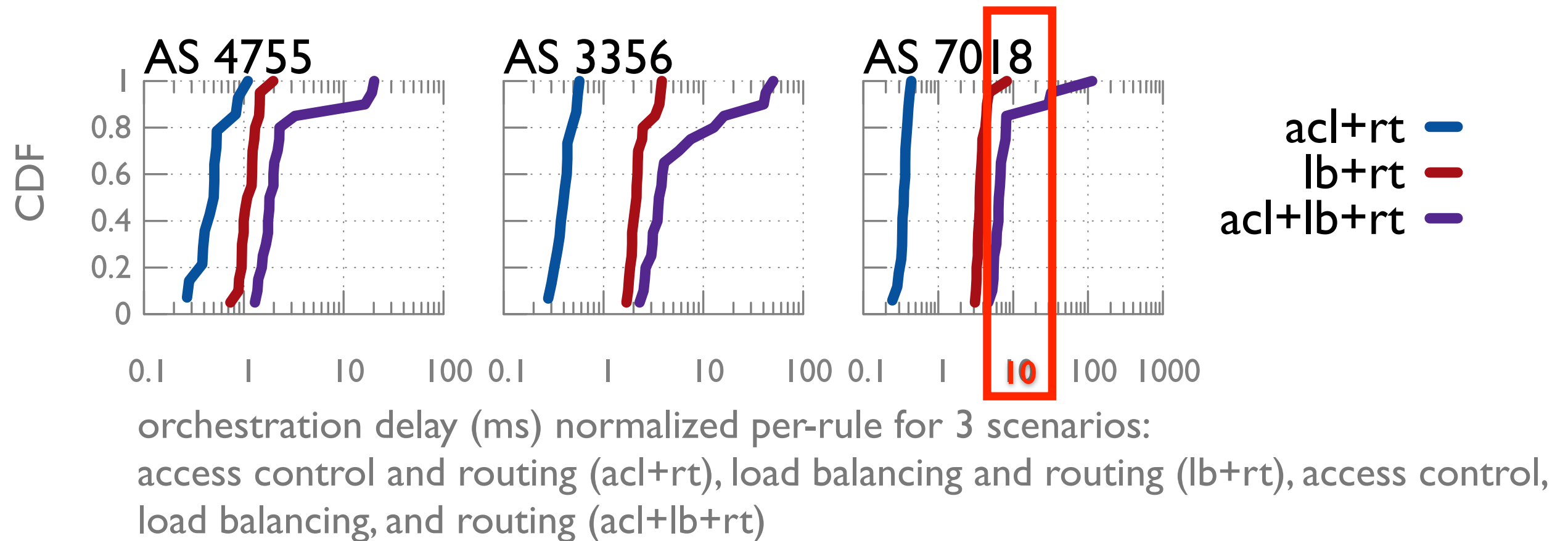
evaluation



evaluation



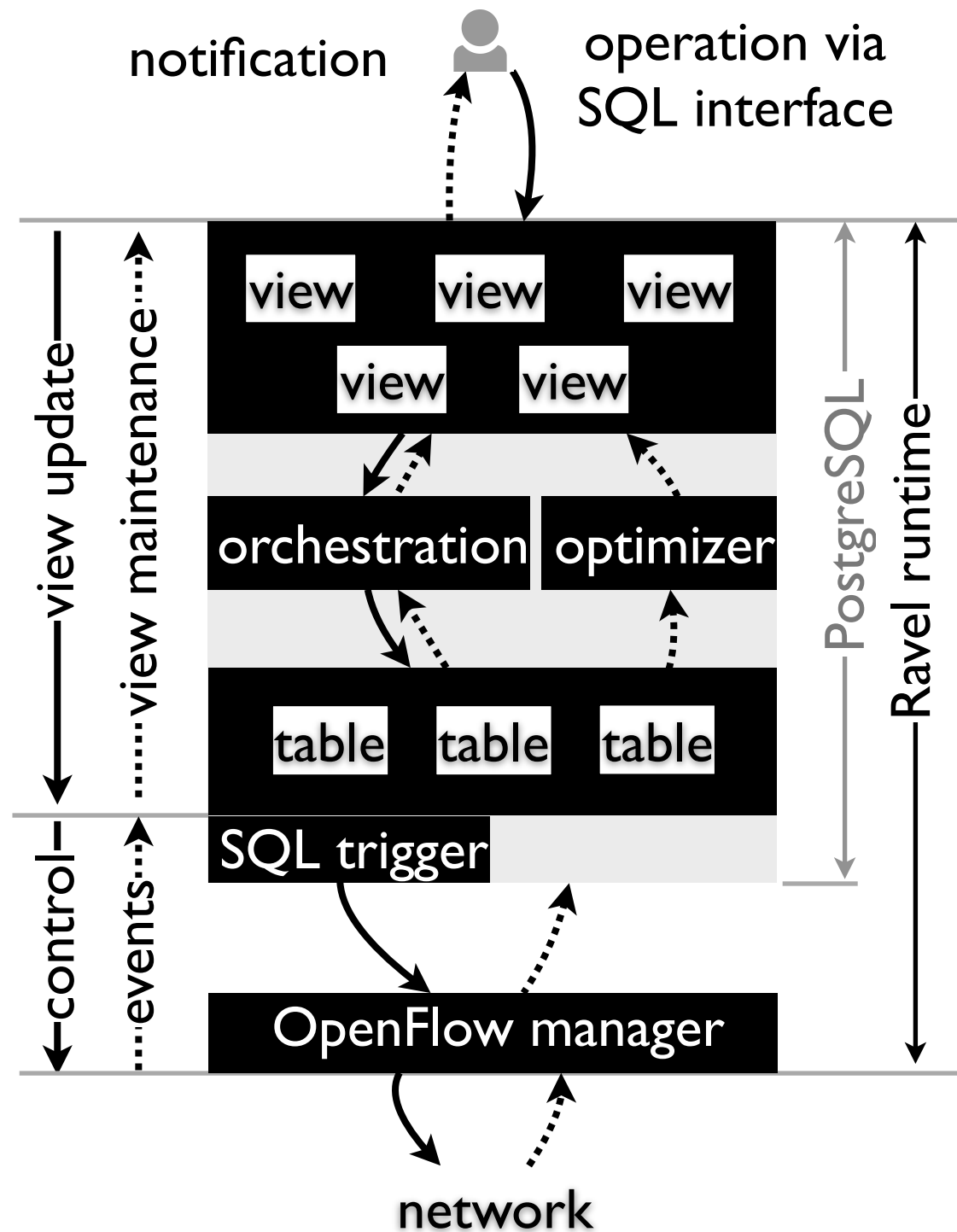
evaluation



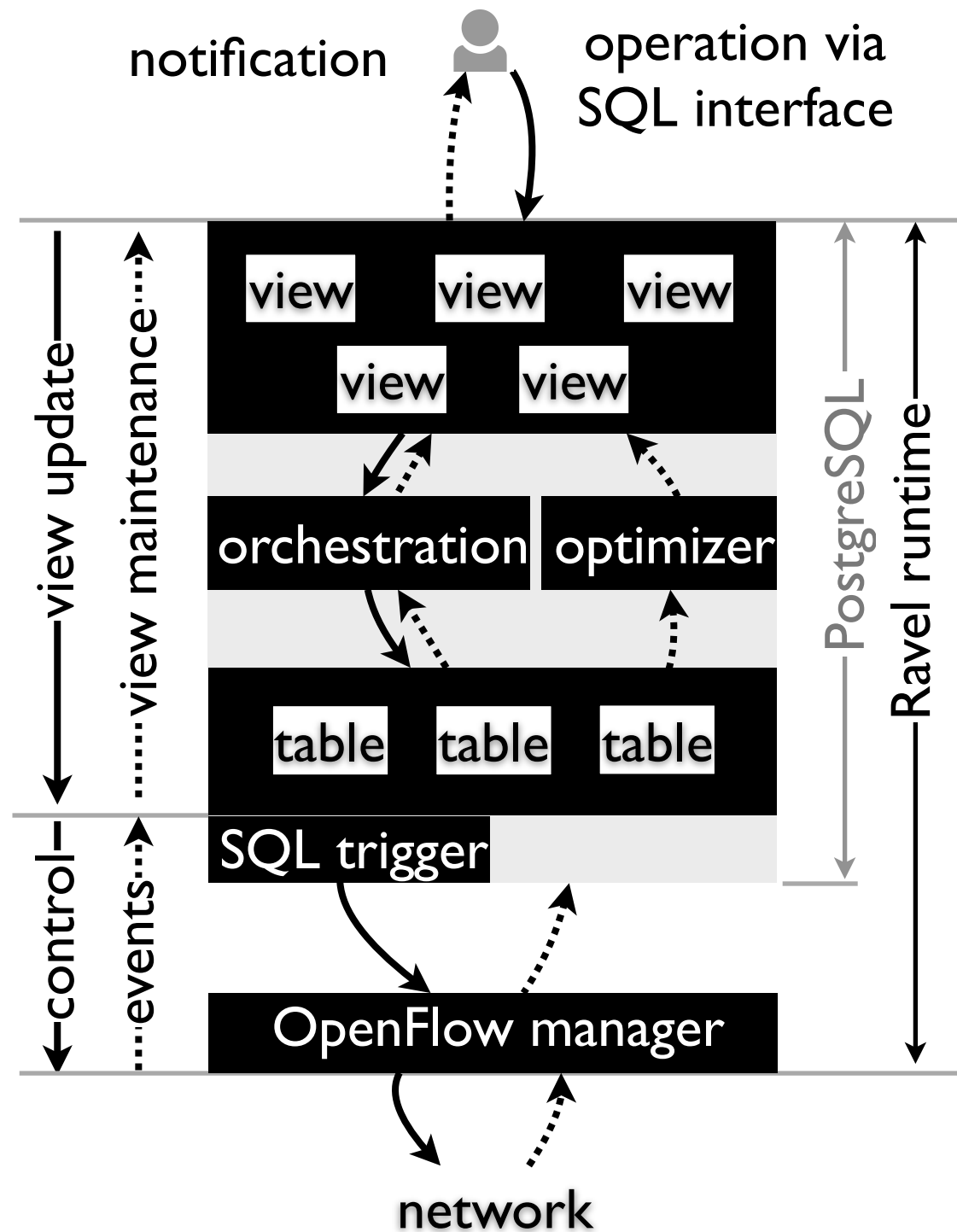
orchestration also scales gracefully on fat-tree

- < 30ms for fat-tree with 5120 switches and 196608 links

conclusion



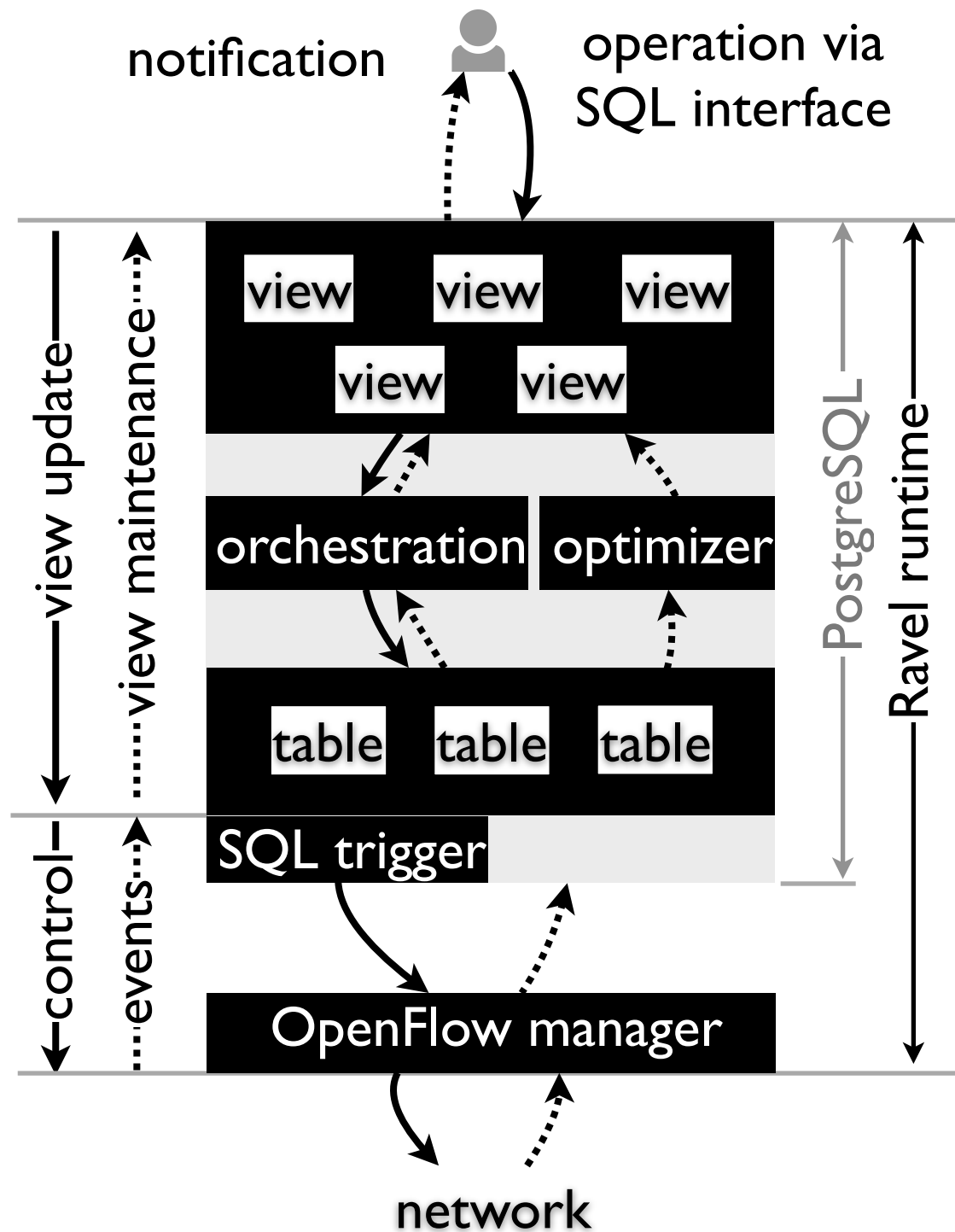
conclusion



this talk

- orchestratable abstraction via SQL

conclusion



this talk

- orchestratable abstraction via SQL

looking forward

- application of database features
 - network-wide transaction
 - bootstrapping legacy networks
- enhancing database
 - better runtime: orchestration
 - better control decision: view analysis
- interpretability
 - integrate foreign applications, plug-n-play 3rd party solvers

demo



demo





playtime

download *Ravel*

ravel-net.org/download

start playing: tutorials, add your own app

ravel-net.org

explore more

github.com/ravel-net