Verifying Parameterized Networks
Specified by
Vertex-Replacement Graph Grammars

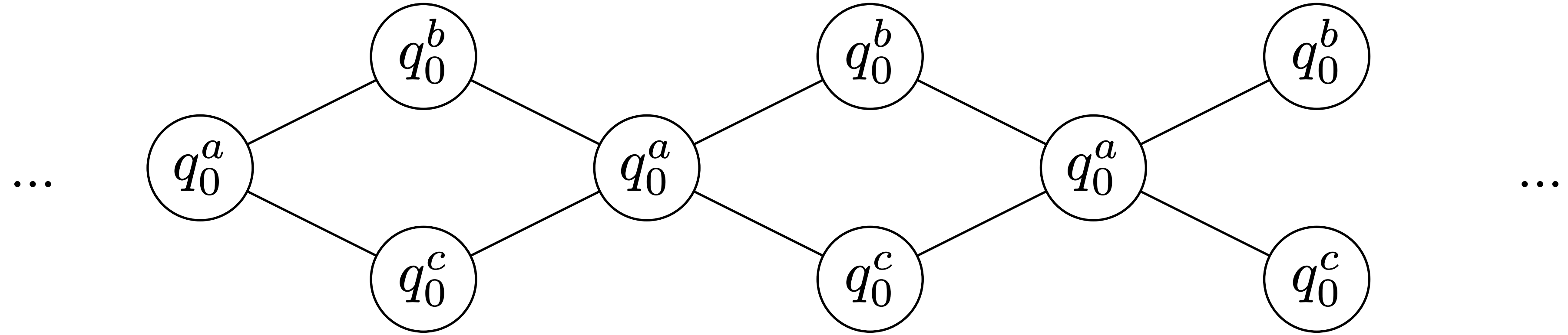
Neven Villani, Radu Iosif, Arnaud Sangnier

Univ. Grenoble Alpes, Verimag

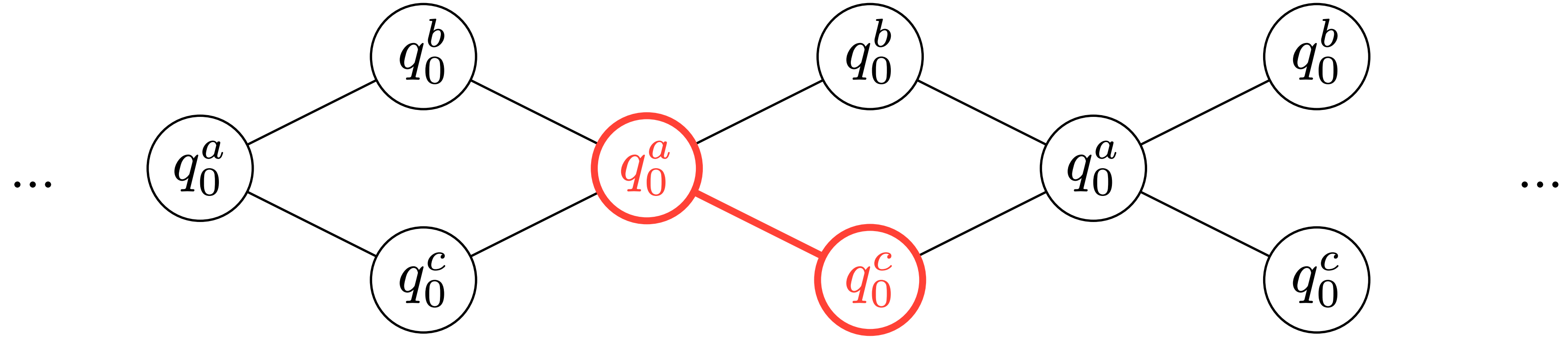
2025-05-13

1. Context

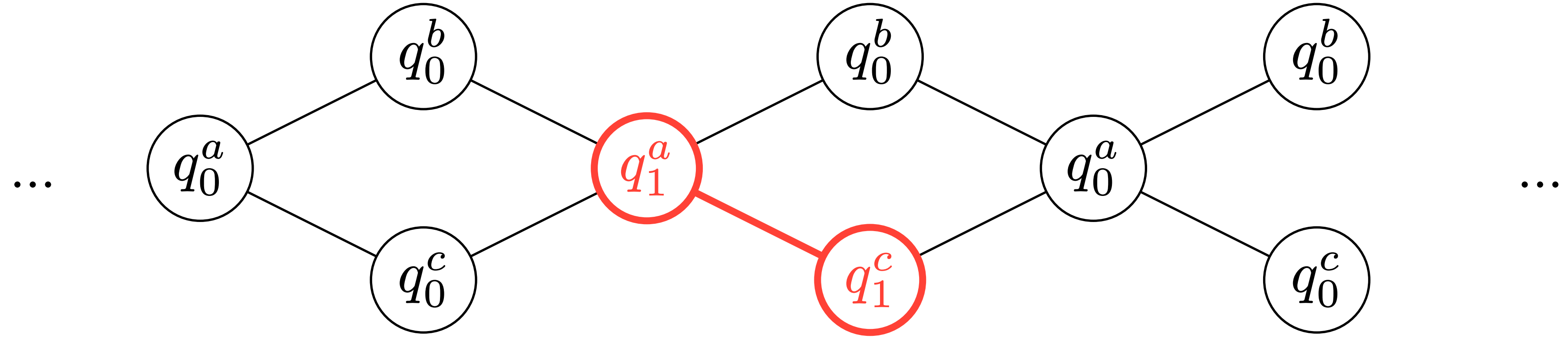
- Parameterized structured networks
- Binary rendezvous
- Safety properties



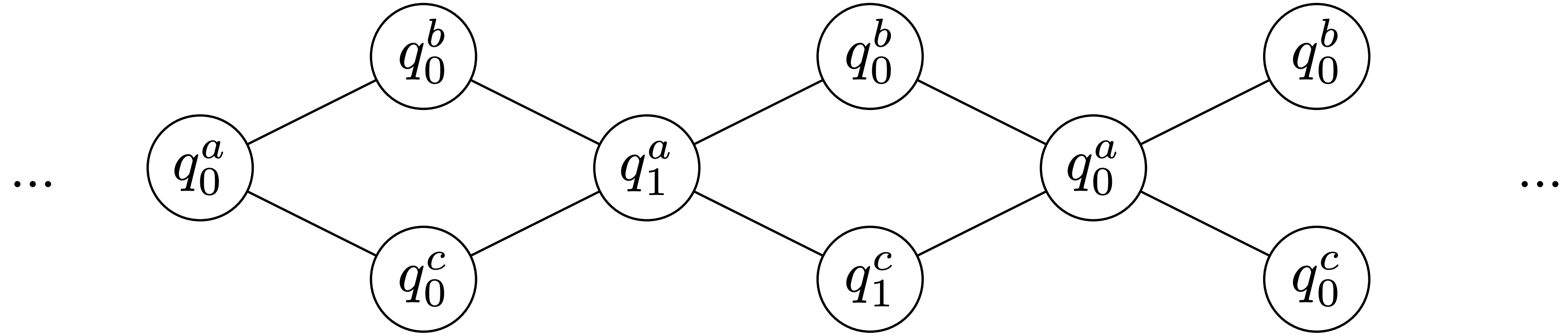
- Parameterized structured networks
- Binary rendezvous
- Safety properties



- Parameterized structured networks
- Binary rendezvous
- Safety properties



- Parameterized structured networks
- Binary rendezvous
- Safety properties



$\#q_1^b > 1 \wedge \#q_1^c > 1$ reachable ?

processes

architecture

$$(P, \Gamma) \vDash ?\, \varphi$$

safety property

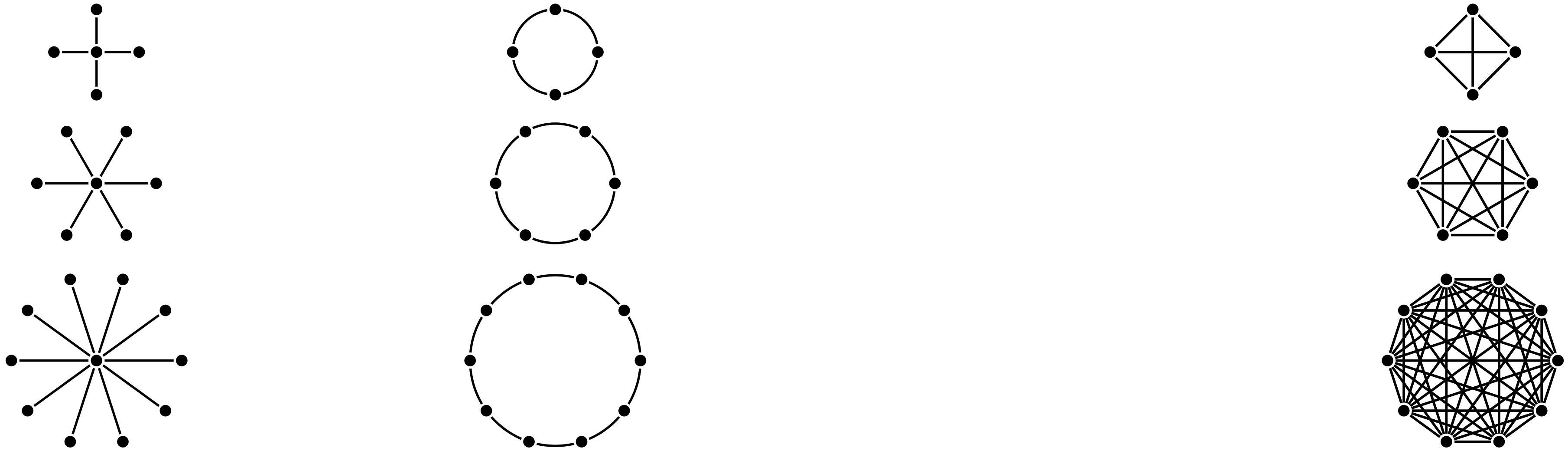
processes

architecture

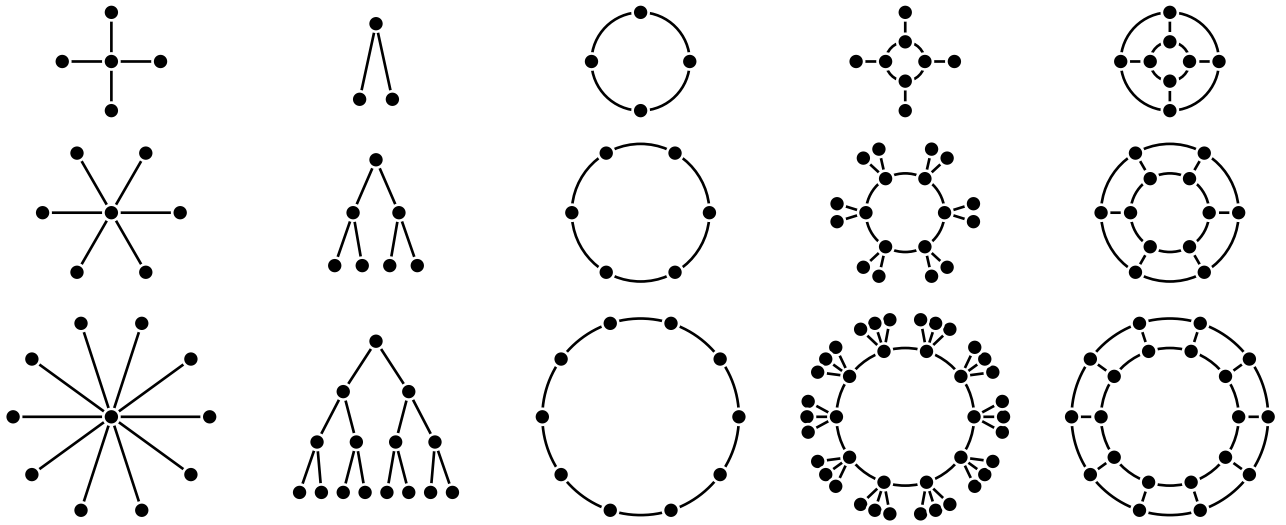
$$(P, \Gamma) \vDash ? \varphi$$

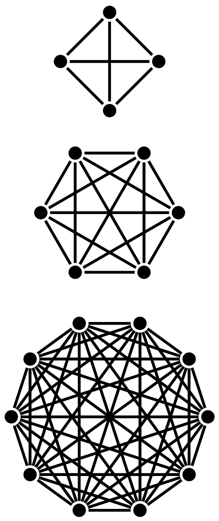
safety property

What formalism to describe architectures ?



What formalism to describe architectures ?

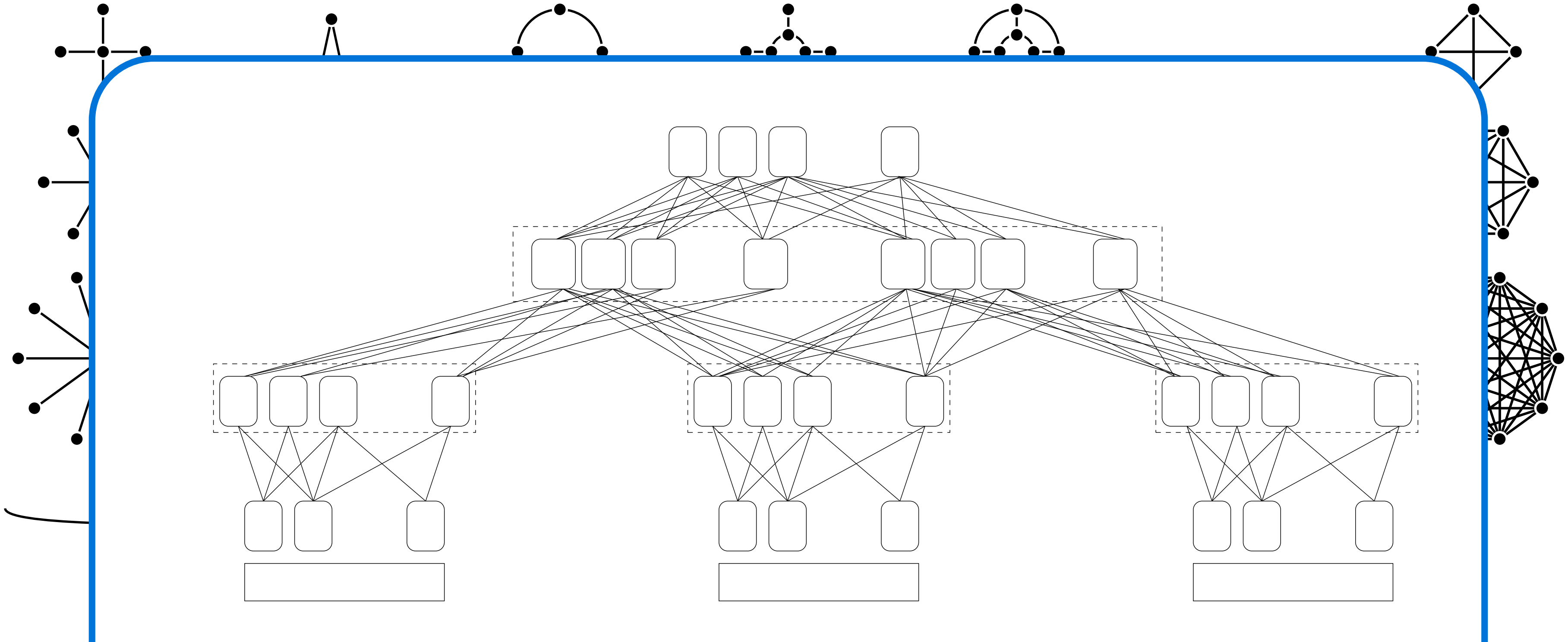




Hyperedge **R**eplacement (sparse only)

e.g., us, CAV 2025

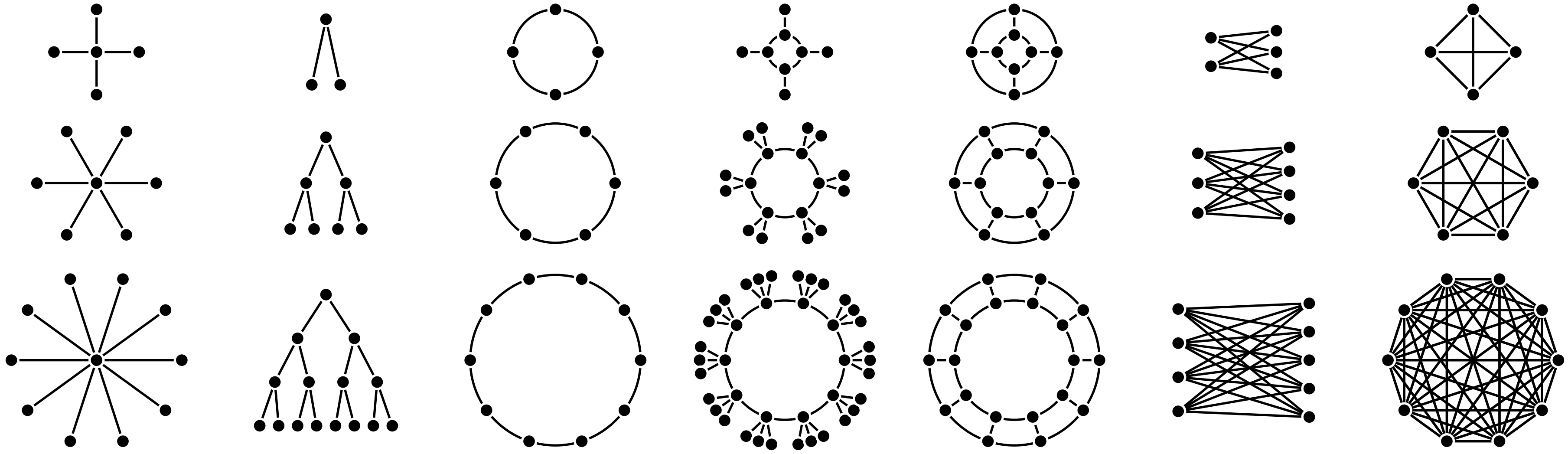
What formalism to describe architectures ?



Azure Datacenter Topology

Greenberg et al., SIGCOMM 2009

What formalism to describe architectures ?



Hyperedge **R**eplacement (sparse only)

e.g., us, CAV 2025

Vertex **R**eplacement (incl. some dense)

Contribution

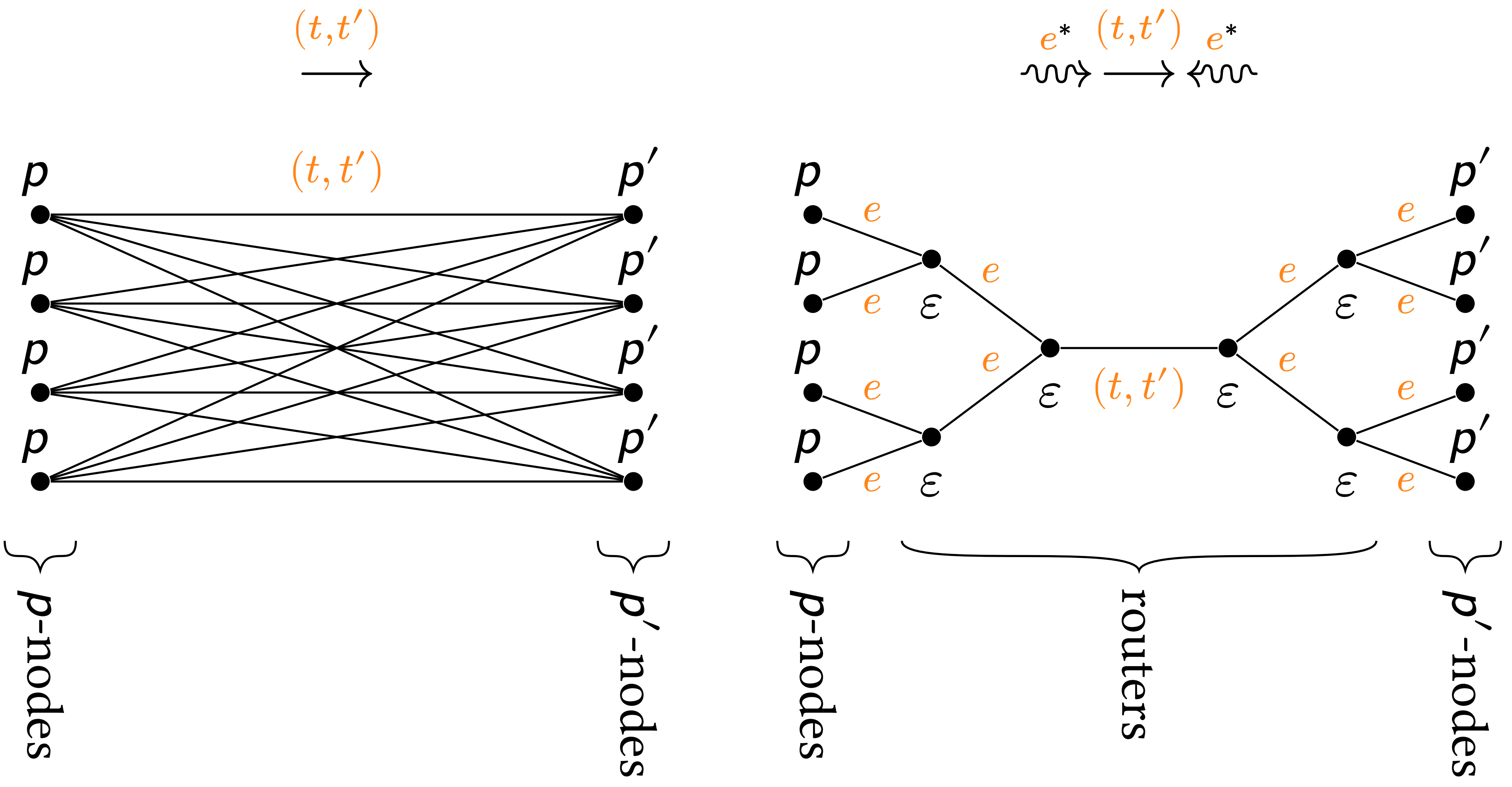
A **translation** from VR to HR architectures.

Contribution

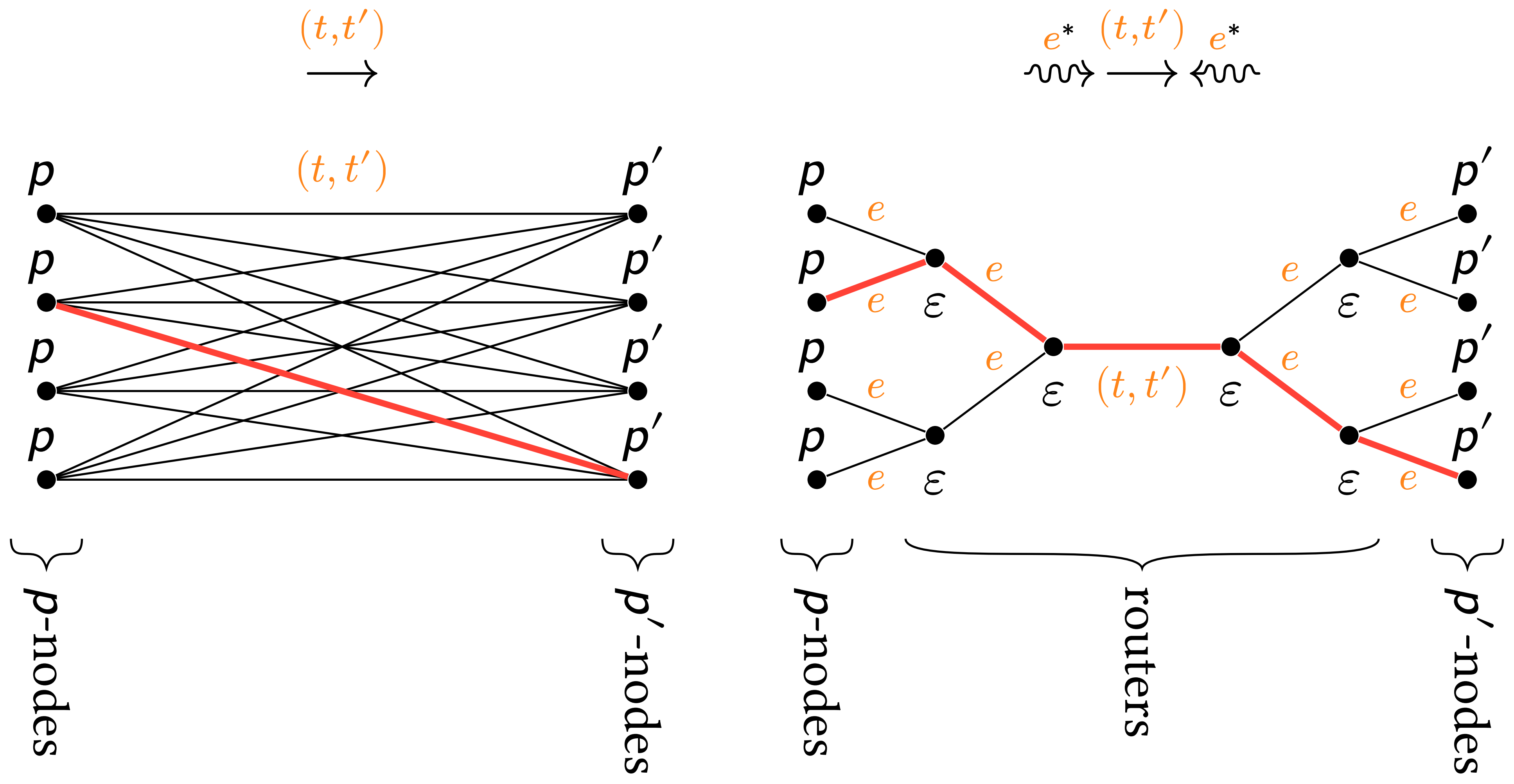
Adapting a translation from VR to HR architectures **to the case of systems**, and studying which safety properties are preserved.

B. Courcelle, Structural Properties of Context-Free Sets of Graphs Generated by Vertex Replacement, *Information and Computation*, 1995

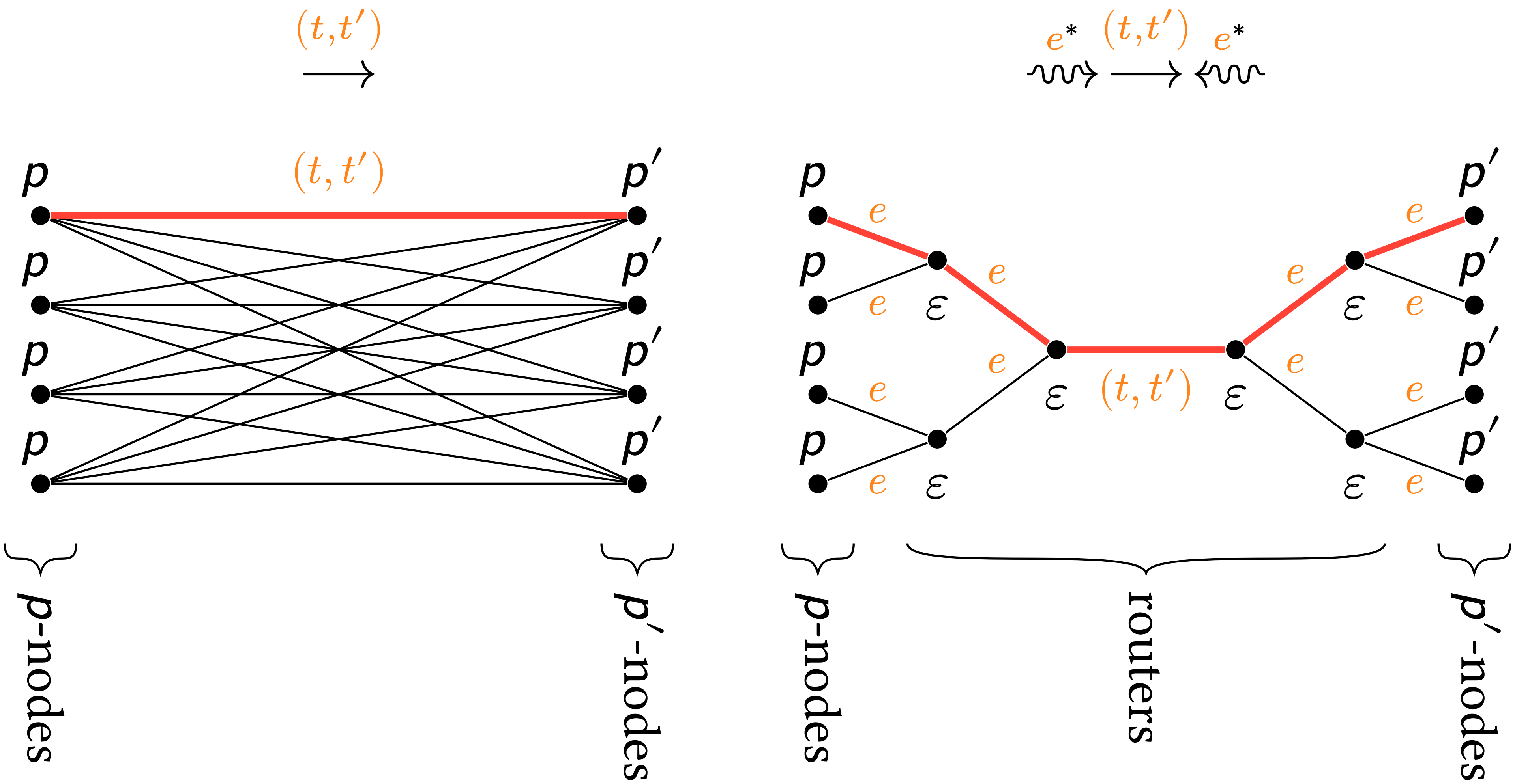
Key idea



Key idea

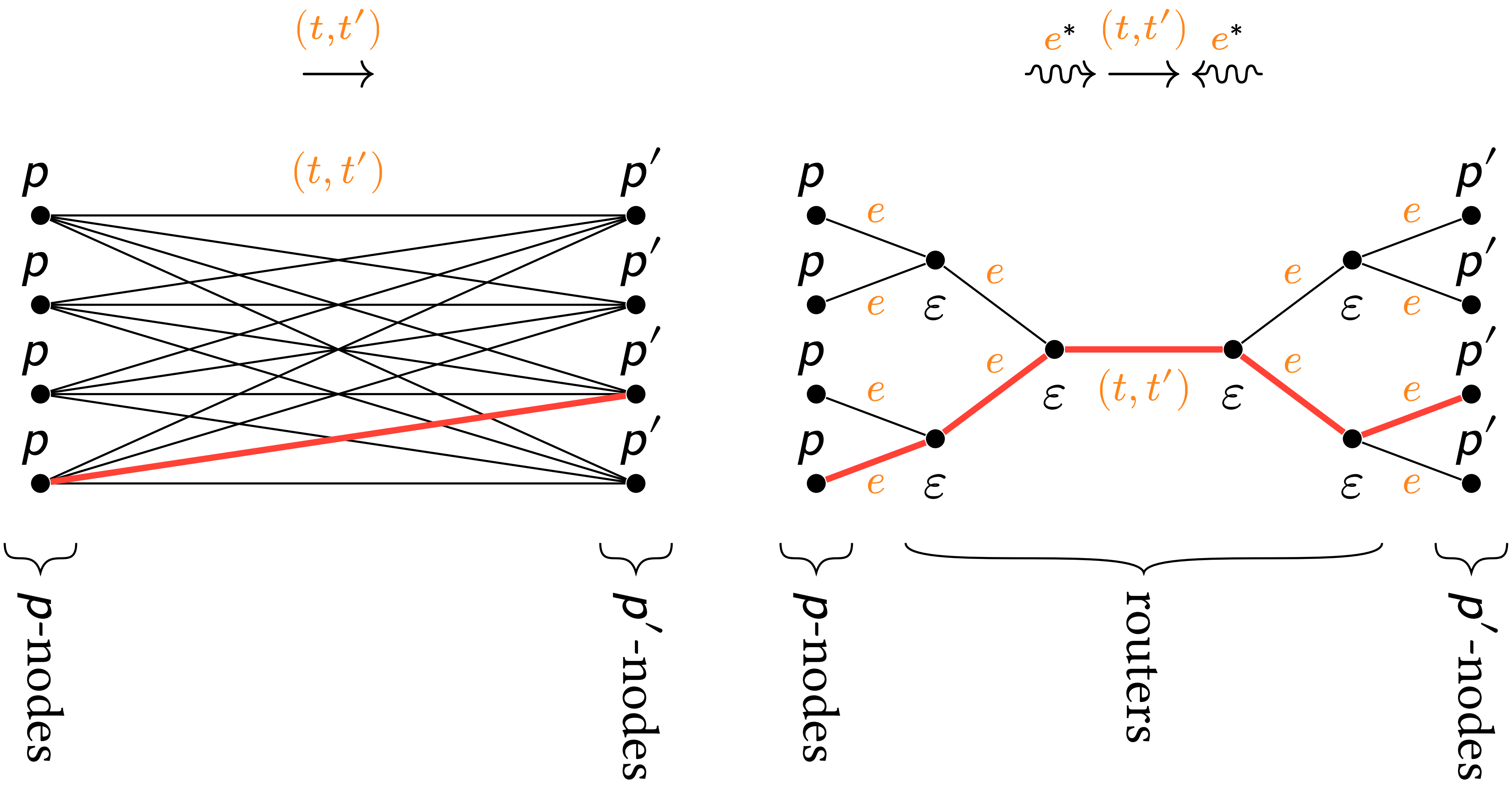


Key idea



$\xrightarrow{(t,t')}$
$\overset{e^*}{\rightsquigarrow} \xrightarrow{(t,t')} \overset{e^*}{\leftsquigarrow}$
p
p'
(t,t')
e
ε
p-nodes
p'-nodes
routers

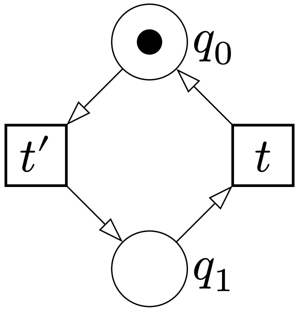
Key idea

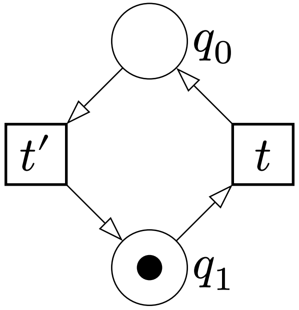


2. Encoding of Networks

Processes and Systems

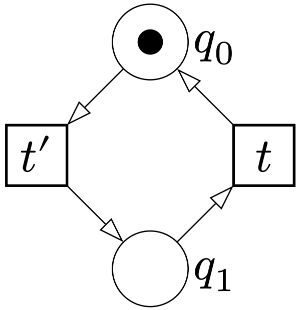
- process types p_1, p_2, … = Petri nets (PN) with observable transitions

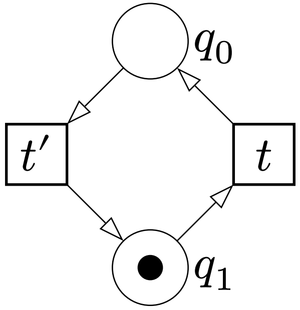
$p_1 =$ 

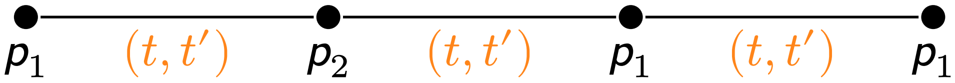
$p_2 =$ 

Processes and Systems

- process types p_1, p_2, … = Petri nets (PN) with observable transitions
- system: graph with
  - vertices labeled by a process type
  - edges labeled by pairs of observable transitions

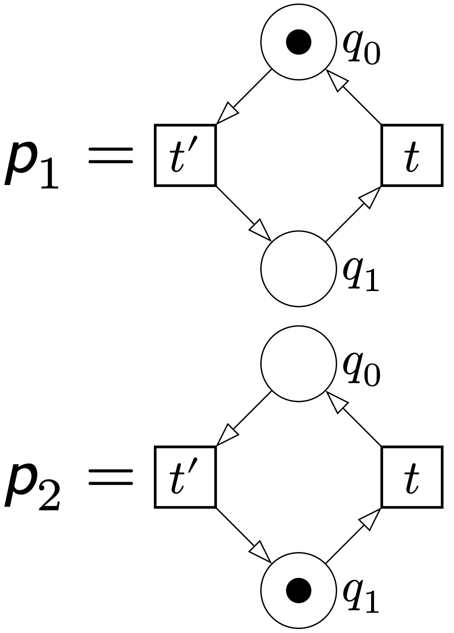
$p_1 =$ 

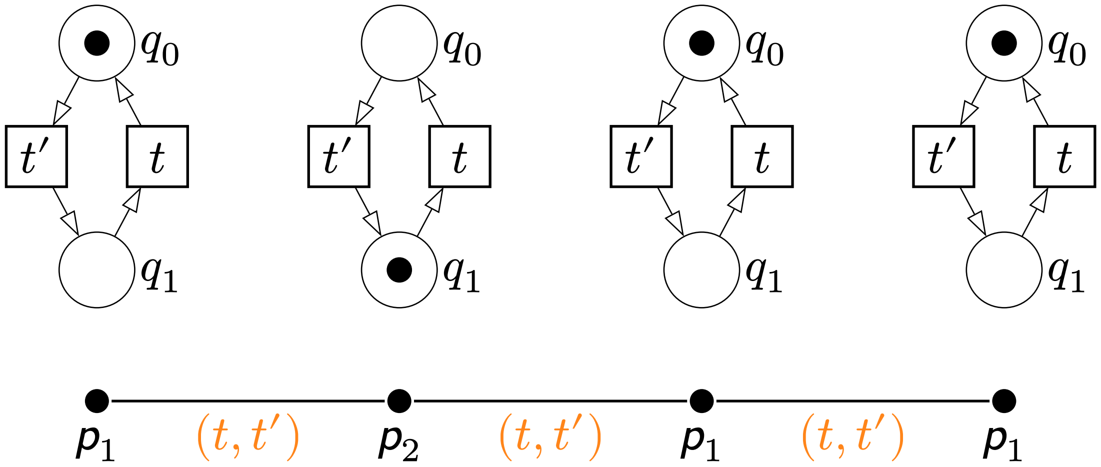
$p_2 =$ 



Processes and Systems

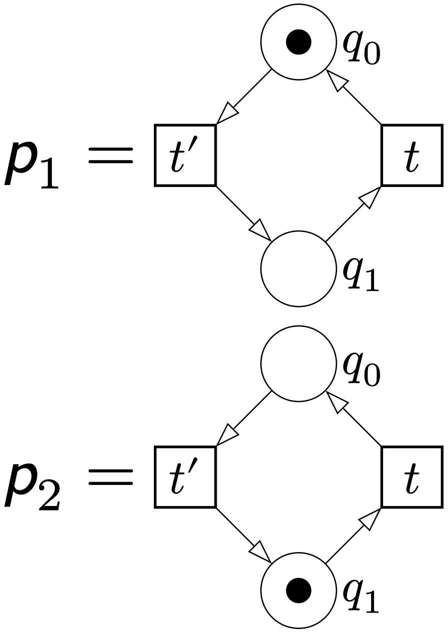
- process types p_1, p_2, … = Petri nets (PN) with observable transitions
- system: graph with
  - vertices labeled by a process type
  - edges labeled by pairs of observable transitions

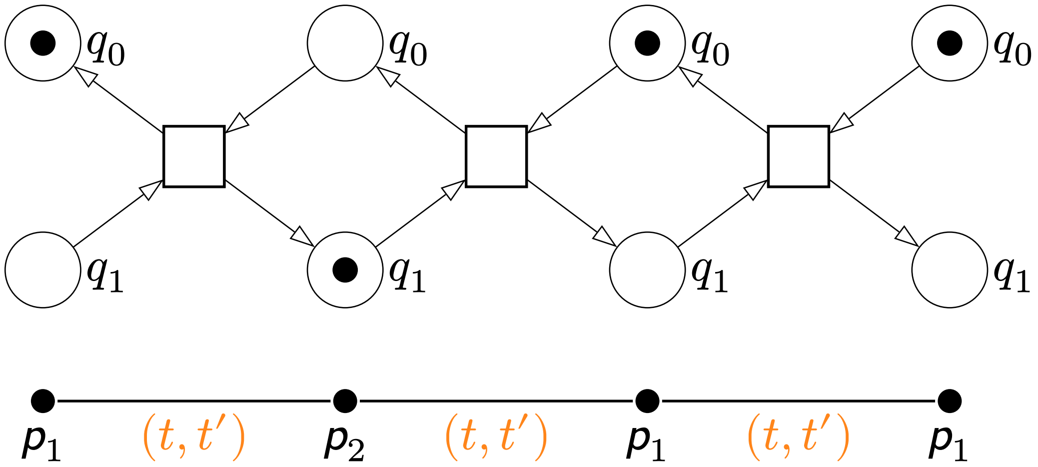




Processes and Systems

- process types p_1, p_2, … = Petri nets (PN) with observable transitions
- system: graph with
  - vertices labeled by a process type
  - edges labeled by pairs of observable transitions

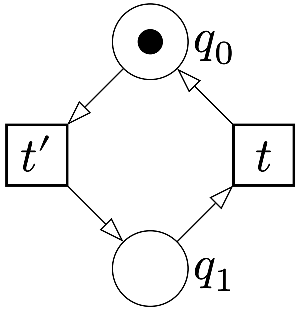




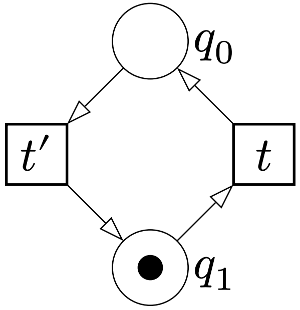
Processes and Systems

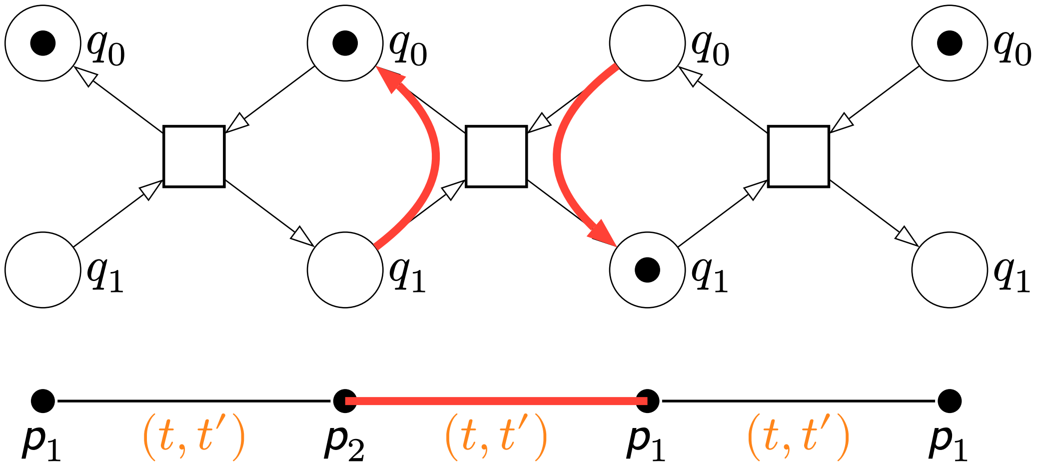
- process types p_1, p_2, … = Petri nets (PN) with observable transitions
- system: graph with
  - vertices labeled by a process type
  - edges labeled by pairs of observable transitions

$p_1 =$



$p_2 =$





3. HR & VR

Single vertex

$$\bullet\pi = \left(\overset{\pi}{\bullet}\right)$$

HR

Single vertex

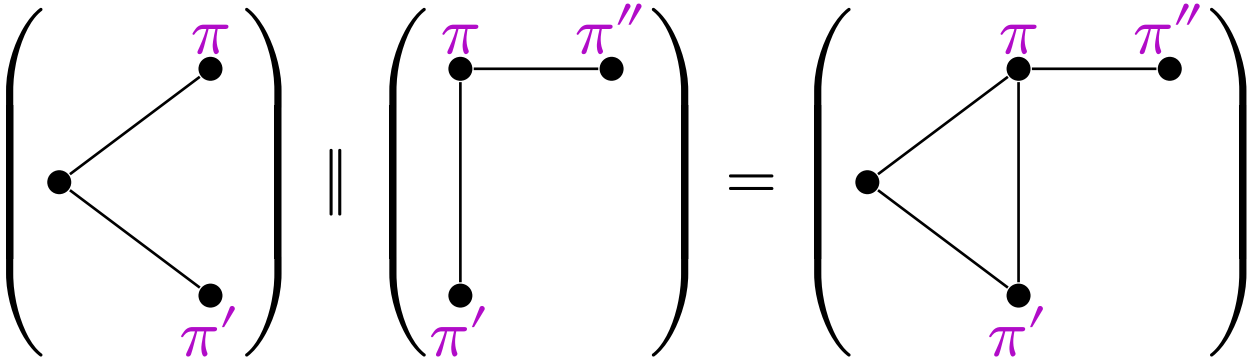
$$\bullet\pi = \left(\overset{\pi}{\bullet}\right)$$

Single edge

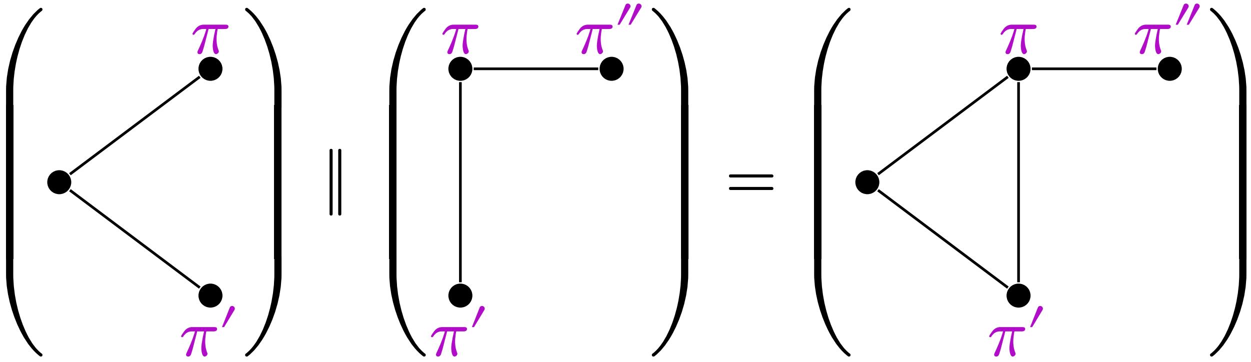
$$\vec{e}_{\pi,\pi'} = \left(\begin{array}{c} \overset{\pi}{\bullet} \\ \Bigg| \; e \\ \underset{\pi'}{\bullet} \end{array}\right)$$

HR

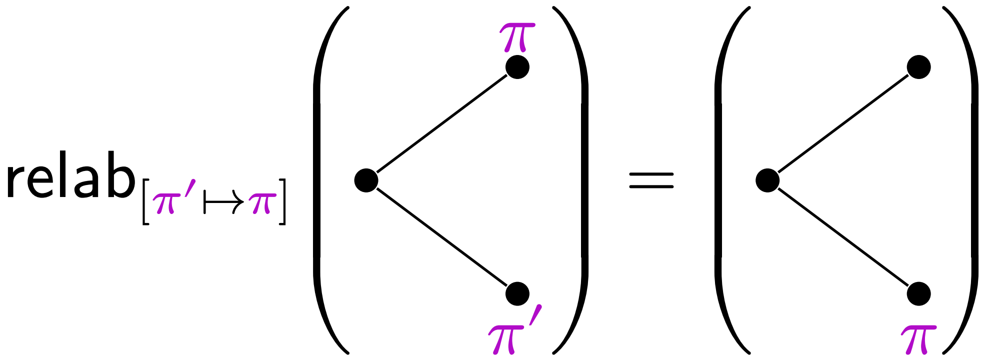
Parallel composition



Parallel composition



Relabeling

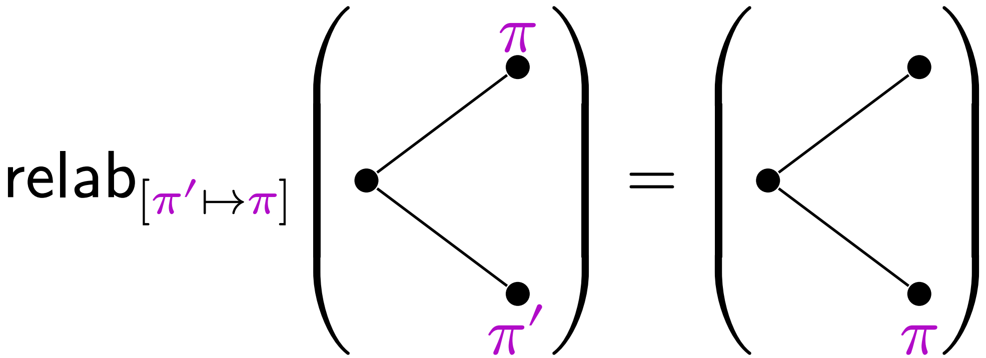


VR

Single vertex

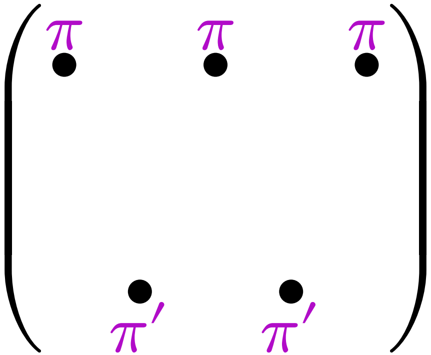
$$\bullet\pi = \left(\overset{\pi}{\bullet}\right)$$

Relabeling

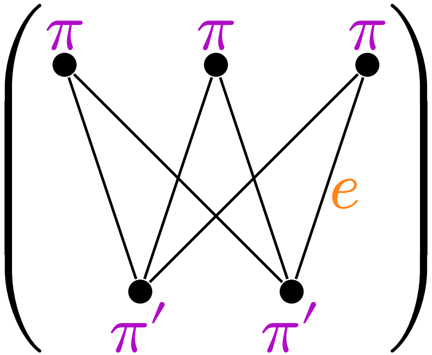


All-pairs edges

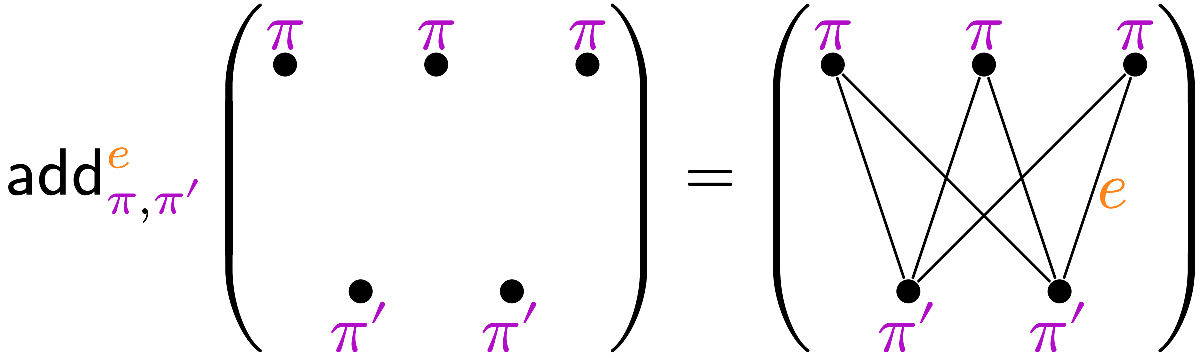
$\mathsf{add}^{e}_{\pi,\pi'}$



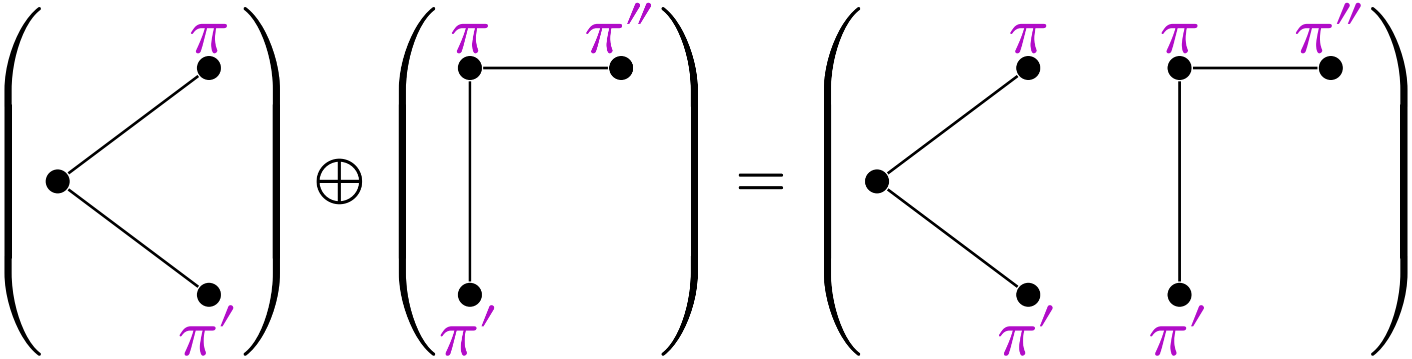
$=$



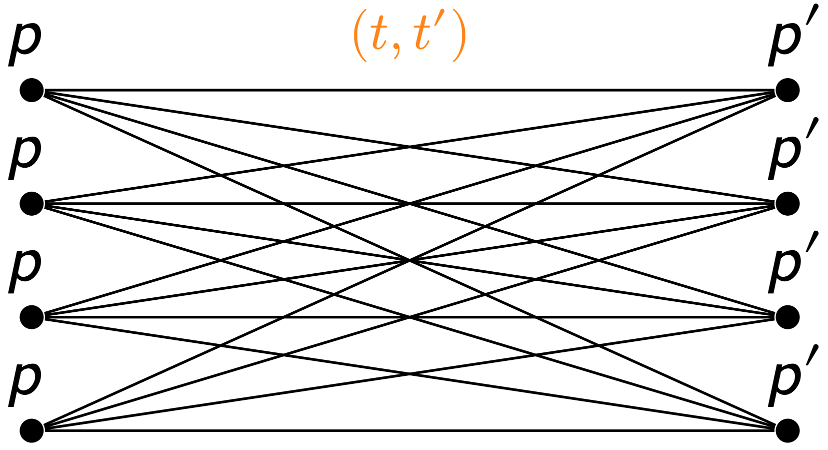
All-pairs edges



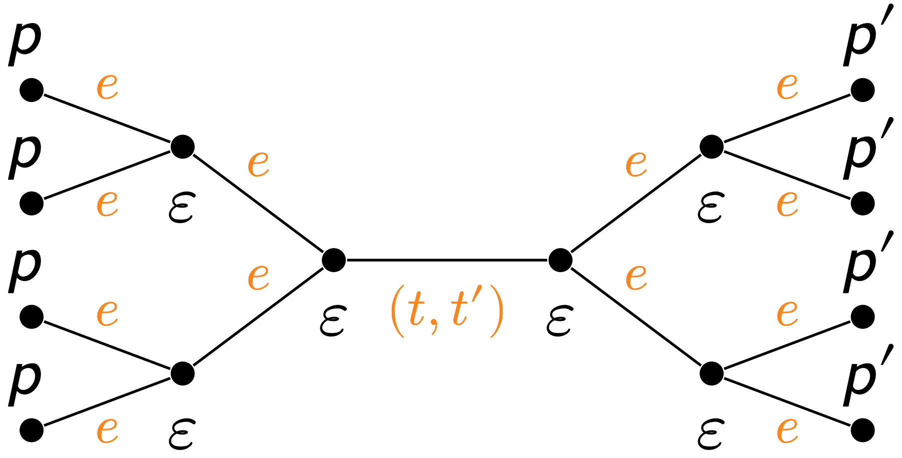
Disjoint union



VR

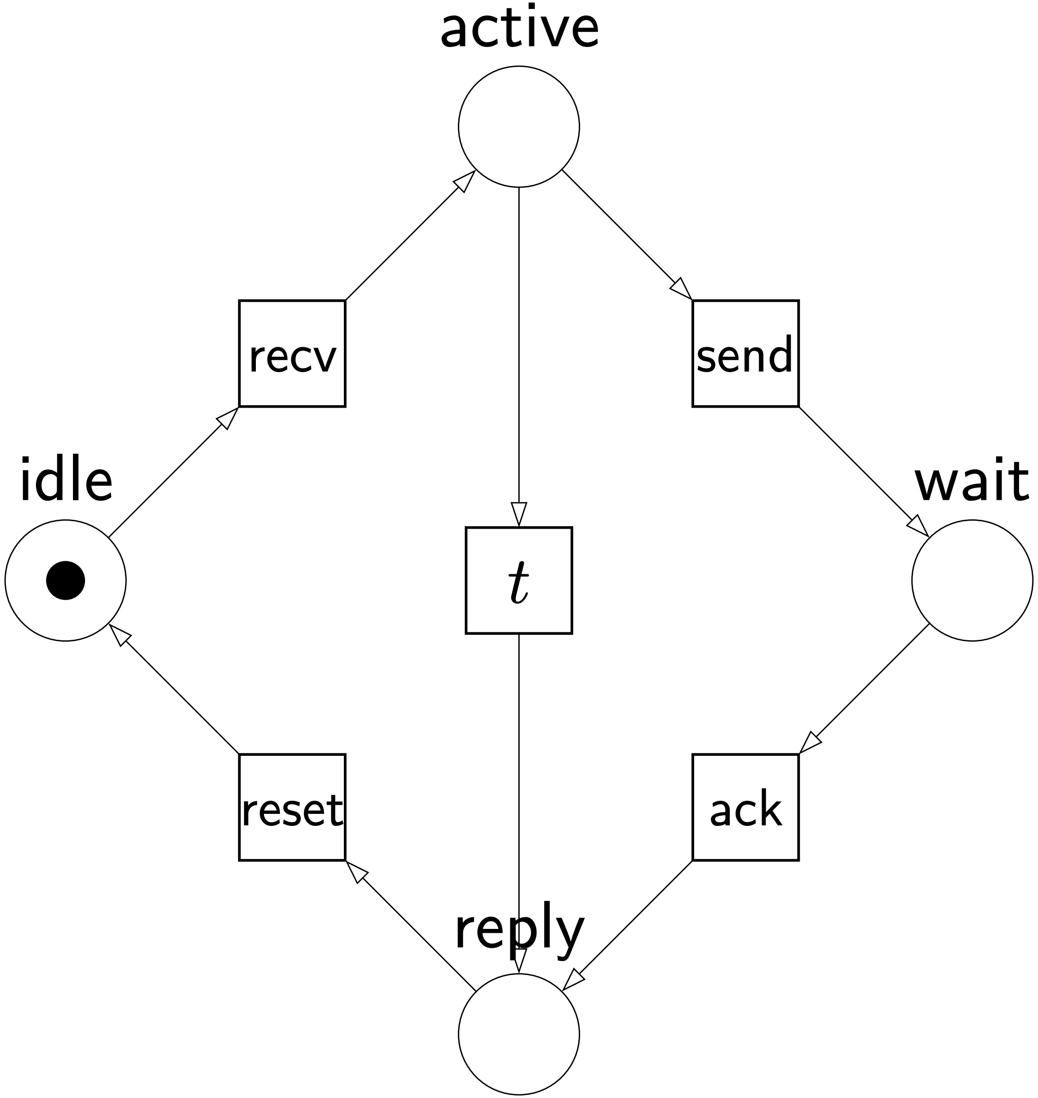


HR

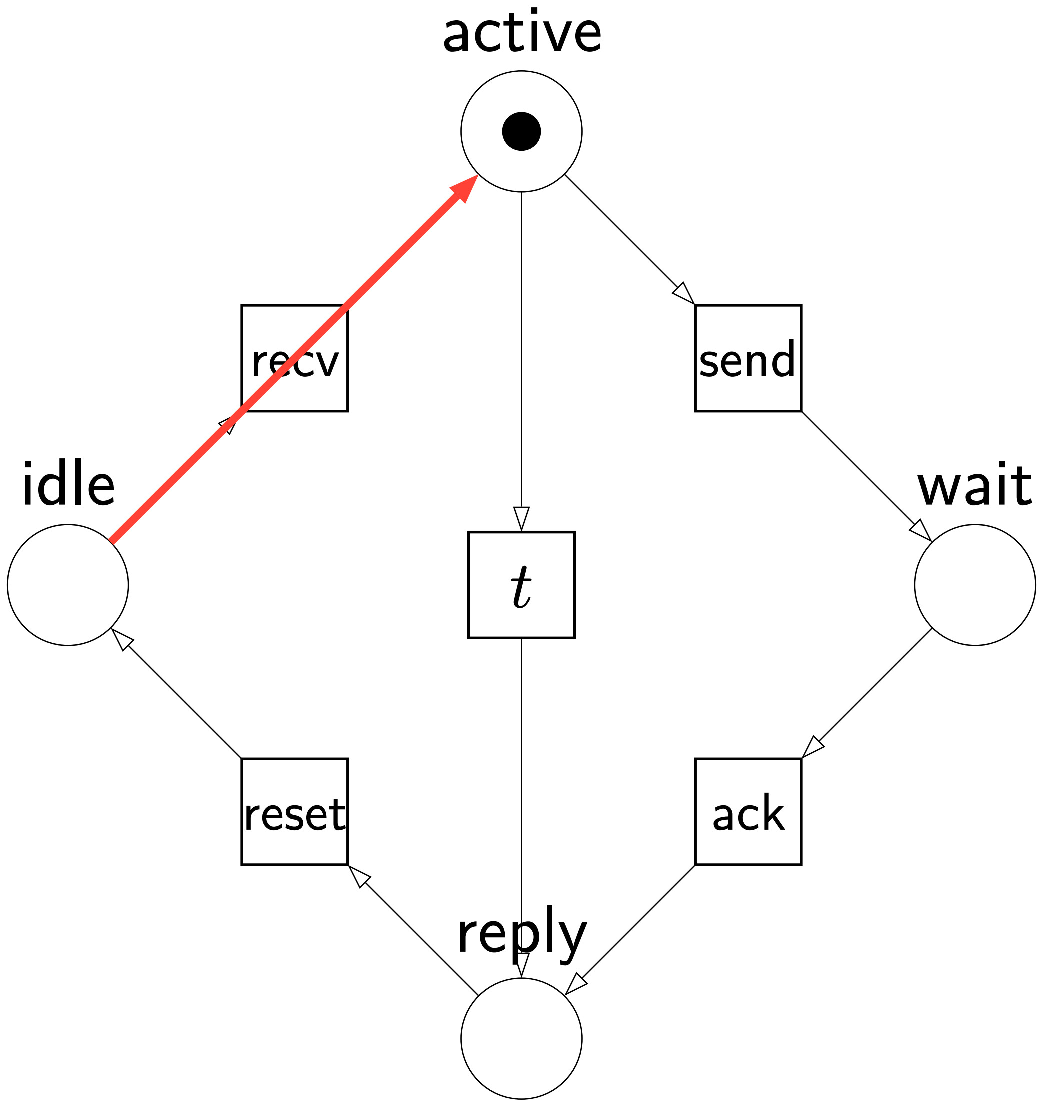


4. Routing

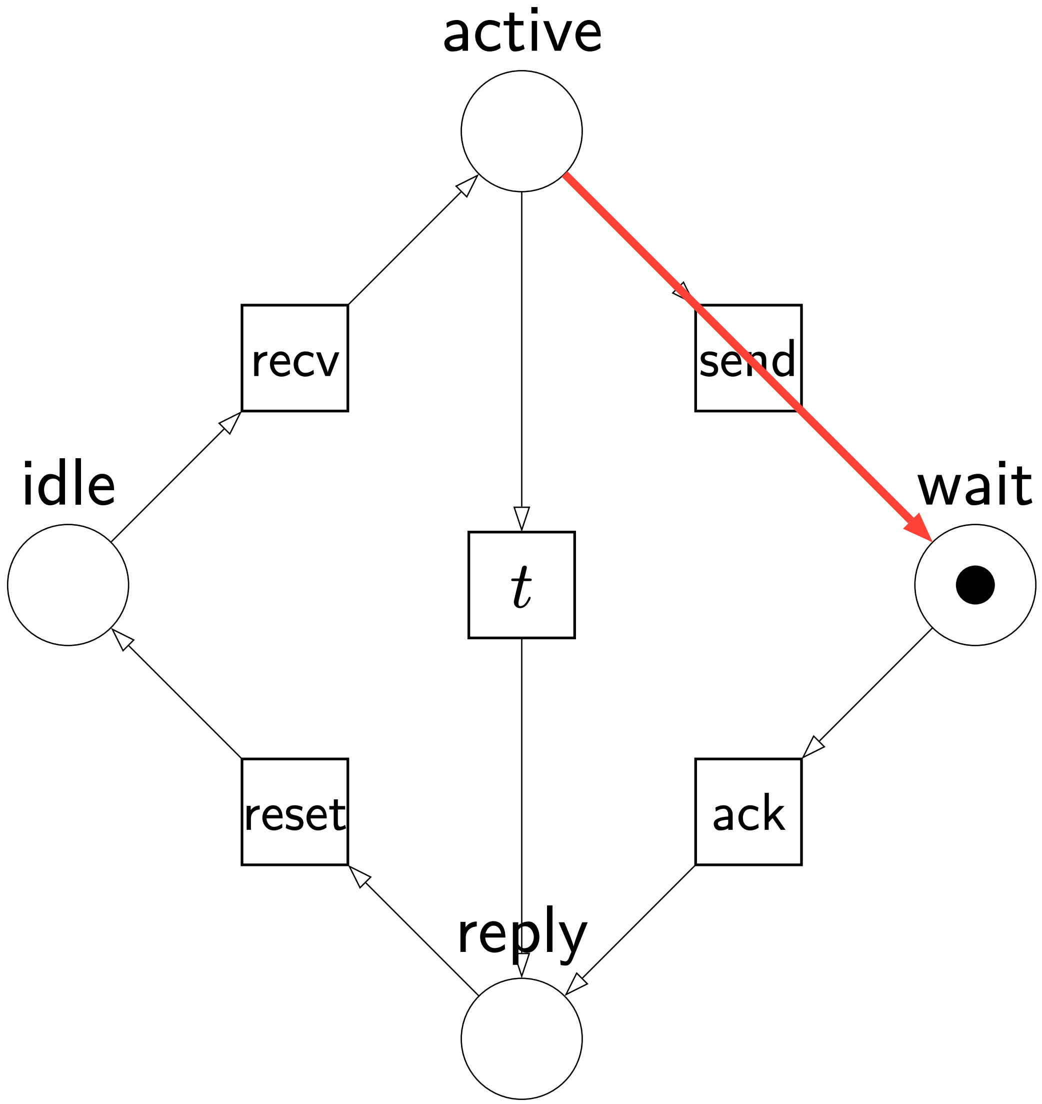
Construction of a Router



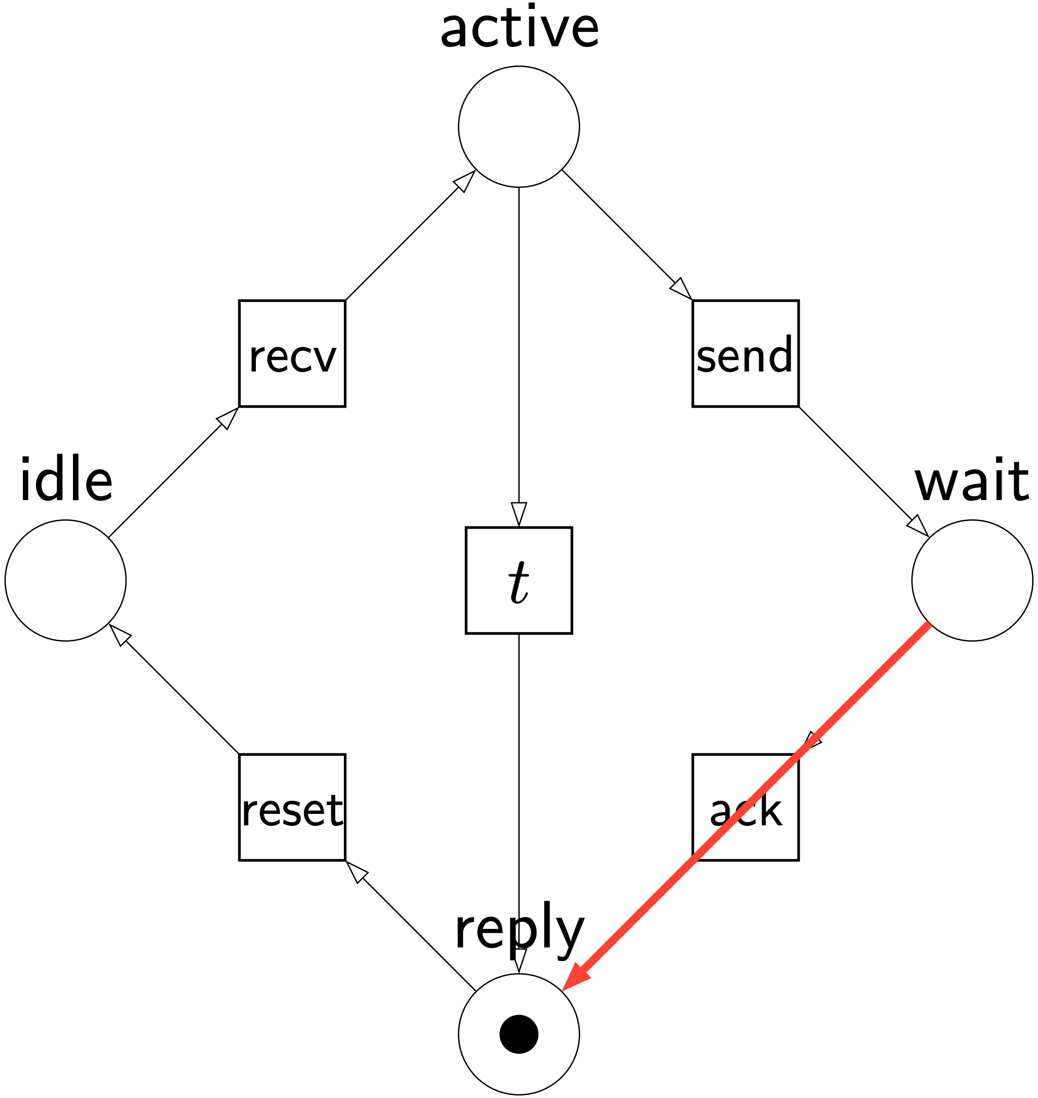
Construction of a Router



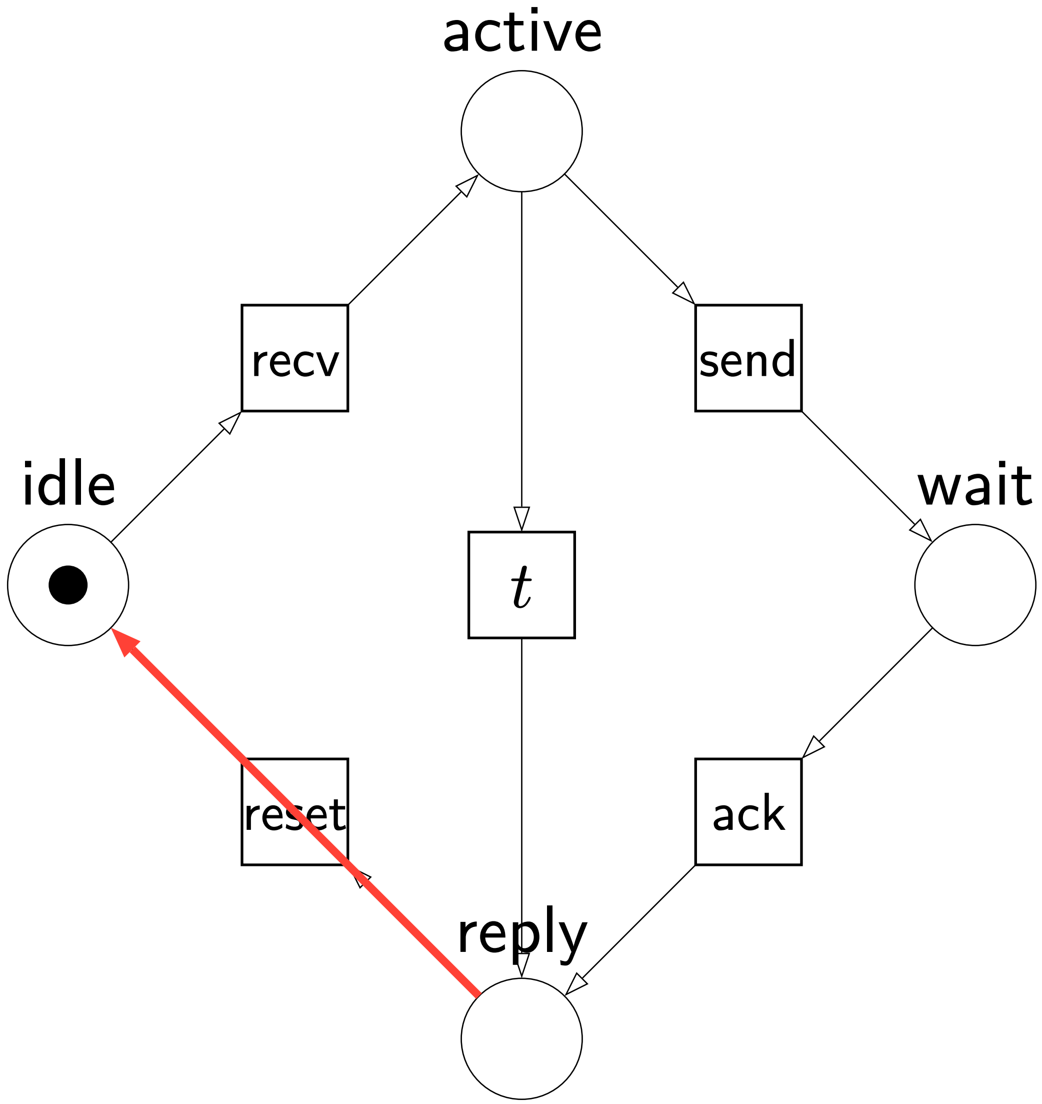
Construction of a Router



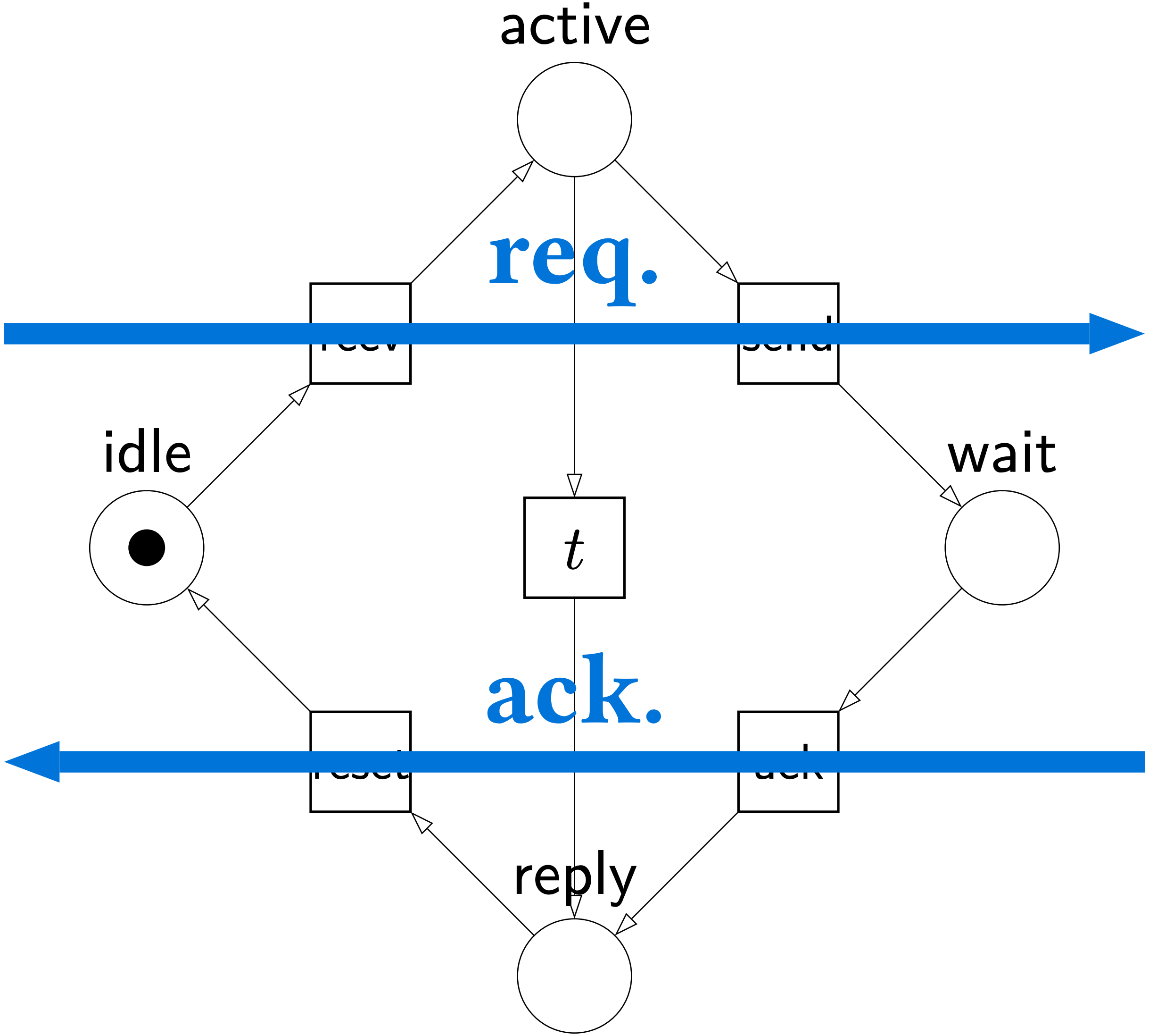
Construction of a Router



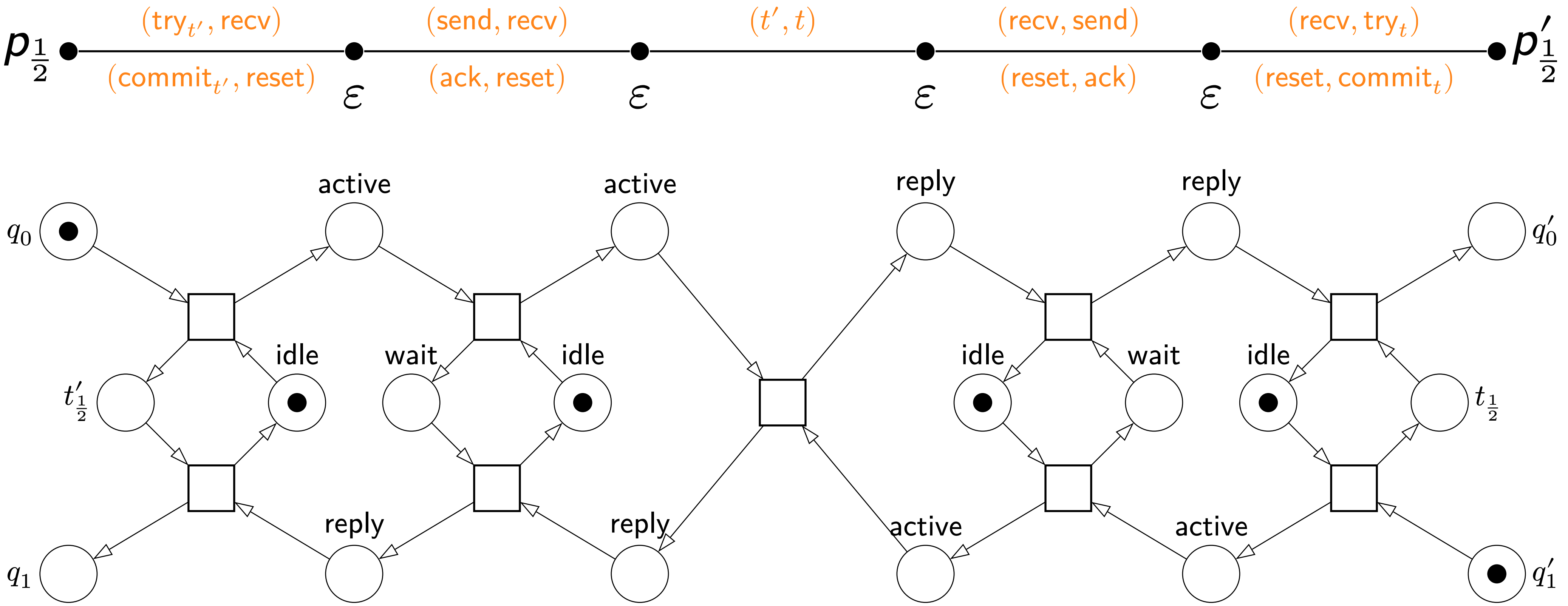
Construction of a Router



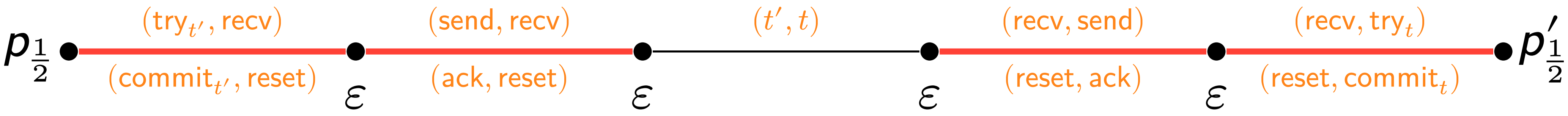
Construction of a Router

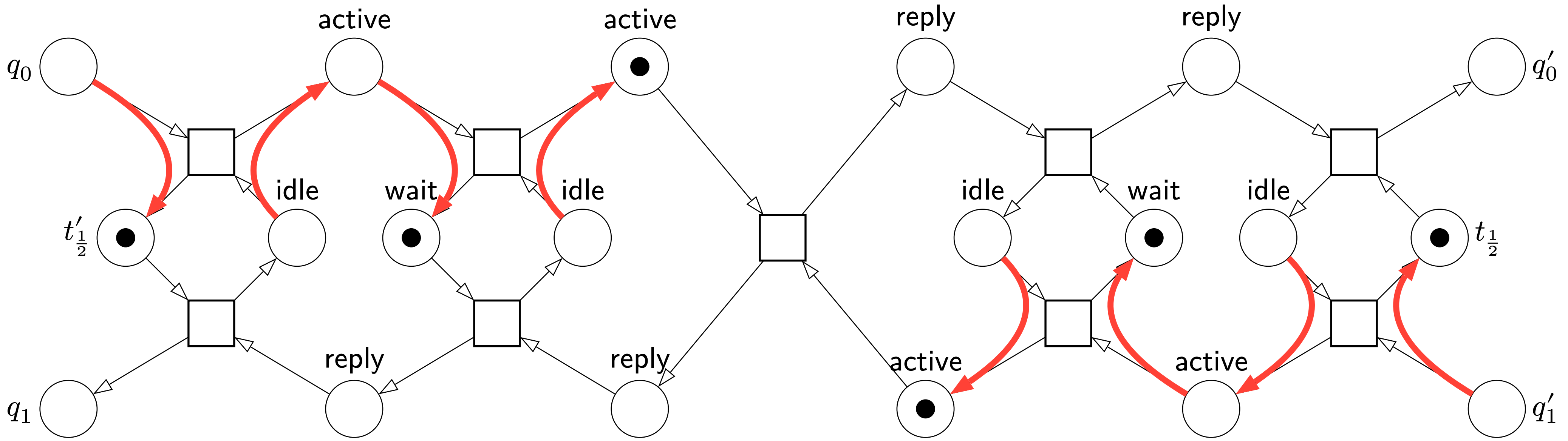


Communication through routers

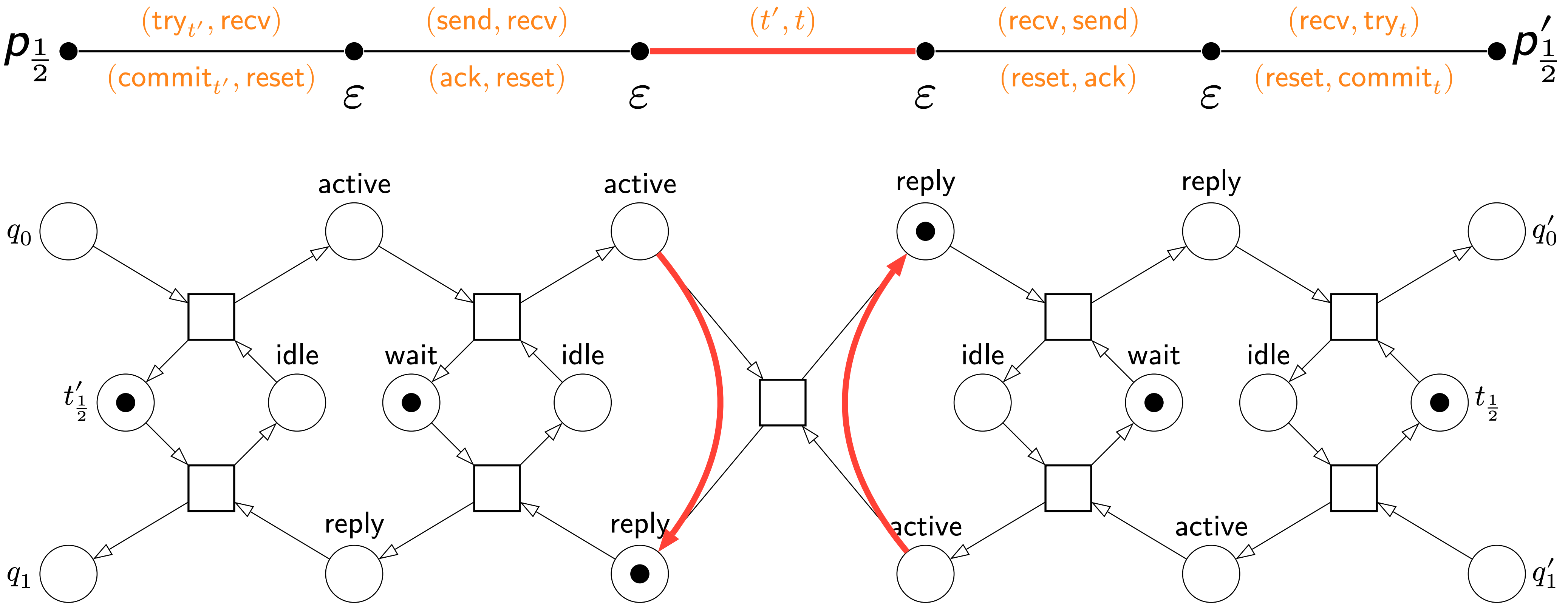


Communication through routers

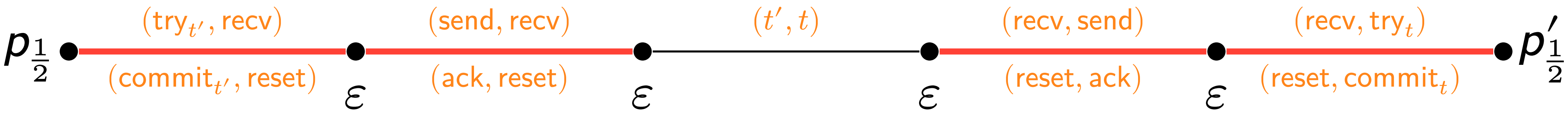


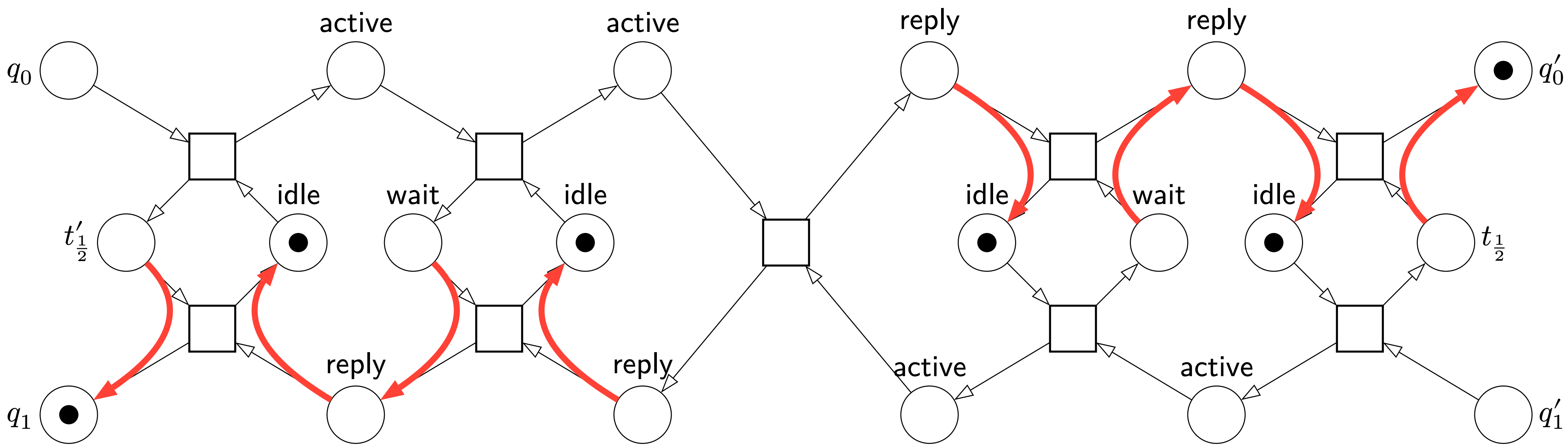


Communication through routers

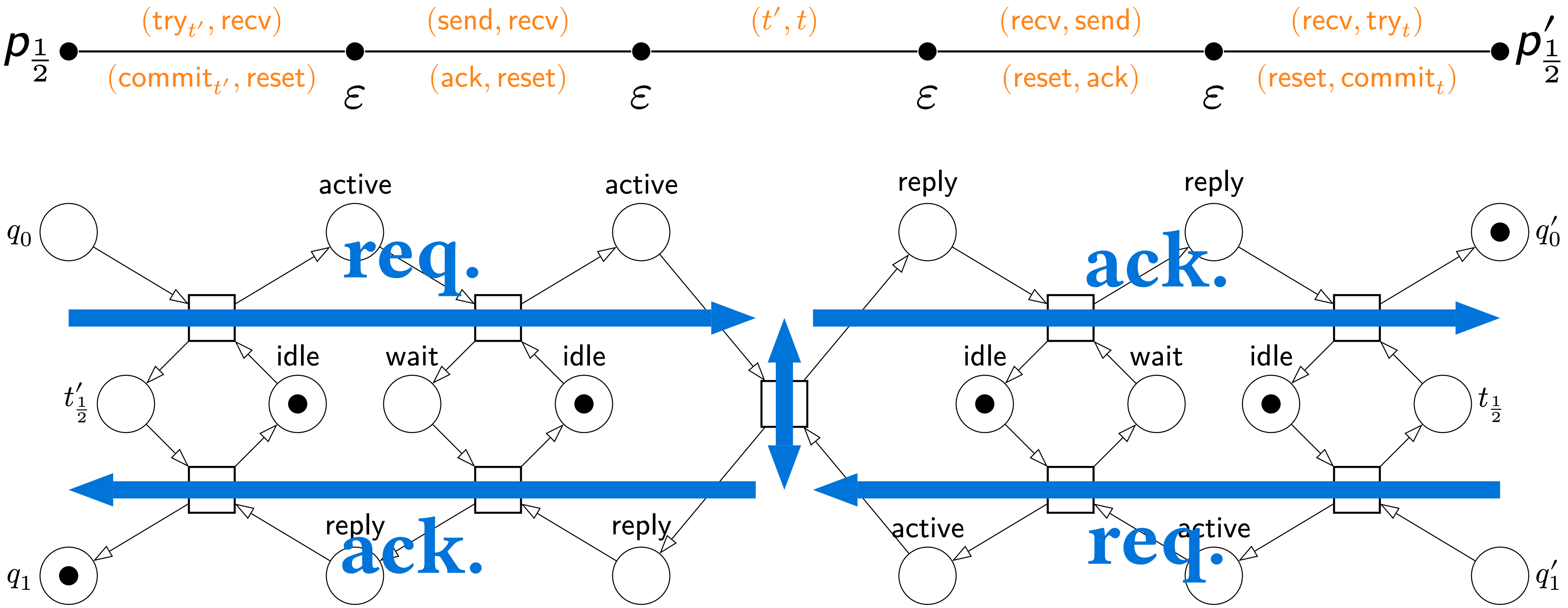


Communication through routers

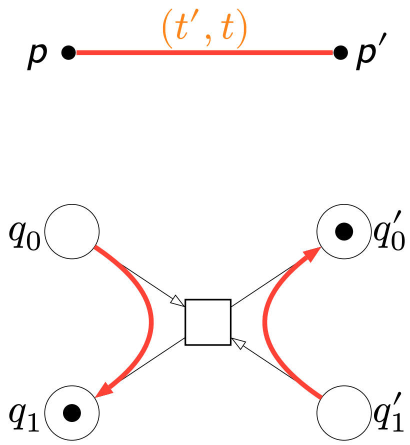




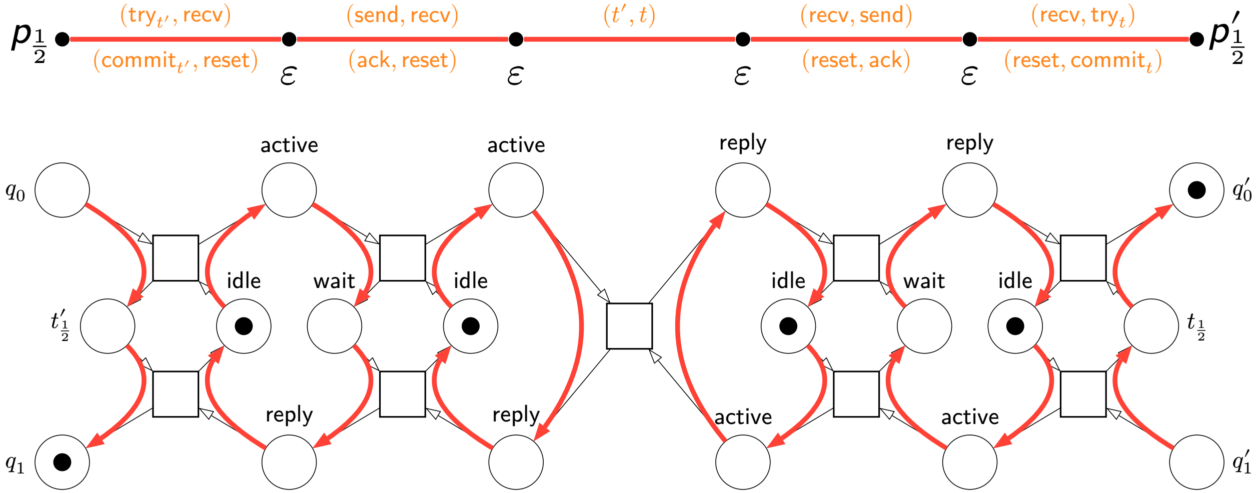
Communication through routers



Stuttering



$s_1 s_2$



$s_1 s_1 s_1 s_1 s_1 s_2 s_2 s_2 s_2 s_2$

Simulation

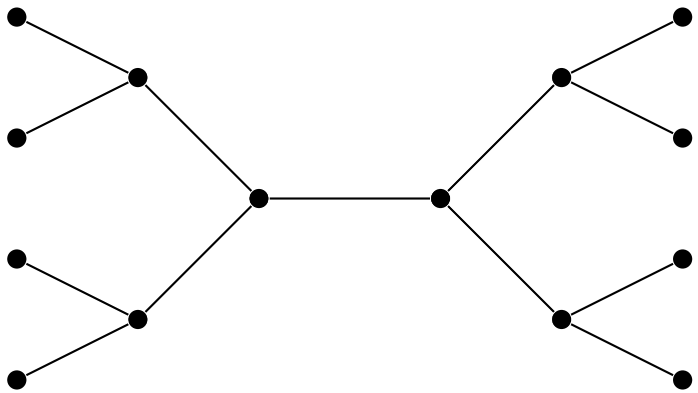
- Preserves all properties that are stuttering-invariant
  - (un)reachability, (un)coverability
  - mutual exclusion
  - reachability in a specific order
- Not preserved:
  - ~~next-step~~ ($s_1 s_2$ vs $s_1 s_1 s_1 s_1 s_1 s_2$)
  - ~~deadlock~~ ($s_1 \bot$ vs $s_1 s_1 s_1 s_1 \bot$)

Simulation

- Linear transformation
  - $|T| \cdot \mathrm{cw} \cdot \Theta(n)$ router nodes
  - sparse graph
- Downside: trace length
  - $\times\ \Theta(n)$ worst-case
  - $\times\ \Theta(\lg n)$ average-case

Simulation

- Linear transformation
  - $|T| \cdot \mathrm{cw} \cdot \Theta(n)$ router nodes
  - sparse graph
- Downside: trace length
  - $\times\ \Theta(n)$ worst-case
  - $\times\ \Theta(\lg n)$ average-case



5. Translation

Invariants

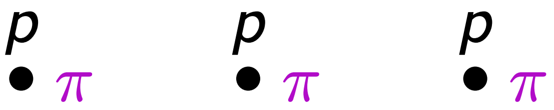
VR

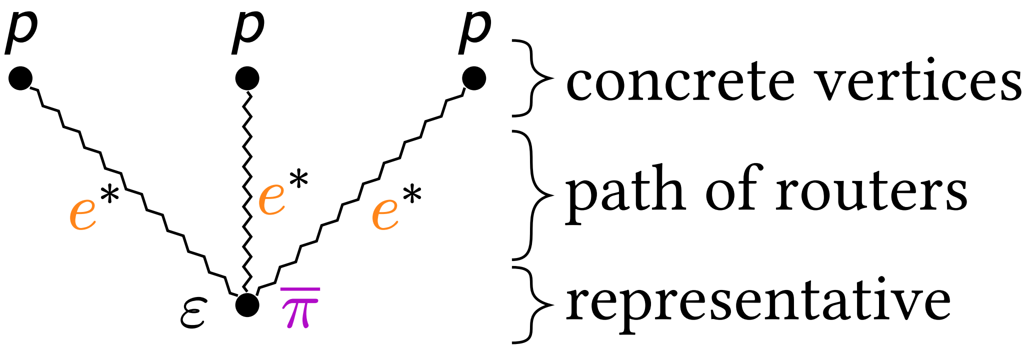
$$\xrightarrow{(t,t')}$$

HR

$$\overset{e^*}{\rightsquigarrow} \xrightarrow{(t,t')} \overset{e^*}{\leftsquigarrow}$$

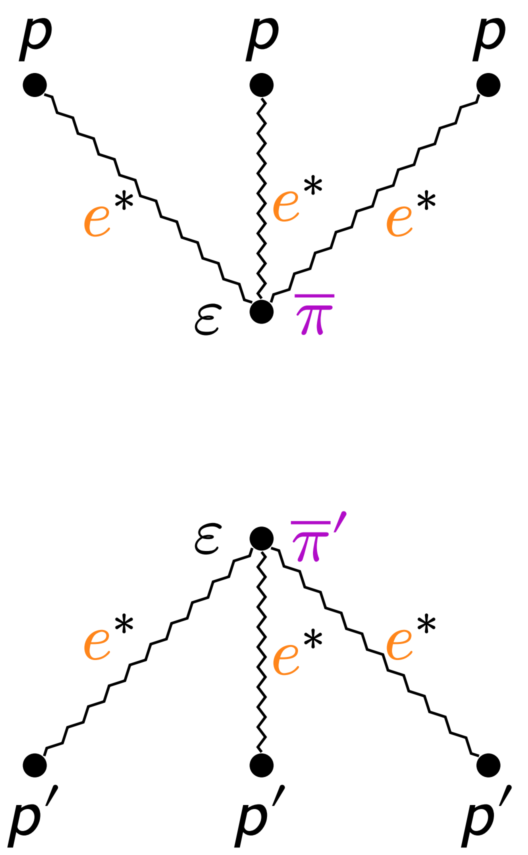
$$\xrightarrow{H}$$





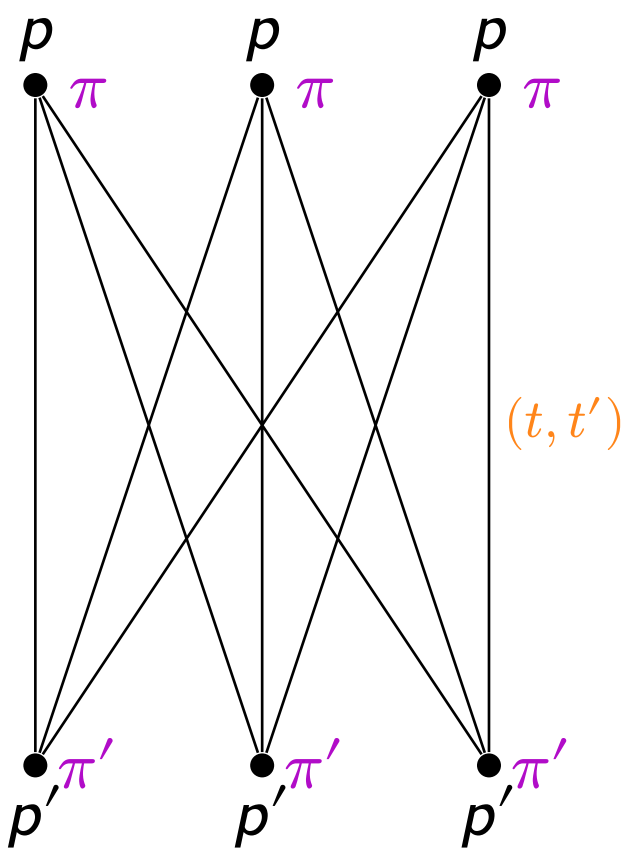
Edge creation

θ

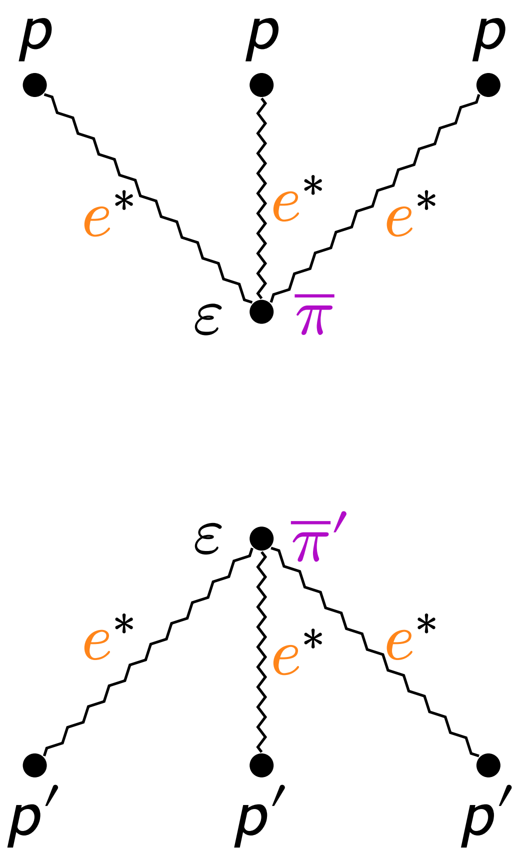


$H(\theta)$

Edge creation

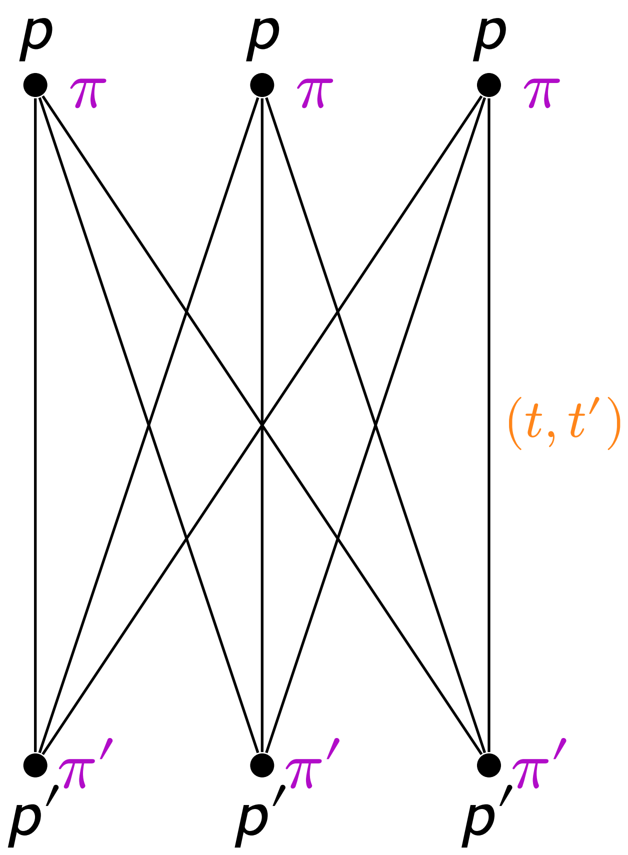


$\mathsf{add}_{\pi,\pi'}^{(t,t')}(\theta)$

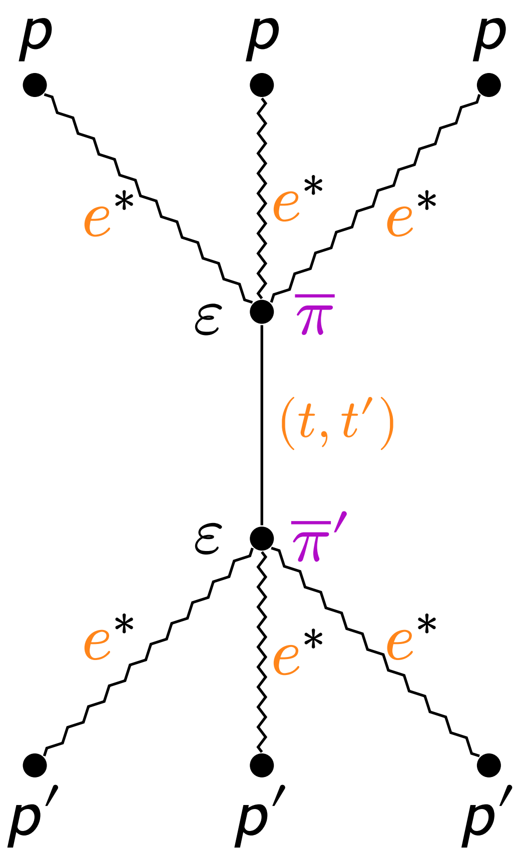


$H(\theta)$

Edge creation

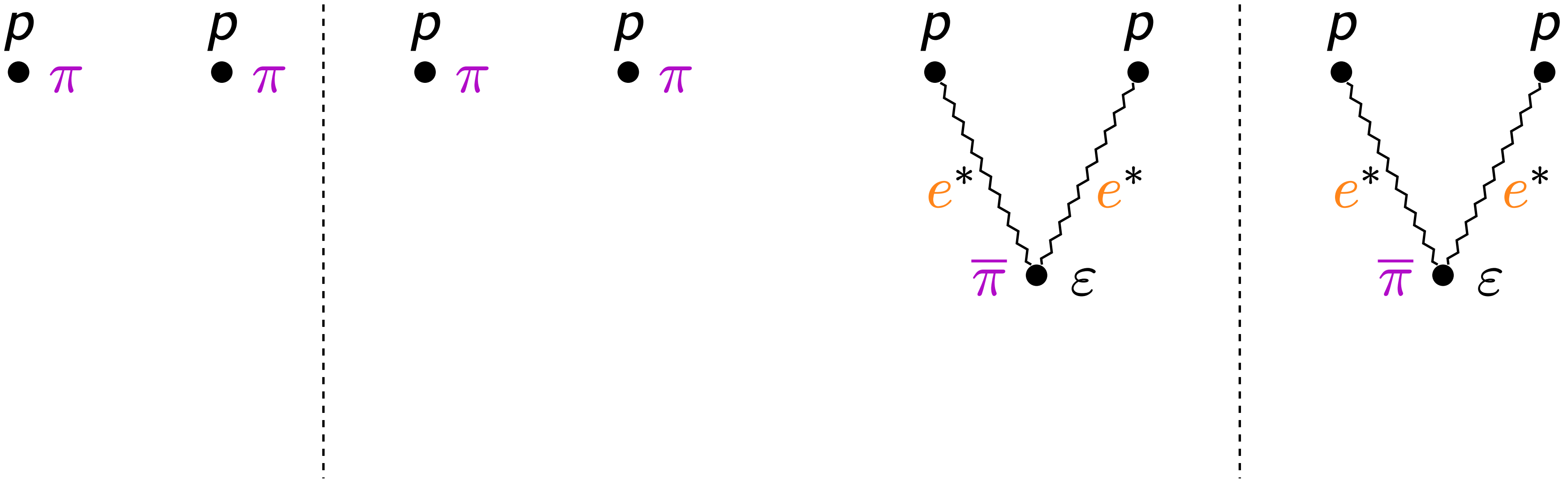


$\mathsf{add}^{(t,t')}_{\pi,\pi'}(\theta)$



$H(\theta) \parallel \overrightarrow{(t, t')}_{\overline{\pi},\overline{\pi}'}$

Union

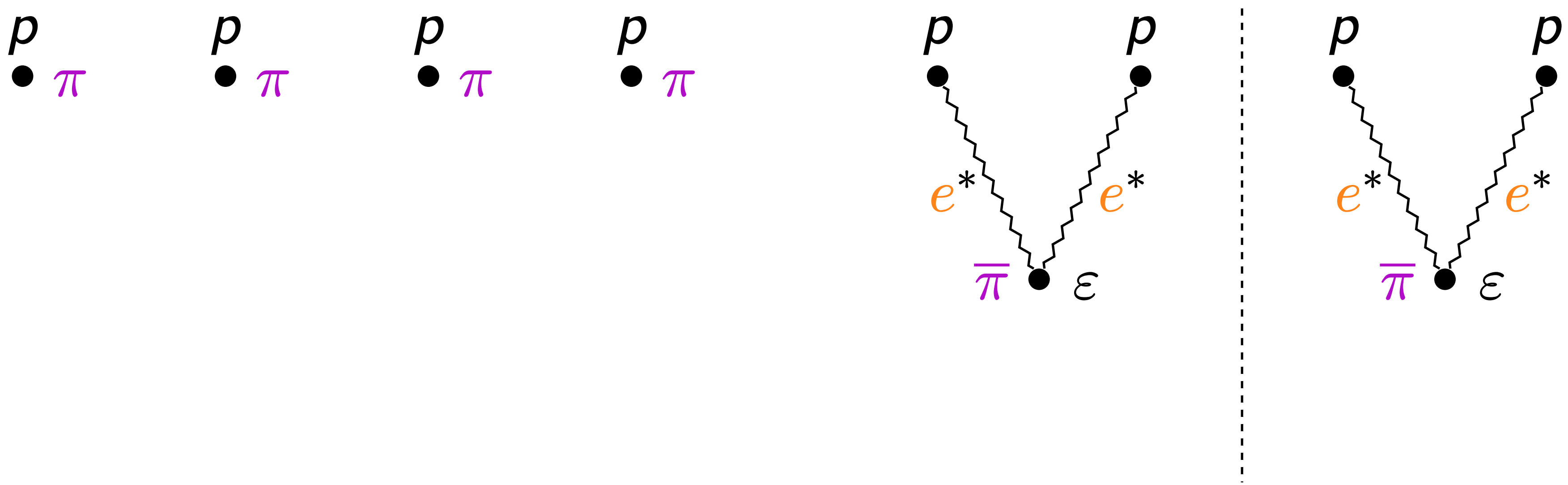


θ_1 θ_2

$H(\theta_1)$

$H(\theta_2)$

Union

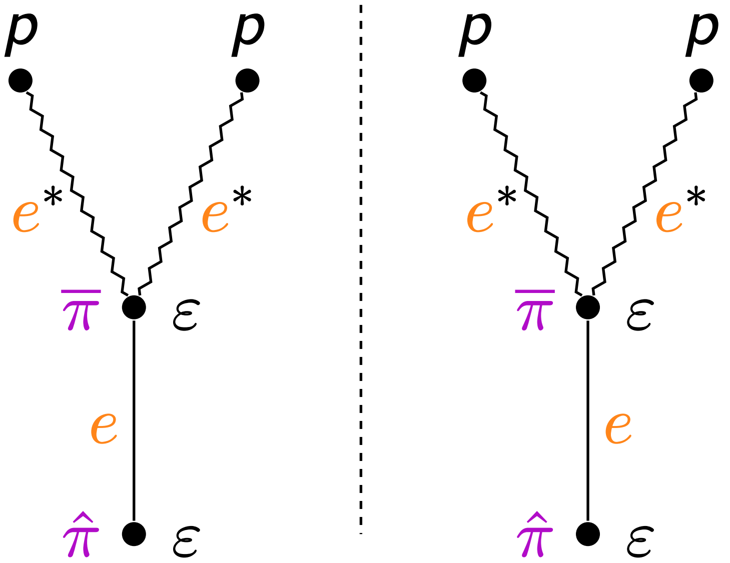


$\theta_1 \oplus \theta_2$

$H(\theta_1)$

$H(\theta_2)$

Union

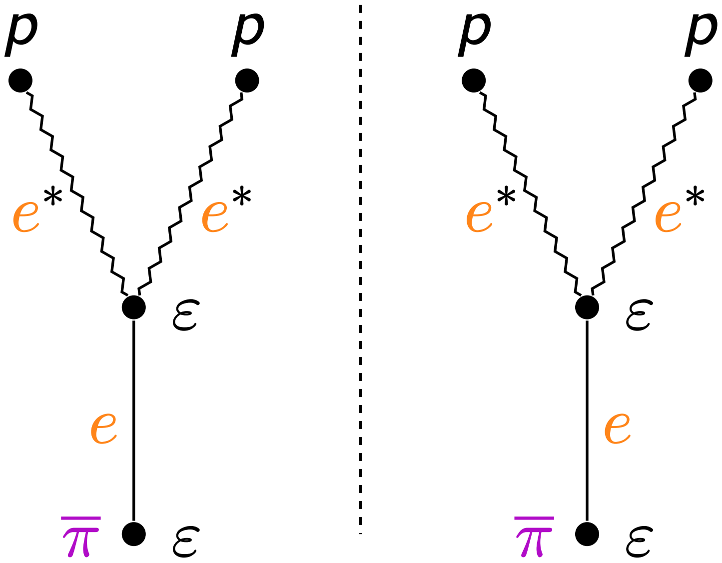


$$\theta_1 \oplus \theta_2$$

$$H(\theta_1) \parallel \vec{e}_{\overline{\pi},\hat{\pi}}$$

$$H(\theta_2) \parallel \vec{e}_{\overline{\pi},\hat{\pi}}$$

Union

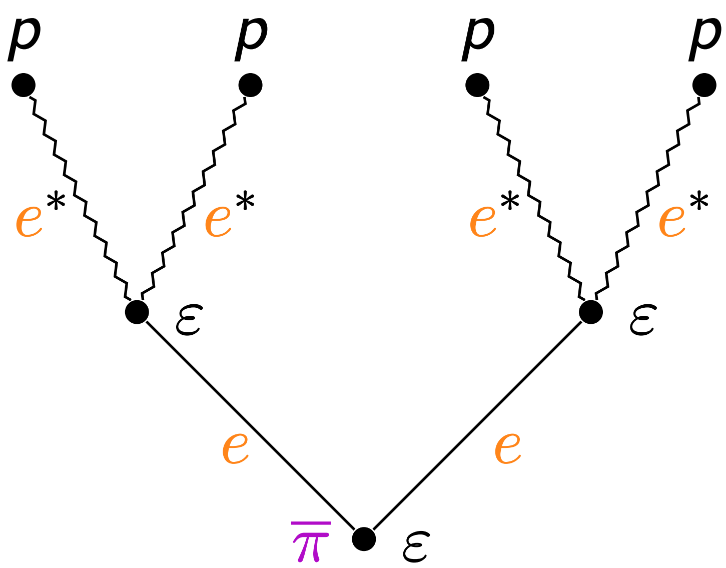


$$\theta_1 \oplus \theta_2$$

$$\mathsf{relab}_{[\hat{\pi} \mapsto \overline{\pi}]}(H(\theta_1) \parallel \vec{e}_{\overline{\pi}, \hat{\pi}})$$

$$\mathsf{relab}_{[\hat{\pi} \mapsto \overline{\pi}]}(H(\theta_2) \parallel \vec{e}_{\overline{\pi}, \hat{\pi}})$$

Union



$$\theta_1 \oplus \theta_2$$

$$\mathsf{relab}_{[\hat{\pi} \mapsto \overline{\pi}]}\left(H(\theta_1) \parallel \vec{e}_{\overline{\pi},\hat{\pi}}\right)$$

$$\parallel \mathsf{relab}_{[\hat{\pi} \mapsto \overline{\pi}]}\left(H(\theta_2) \parallel \vec{e}_{\overline{\pi},\hat{\pi}}\right)$$

6. Conclusion

- Translation of systems from **VR to HR**
- Preserves **stuttering-invariant** properties
- Enables applying results proven on HR to **dense families**

Future work

- Implementation
- Could be made TPS