Programming
Weak Synchronization
Models

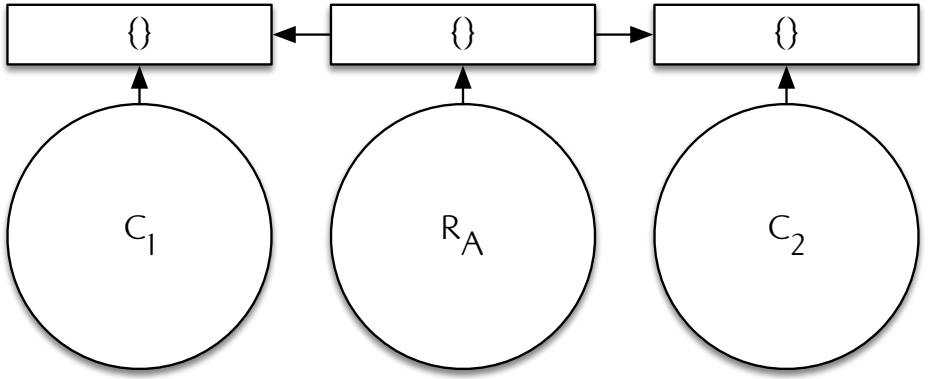
Christopher S. Meiklejohn
Université catholique de Louvain, Belgium
Instituto Superior Técnico, Portugal

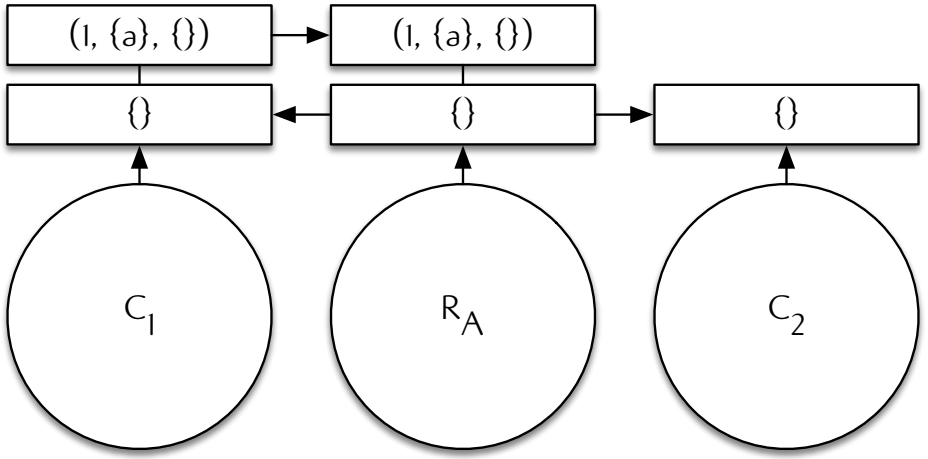


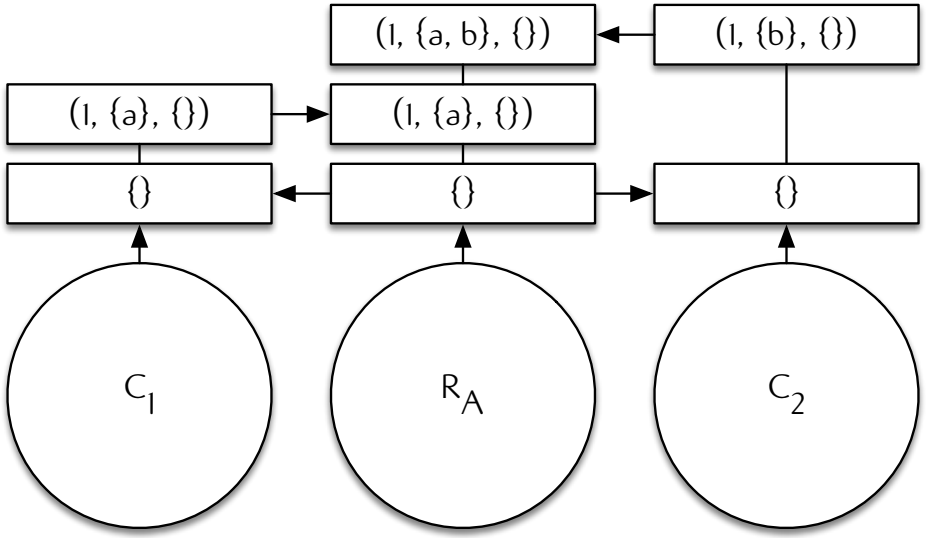

Processes

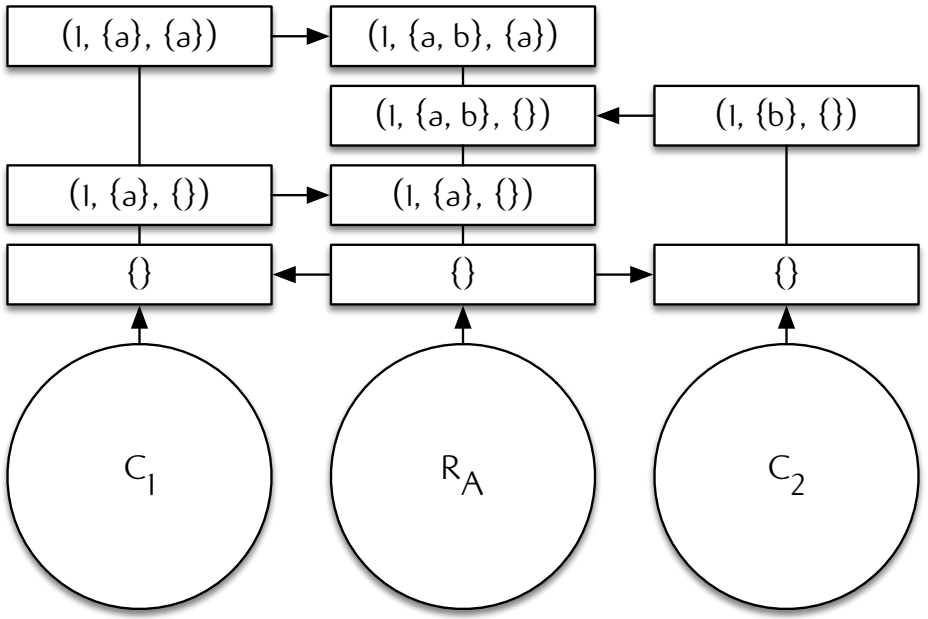
- Replicas as monotonic streams
  Each replica of a CRDT produces a **monotonic stream of states**
- Monotonic processes
  Read from one or more input replica streams and produce a single output replica stream
- Inflationary reads
  Read operation ensures that we only read **inflationary** updates to replicas

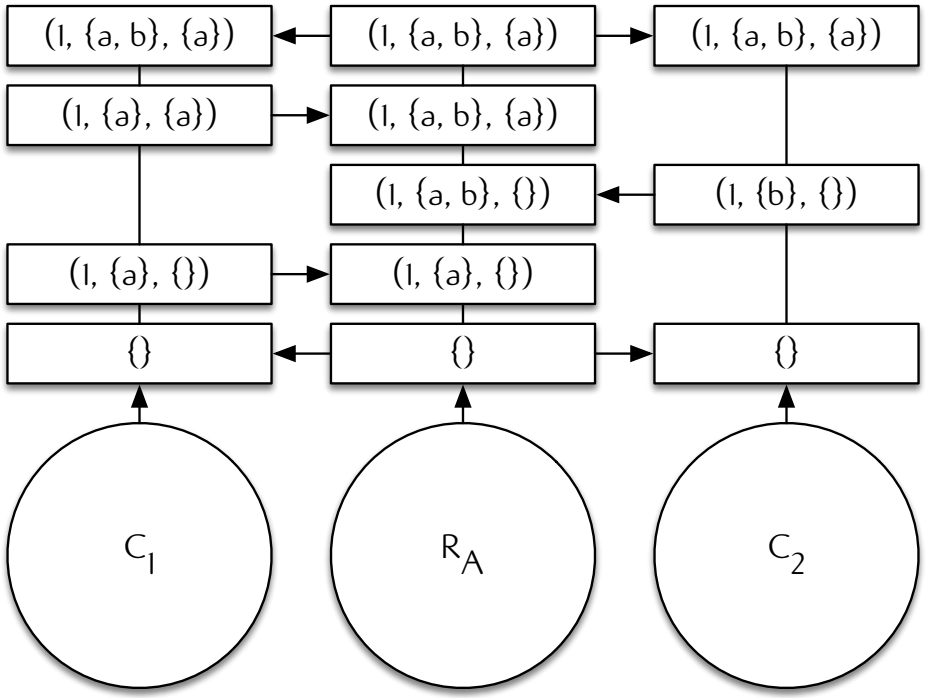
Lattice Processing
Monotonic Streams


{}
{}
{}
C_1
R_A
C_2

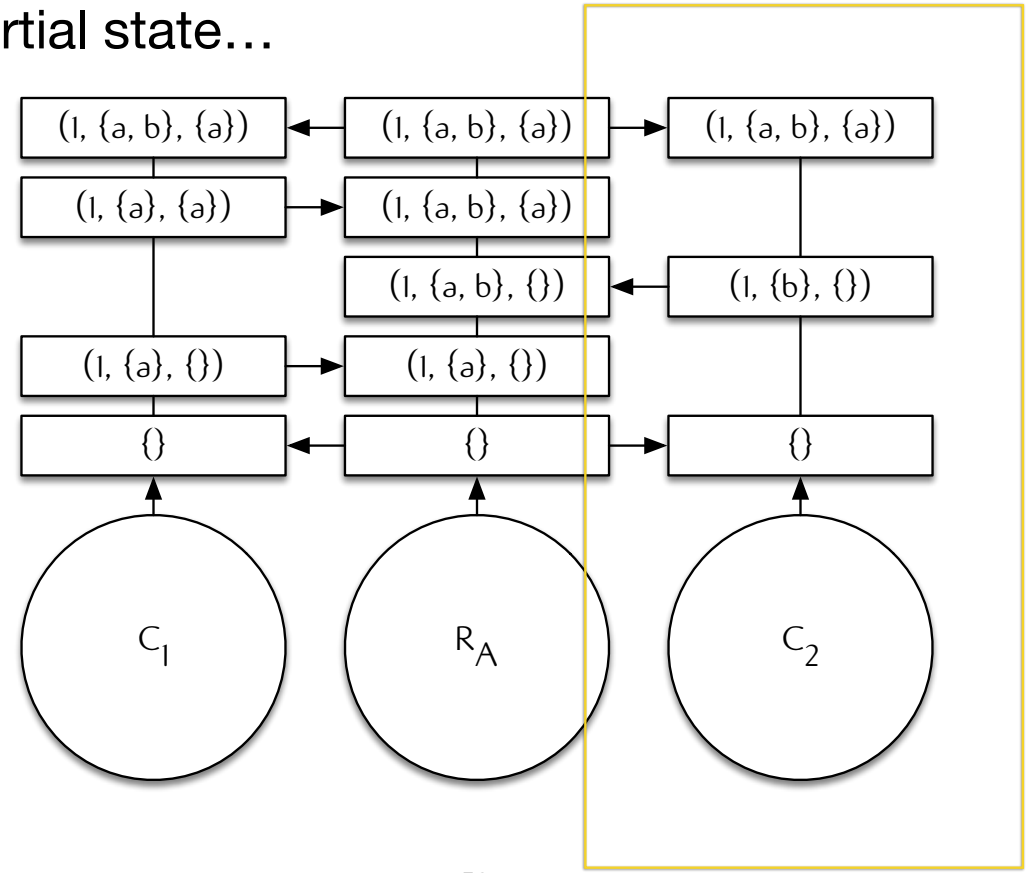

(l, {a}, {})
(l, {a}, {})
{}
{}
{}
C1
RA
C2


(1, {a, b}, {})
(1, {b}, {})
(1, {a}, {})
(1, {a}, {})
{}
{}
{}
C1
RA
C2

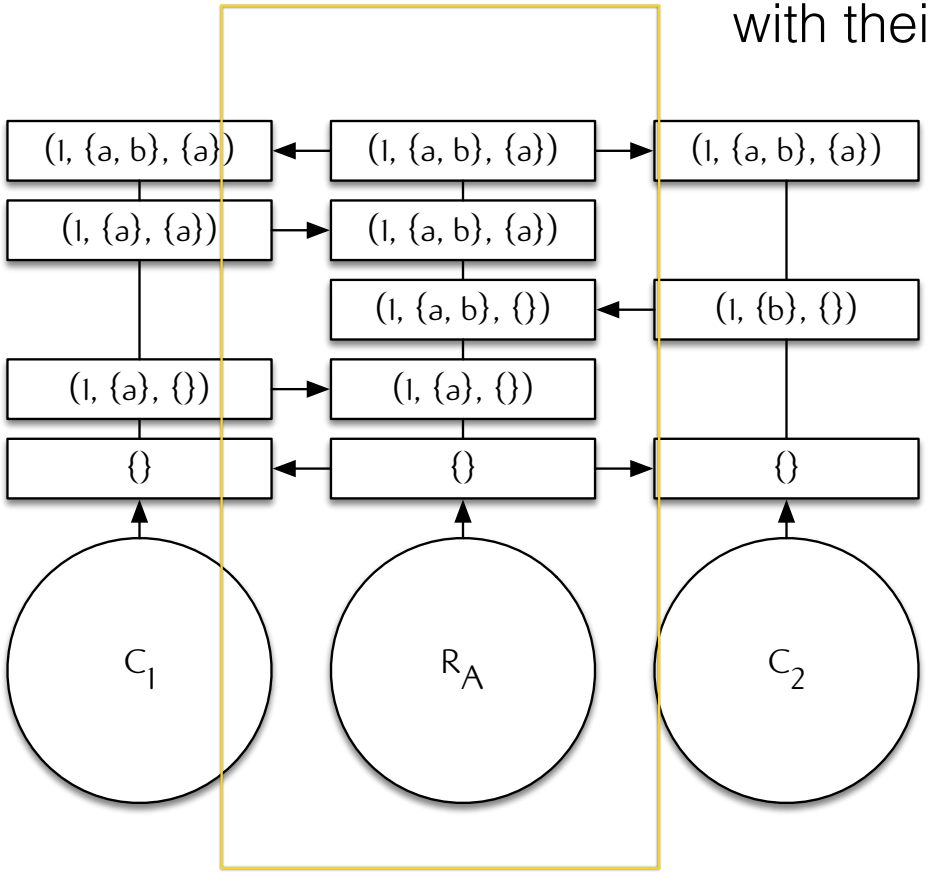

(I, {a}, {a})
(I, {a, b}, {a})
(I, {a, b}, {})
(I, {b}, {})
(I, {a}, {})
(I, {a}, {})
{}
{}
{}
C1
RA
C2


(I, {a, b}, {a})
(I, {a, b}, {a})
(I, {a, b}, {a})
(I, {a}, {a})
(I, {a, b}, {a})
(I, {a, b}, {})
(I, {b}, {})
(I, {a}, {})
(I, {a}, {})
{}
{}
{}
C1
RA
C2

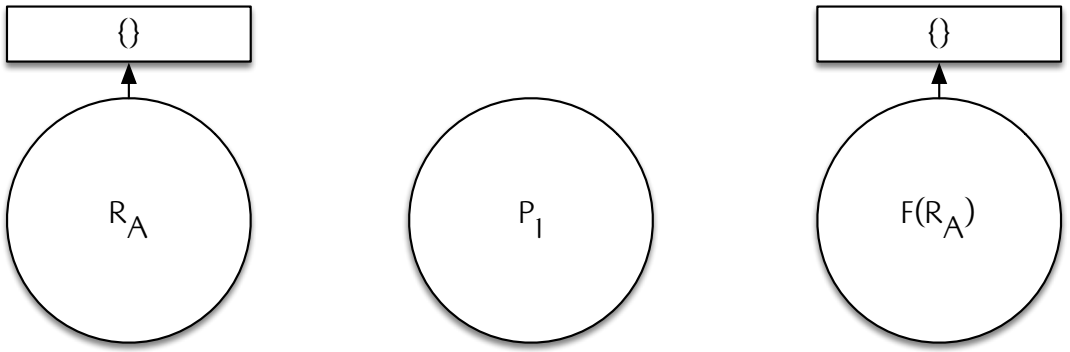
Clients can
operate with **partial state…**

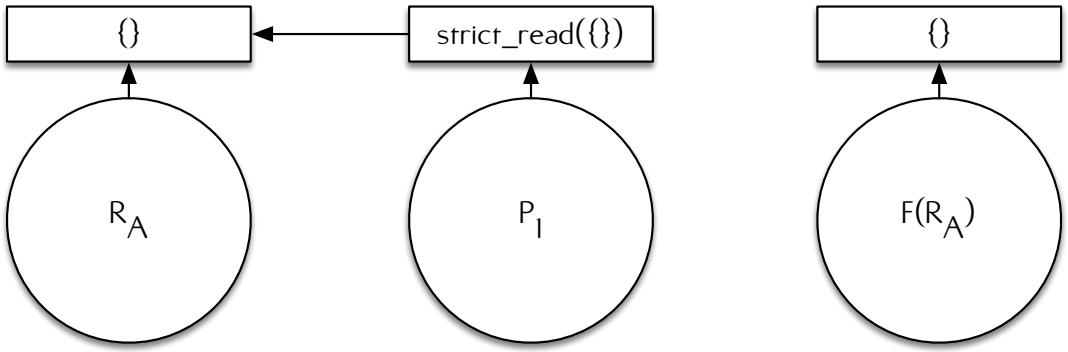


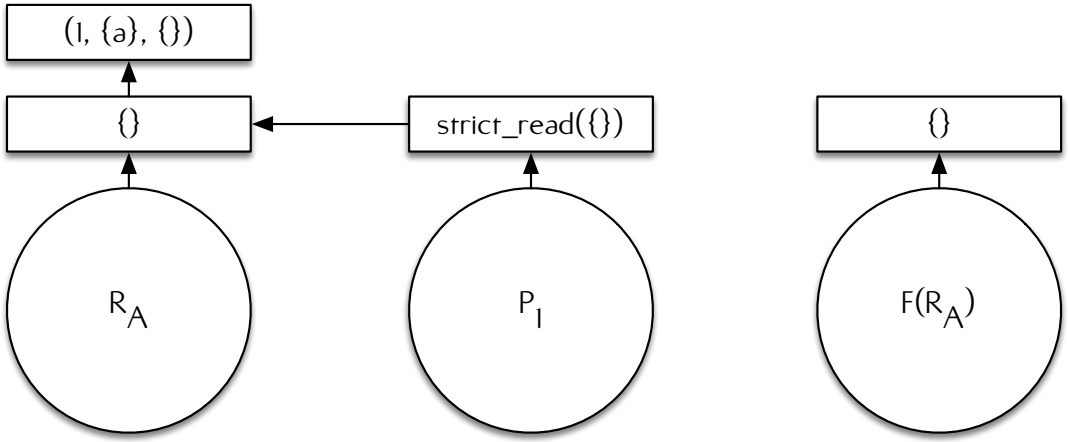
… and synchronize
with their **local replica**.

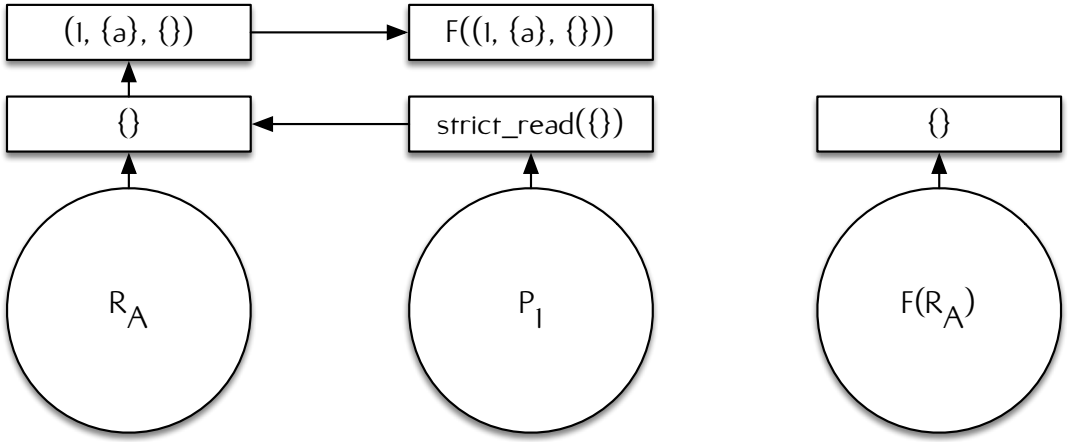


Lattice Processing
Monotonic Processes

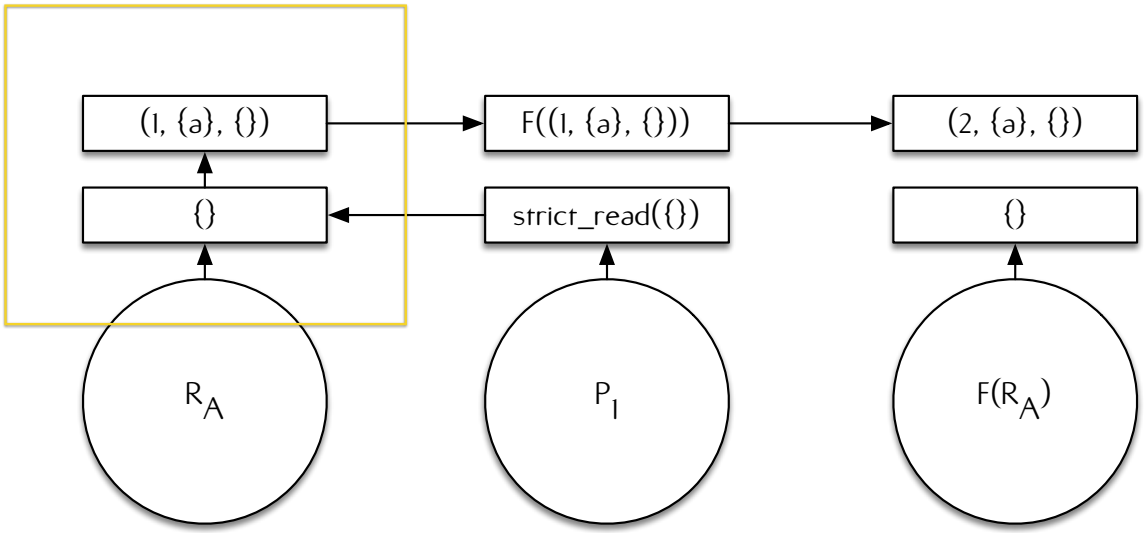

{}
R_A
P_I
{}
$F(R_A)$


{}
strict_read({})
{}
R_A
P_I
$F(R_A)$

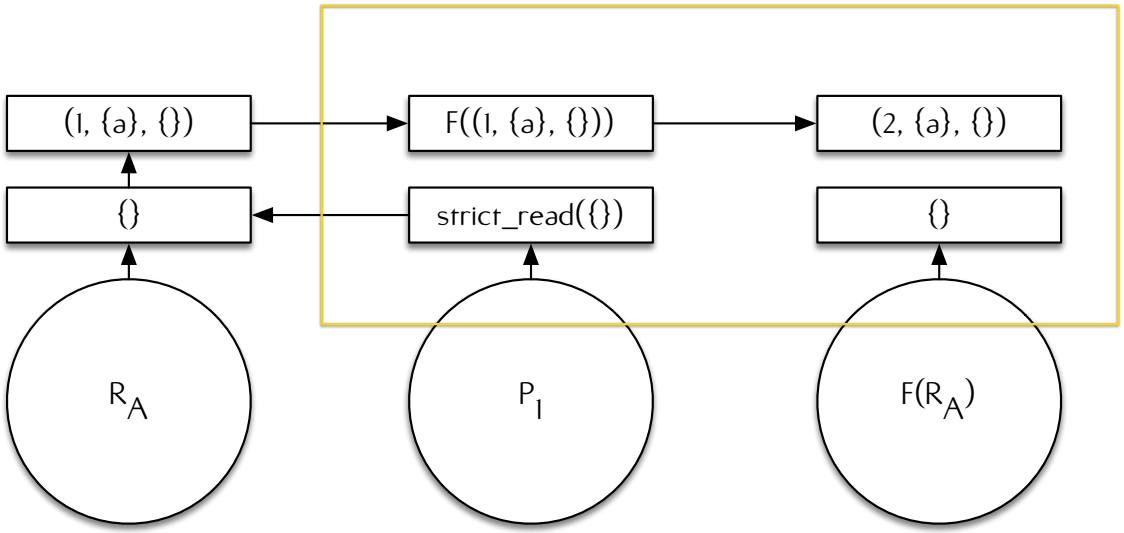

(I, {a}, {})
{}
strict_read({})
{}
R_A
P_I
$F(R_A)$

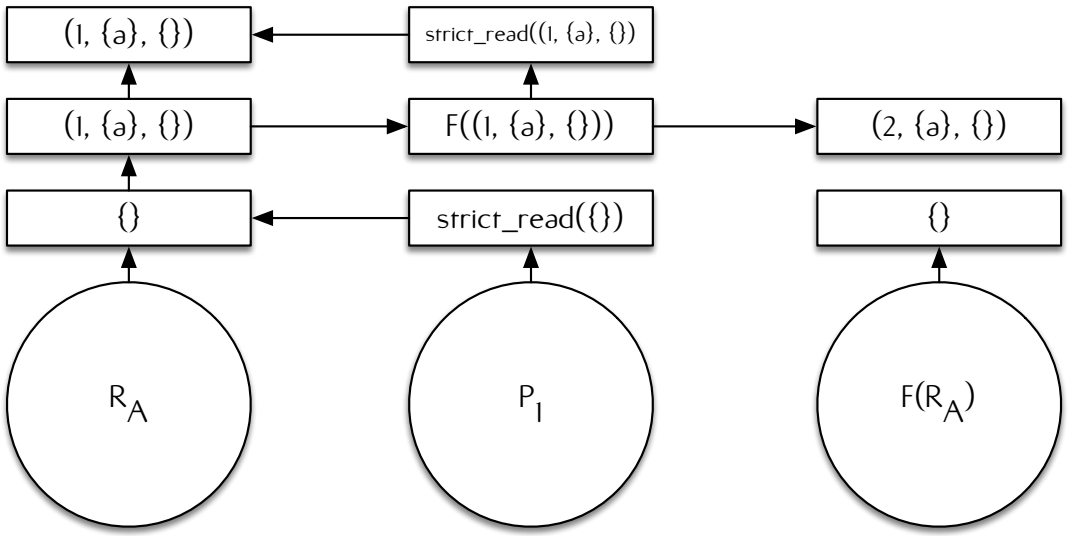

(I, {a}, {})
F((I, {a}, {}))
{}
strict_read({})
{}
R_A
P_I
$F(R_A)$

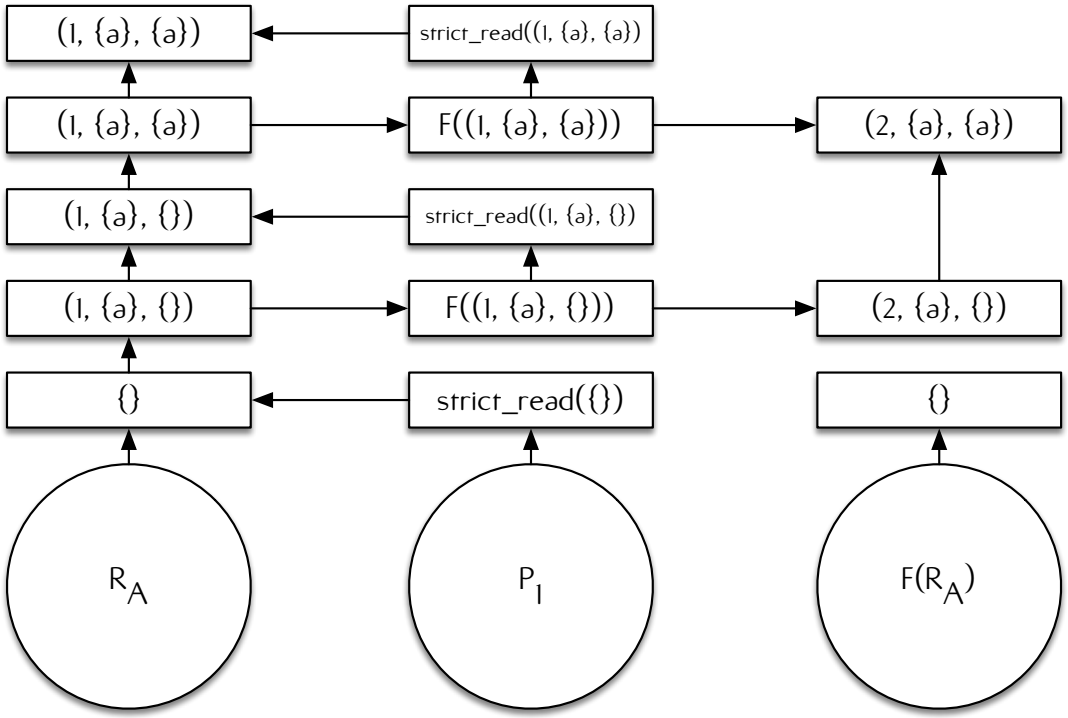
Every time **replica** changes…



….the **process** will compute a new result.




(1, {a}, {})
strict_read((1, {a}, {})
(1, {a}, {})
F((1, {a}, {}))
(2, {a}, {})
{}
strict_read({})
{}
R_A
P_1
$F(R_A)$


(1, {a}, {a})
strict_read((1, {a}, {a}))
(1, {a}, {a})
F((1, {a}, {a}))
(2, {a}, {a})
(1, {a}, {})
strict_read((1, {a}, {}))
(1, {a}, {})
F((1, {a}, {}))
(2, {a}, {})
{}
strict_read({})
{}
R_A
P_1
F(R_A)

Omitted interleaving
does not sacrifice **correctness.**