RHPL 2023

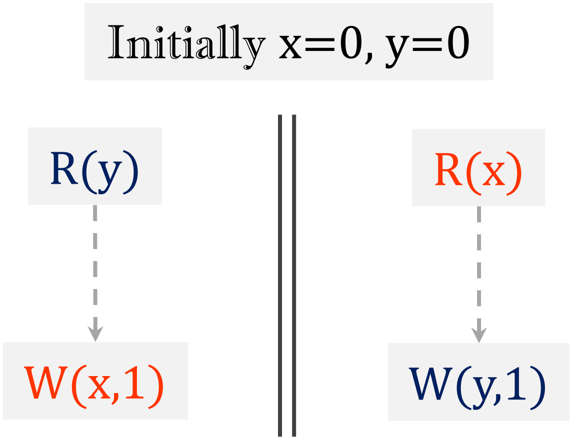
ViEqui

Optimal Stateless Model Checking based on *View-equivalence*

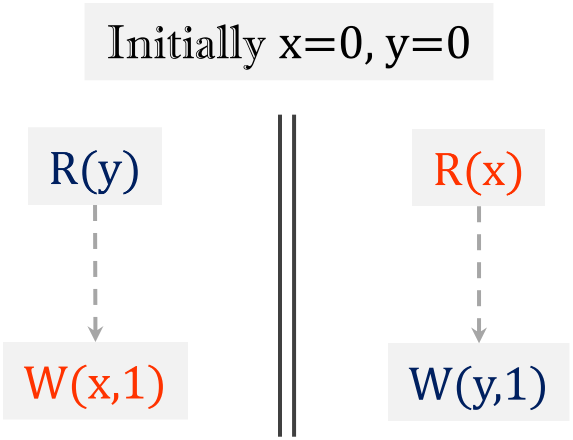
Sanjana Singh and Subodh Sharma

Indian Institute of Technology Delhi

Interleaving model of concurrency

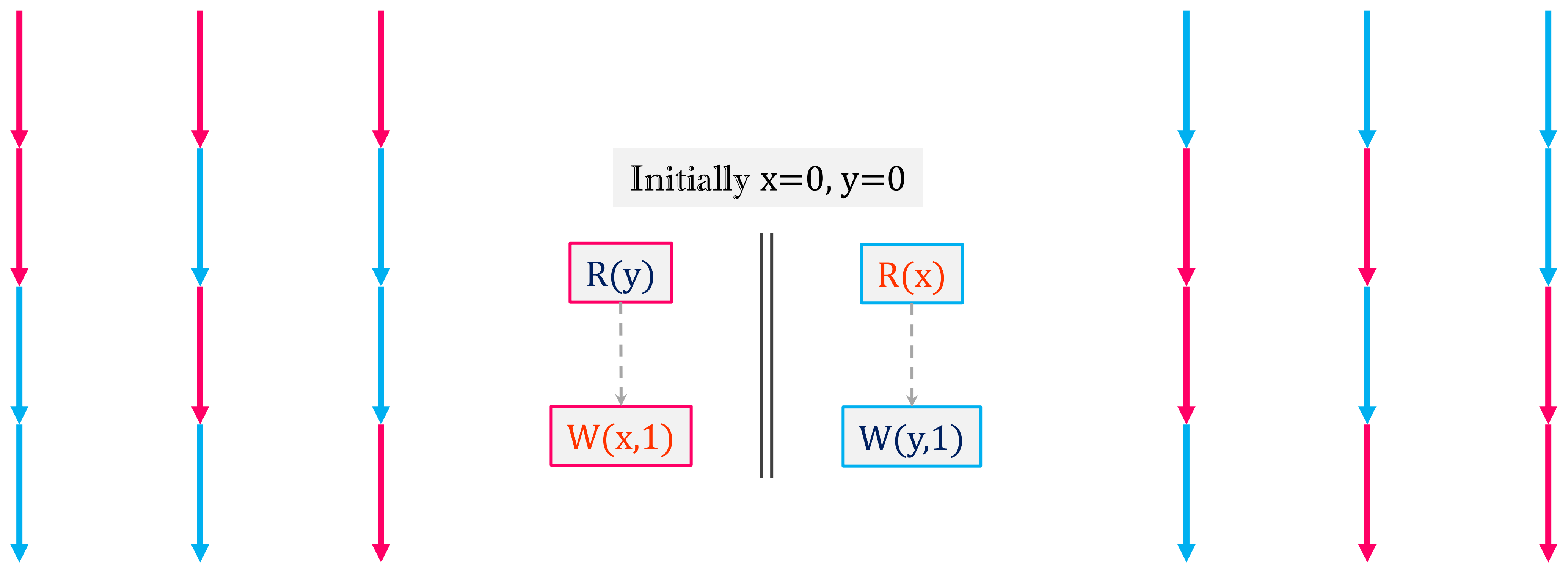


Interleaving model of concurrency

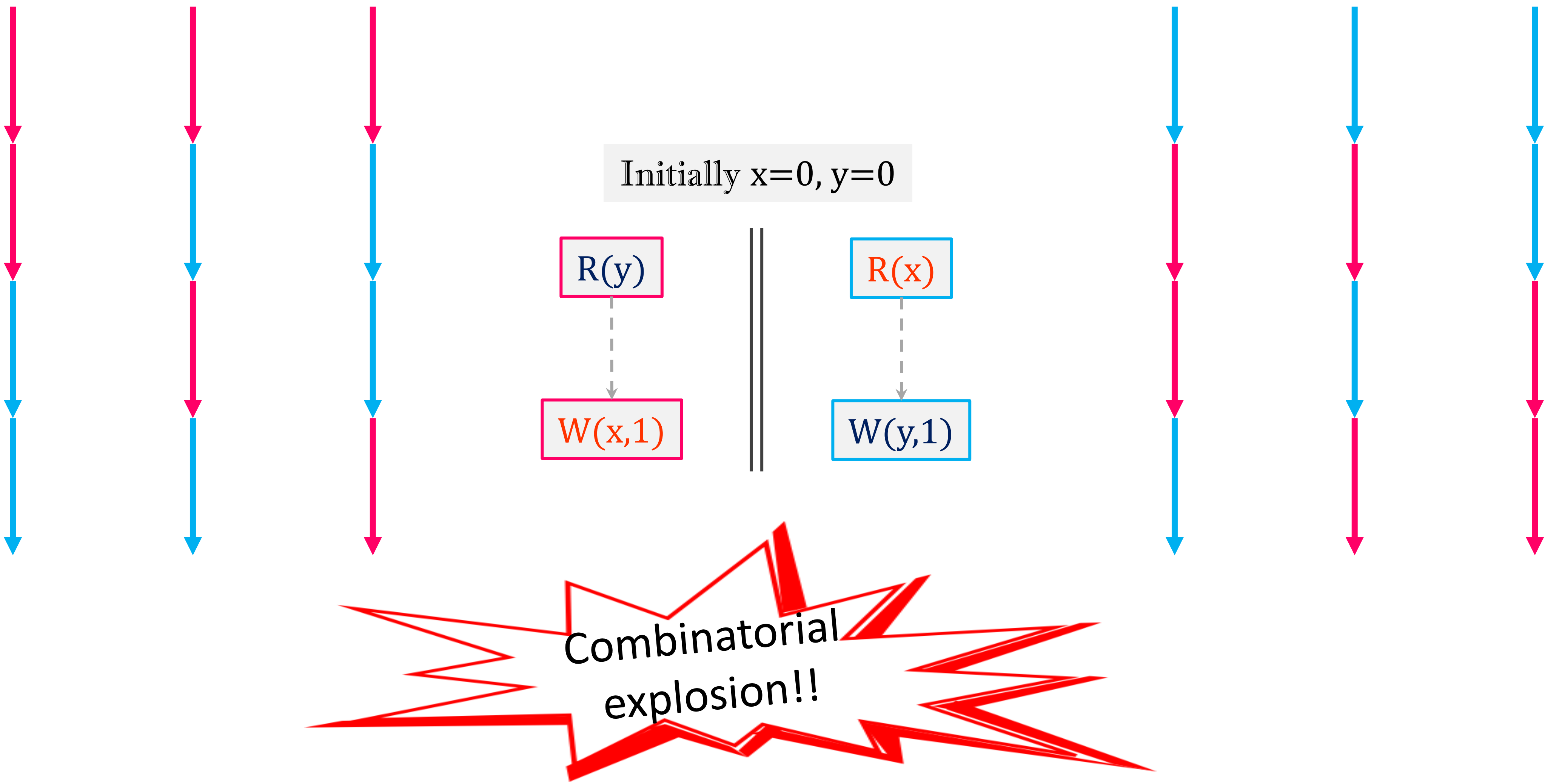


ψ satisfied ?

Interleaving model of concurrency

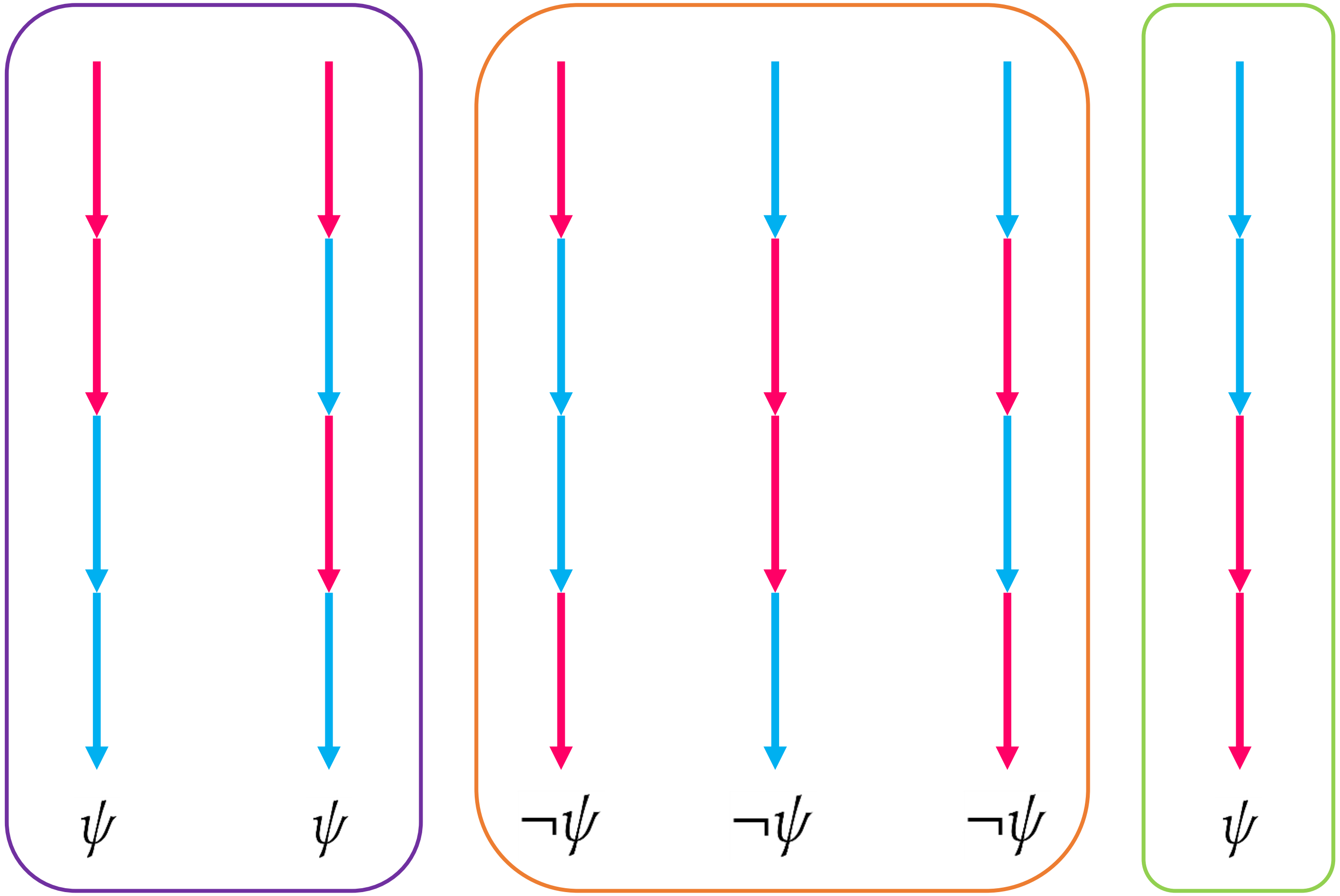


Interleaving model of concurrency



Equivalence classes

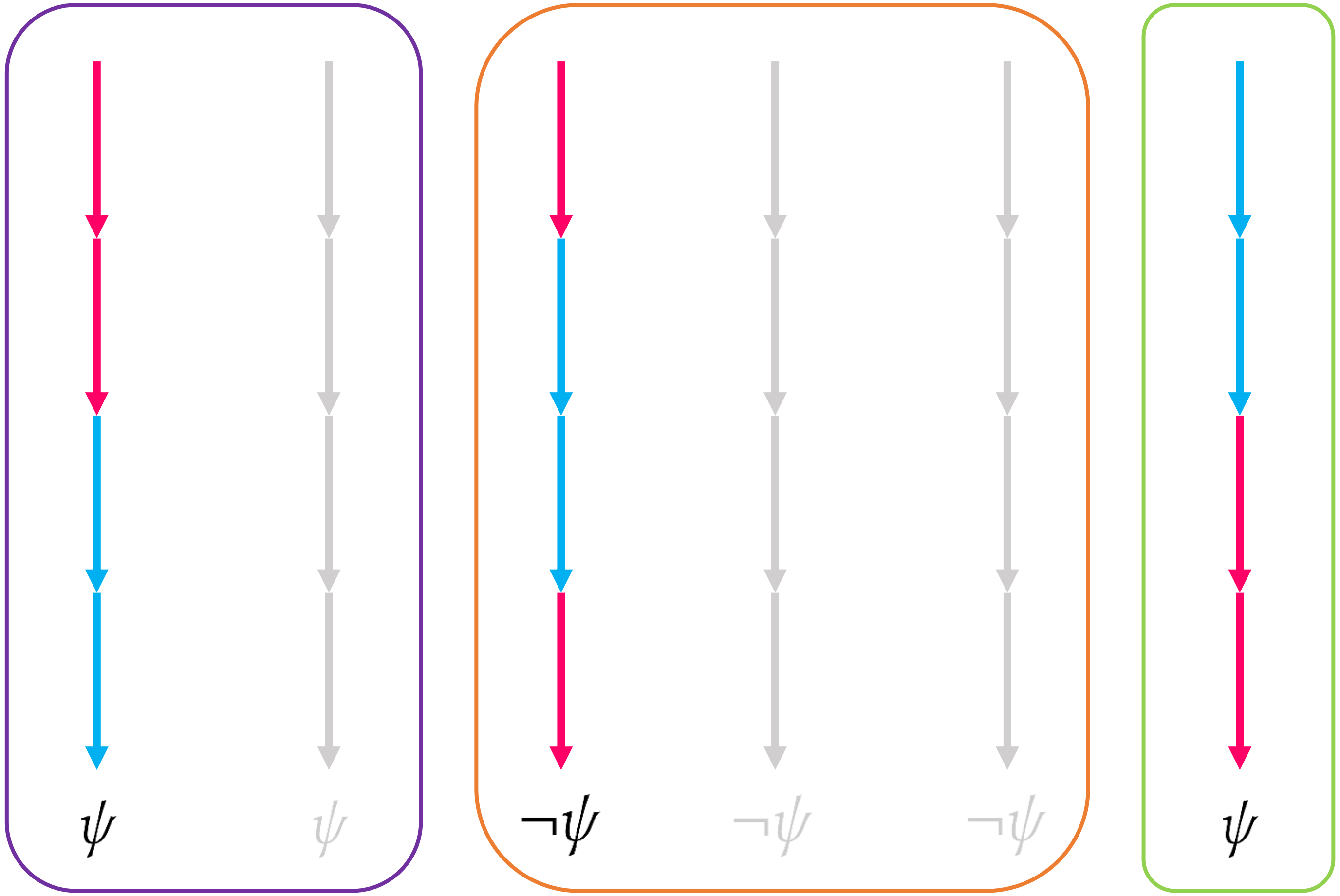
Stateless model checkers partition executions into *equivalence classes* based on *equivalence relations*



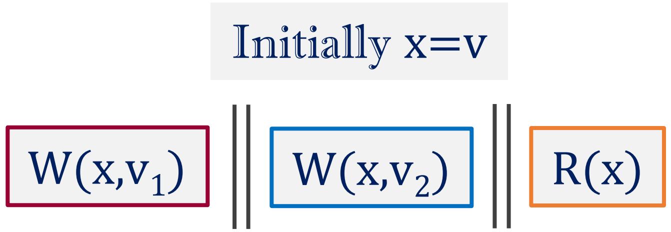
Stateless model checkers explore a reduced state graph

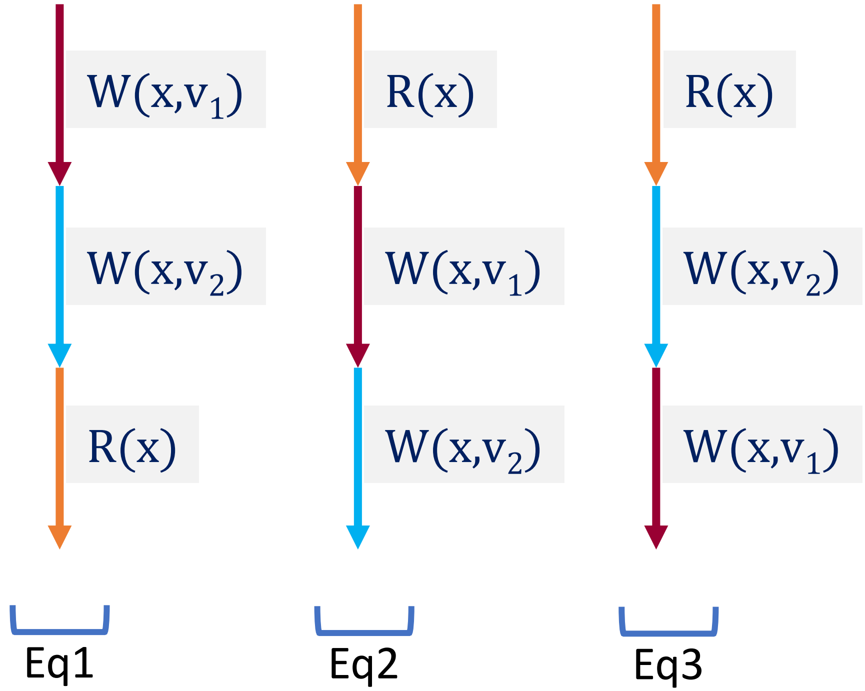
Stateless model checkers partition executions into

equivalence classes based on *equivalence relations*



Existing equivalence relations



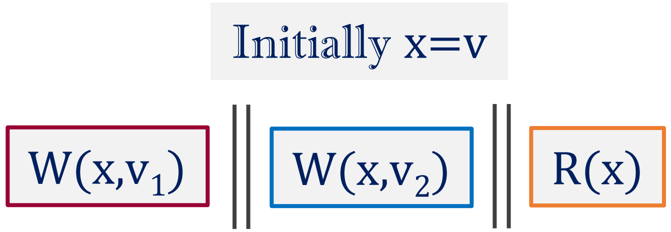


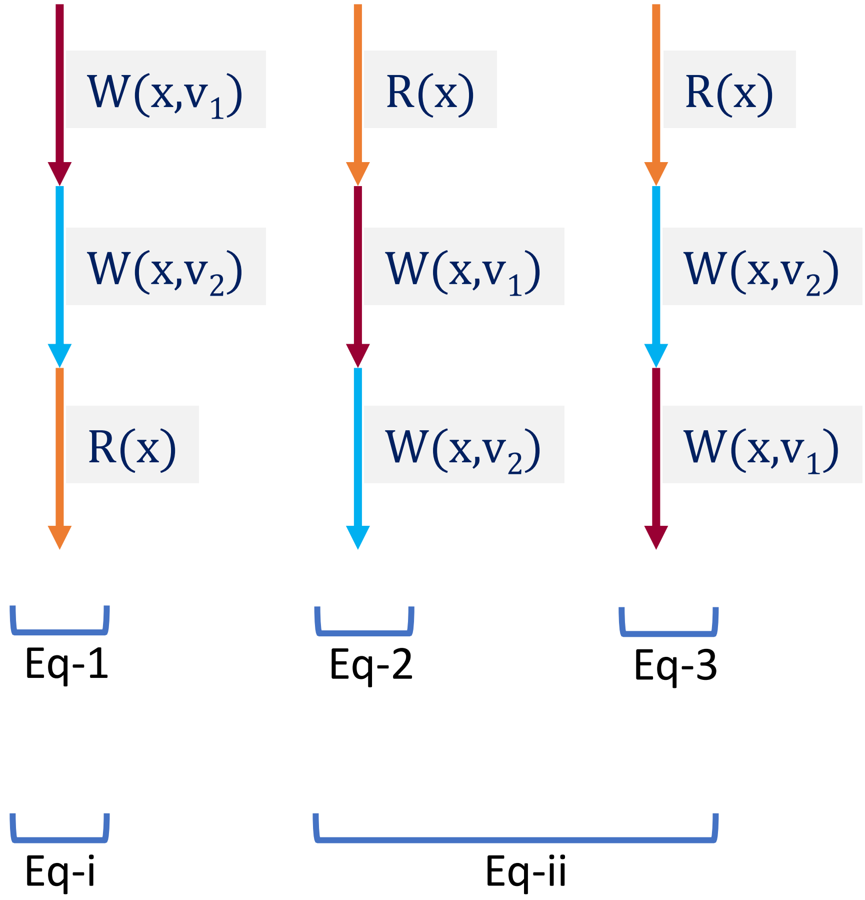
- Classical (*Mazurkiewicz*)

Same order of occurrence on racing event pairs

[Flanagan & Godefroid, POPL'05], [Abdulla et al., POPL'14] [Nguyen et al., CAV'18] [Zhang et al., PLDI'15] [Abdulla et al., TACAS'15]

Existing equivalence relations



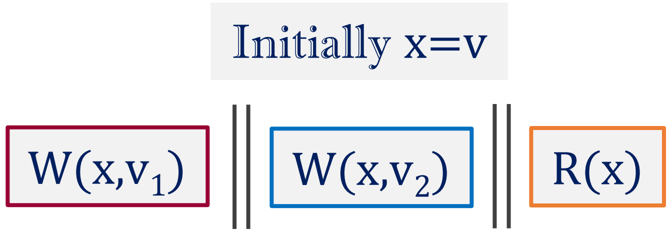


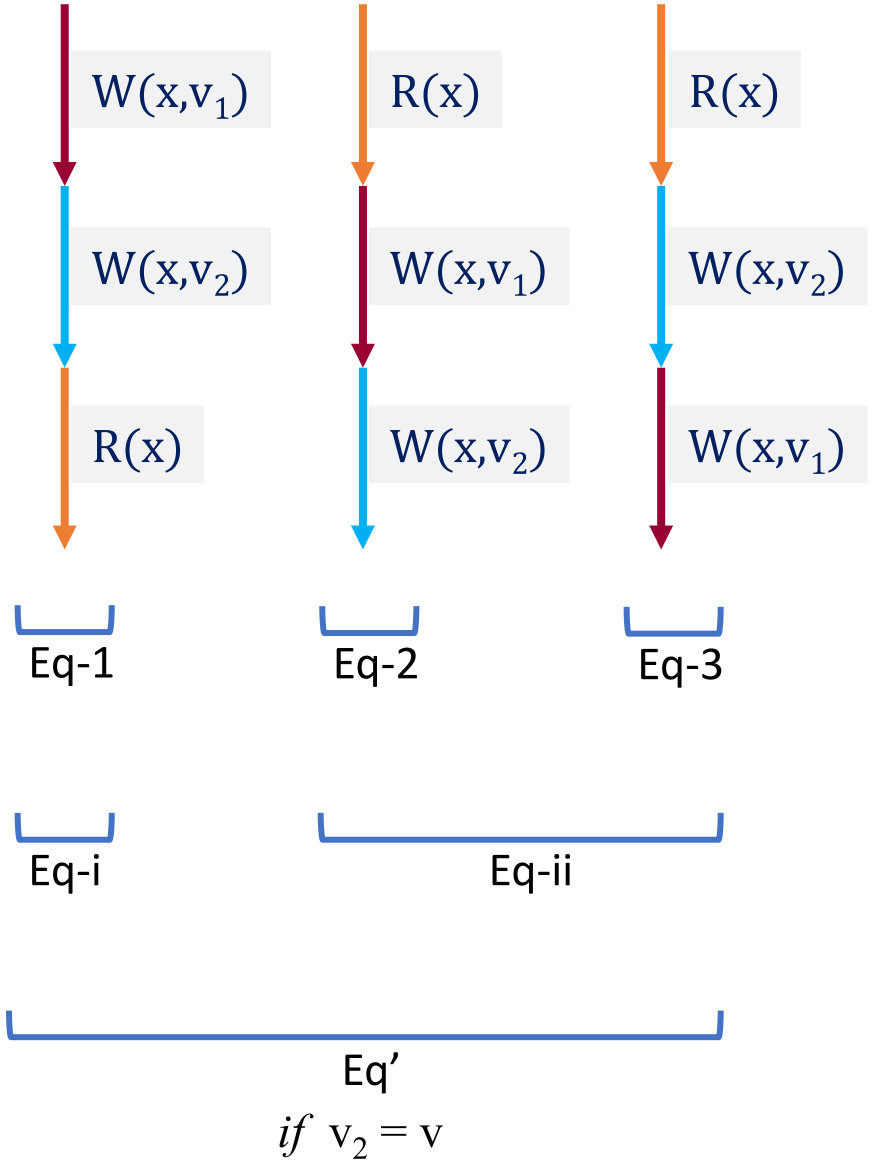
- Classical (*Mazurkiewicz*)
- reads-from

All reads read-from the same write

[Albert et al., CAV'17] [Chalupa et al., POPL'18] [Abdulla et al., OOPSLA'18]

Existing equivalence relations



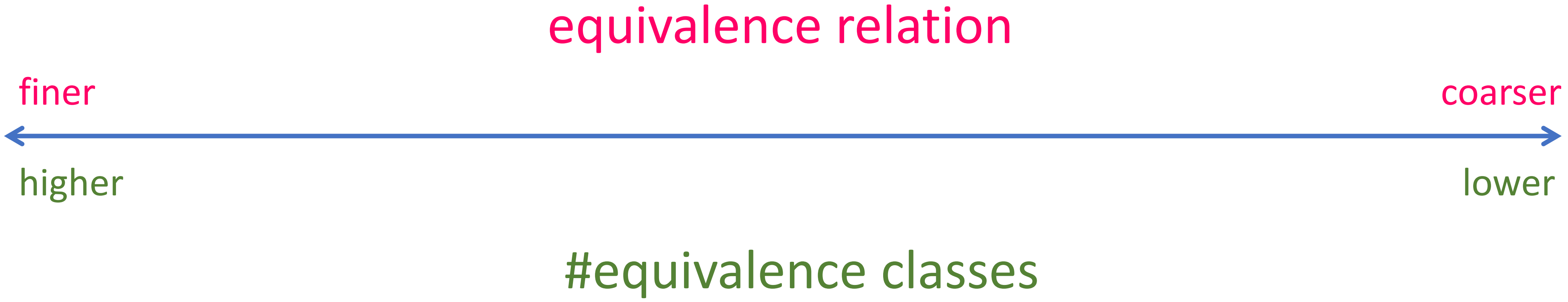


- Classical (*Mazurkiewicz*)
- reads-from
- reads-value-from

All reads read the same value
And they are causally consistent

[Agarwal et al., CAV ‘21]

Existing equivalence relations



View-equivalence

equivalence relation

finer ⟷ coarser

higher ⟷ lower

#equivalence classes

Classical equivalence

Classical modulo observers equivalence

value equivalence

reads-from equivalence

reads-value-from equivalence

View-equivalence

coarsest existing

↓

lowest #equivalence classes

View-equivalence

Exn-1 $\sim$ *Exn-2*

$\{ r_1, r_2, r_3 \}$ $\{ r_1, r_2, r_3 \}$

I. same set of *read events*

View-equivalence

$$Exn\text{-}1 \sim Exn\text{-}2$$

$$\{\ r_1 \rightarrow v_1,\ r_2 \rightarrow v_2,\ r_3 \rightarrow v_3\ \} \qquad \{\ r_1 \rightarrow v_1,\ r_2 \rightarrow v_2,\ r_3 \rightarrow v_3\ \}$$

I. same set of *read events*

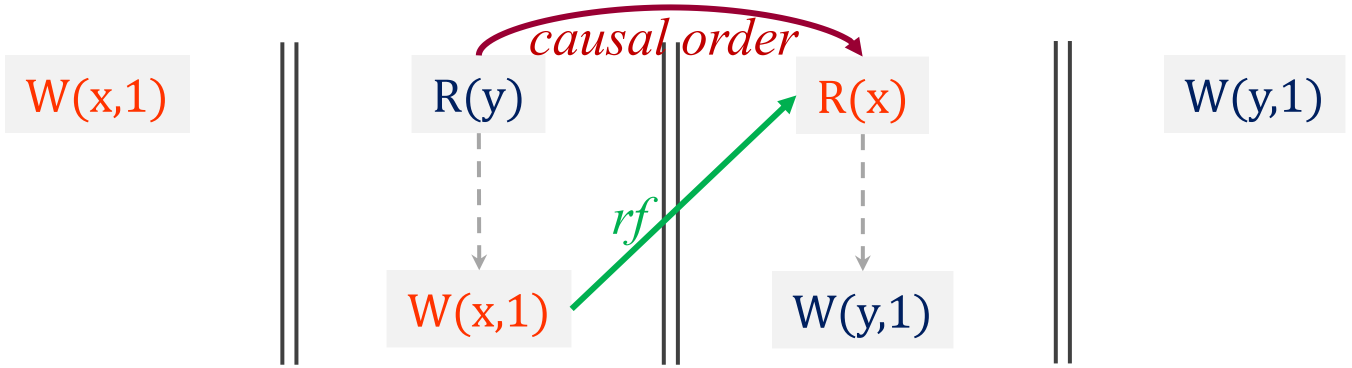
II. Each *read event* reads the same value

View-equivalence vs Existing equivalence relations

- classical (*Mazurkiewicz*) — *reads-from* ordering, *from-reads* ordering, *modification* ordering
- reads-from — *reads-from* ordering
- reads-value-from — *causal-reads* ordering ≤ *reads-from* ordering
- View-equivalence — *NO ordering*

reads-value-from *vs* view-equivalence

Initially x=0, y=0



6

reads-value-from

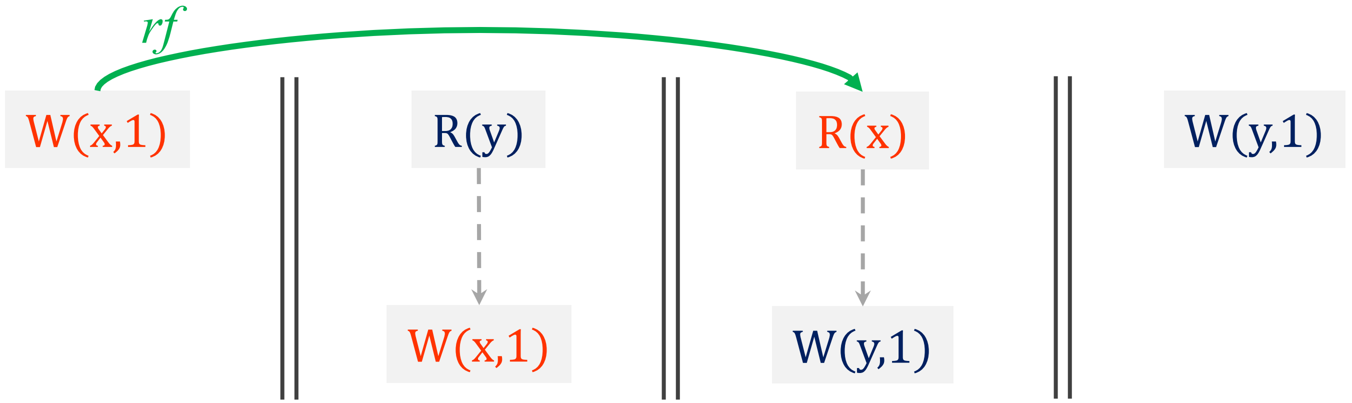
equivalence classes

4

view

equivalence class

I(x,0) I(y,0)



Tradeoff of causal ordering

equivalence relation

finer ⟷ coarser

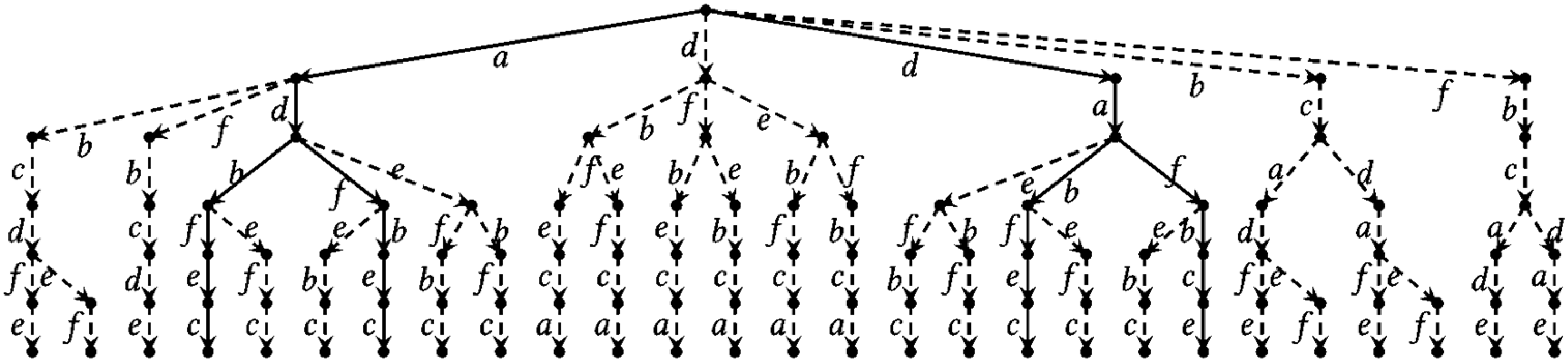
higher ⟷ lower

ordering on events

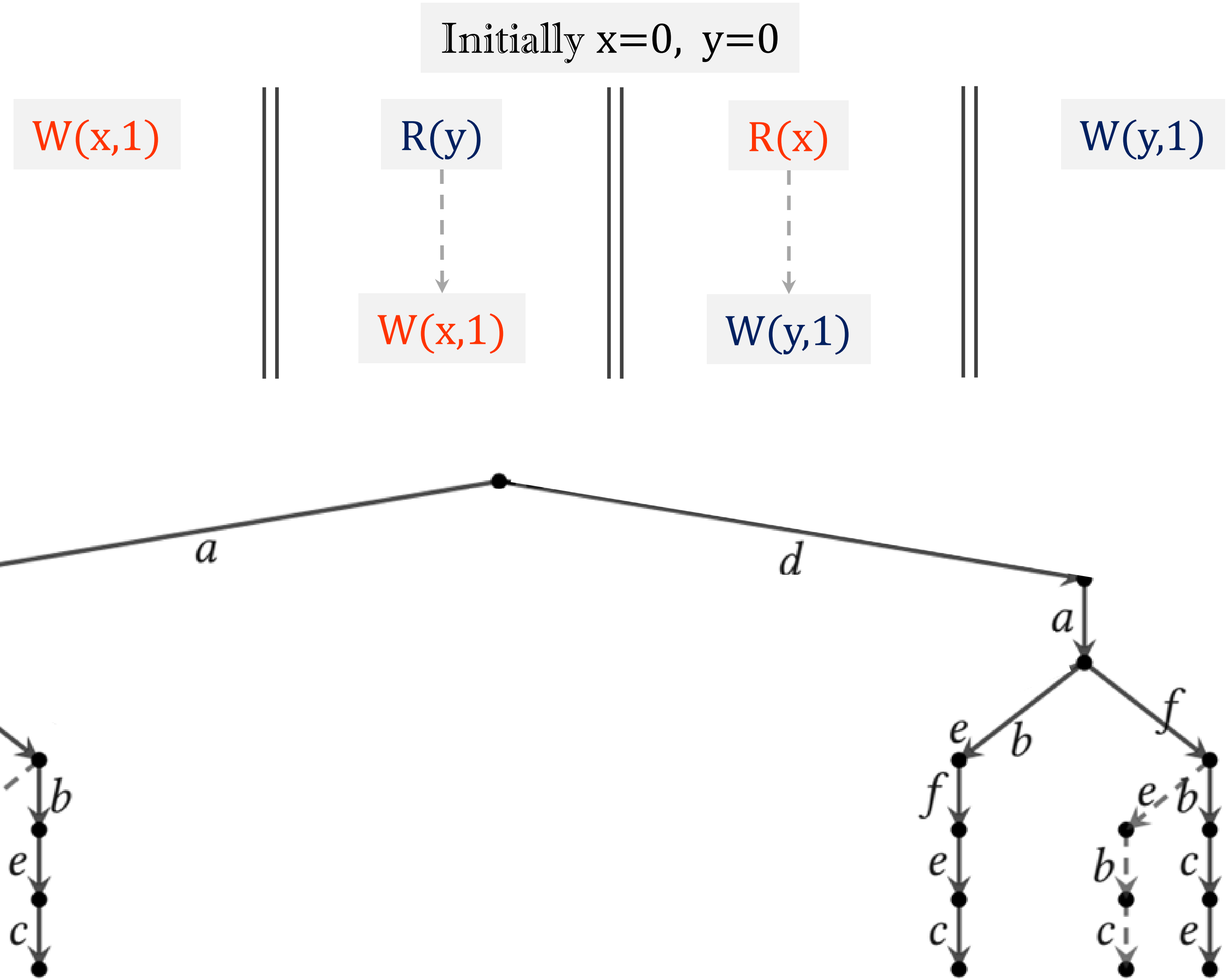
Tradeoff of causality

Initially x=0, y=0

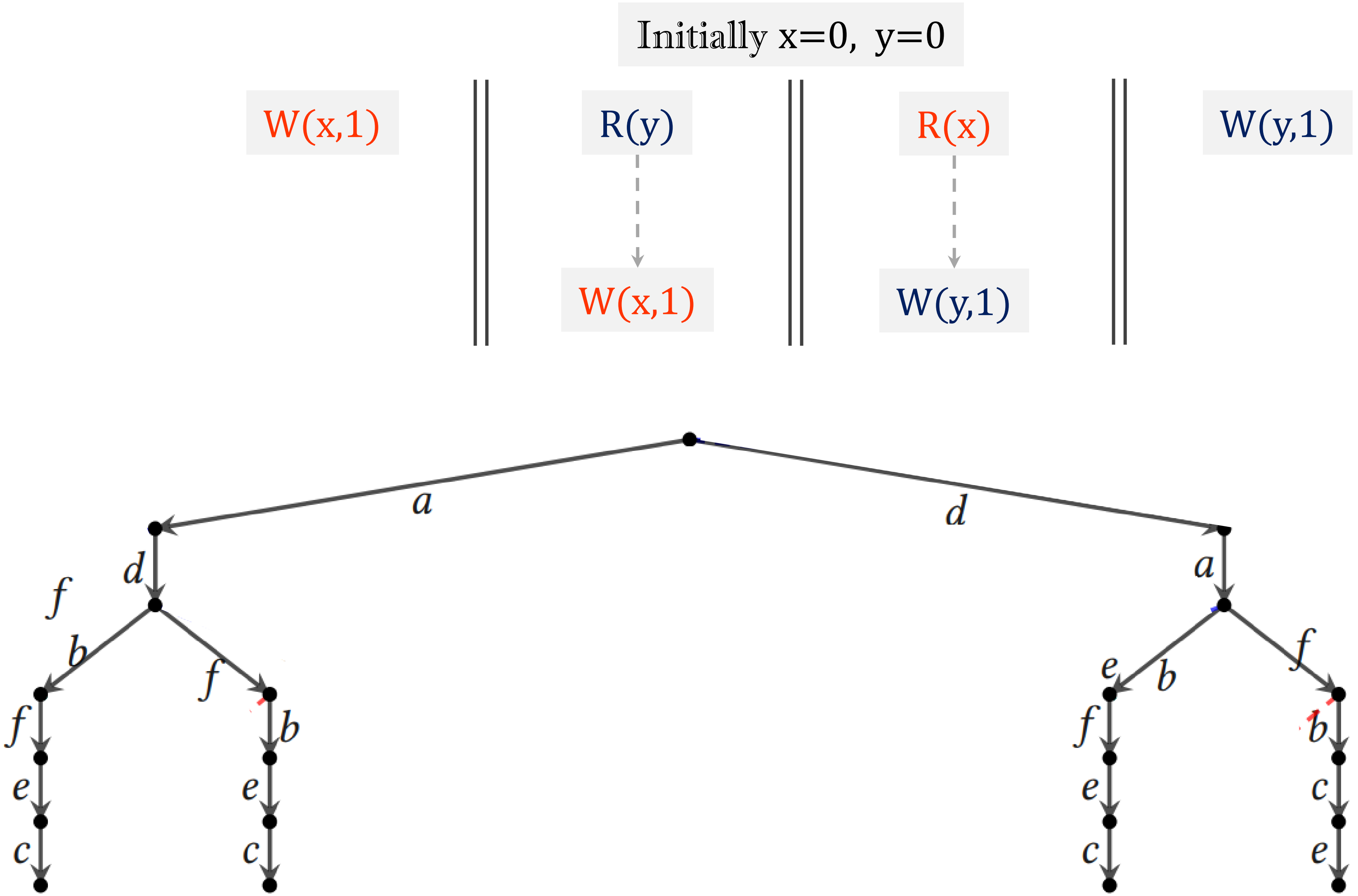
W(x,1)	‖	R(y) → W(x,1)	‖	R(x) → W(y,1)	‖	W(y,1)



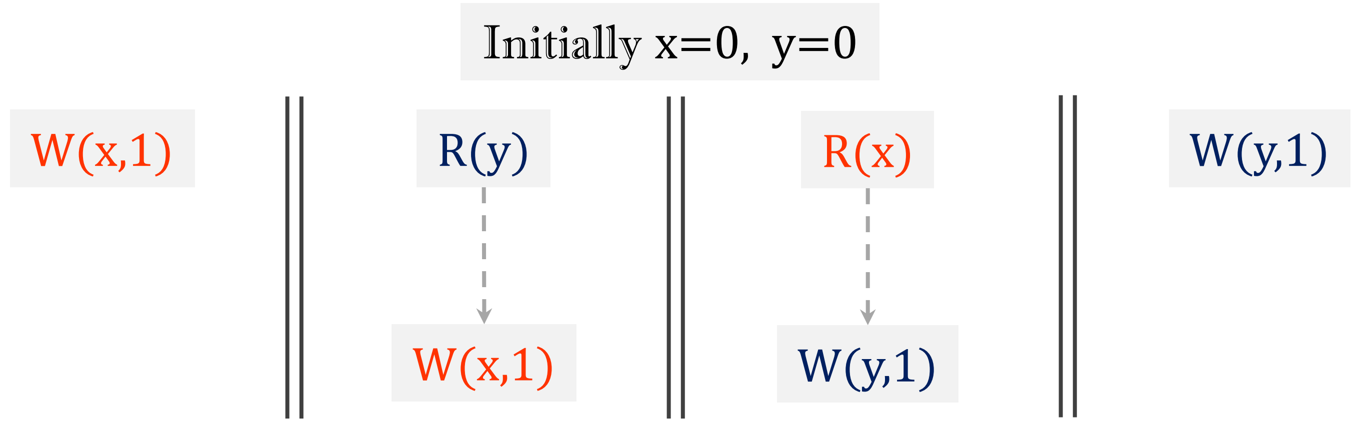
Tradeoff of causality



Tradeoff of causality



Tradeoff of causality



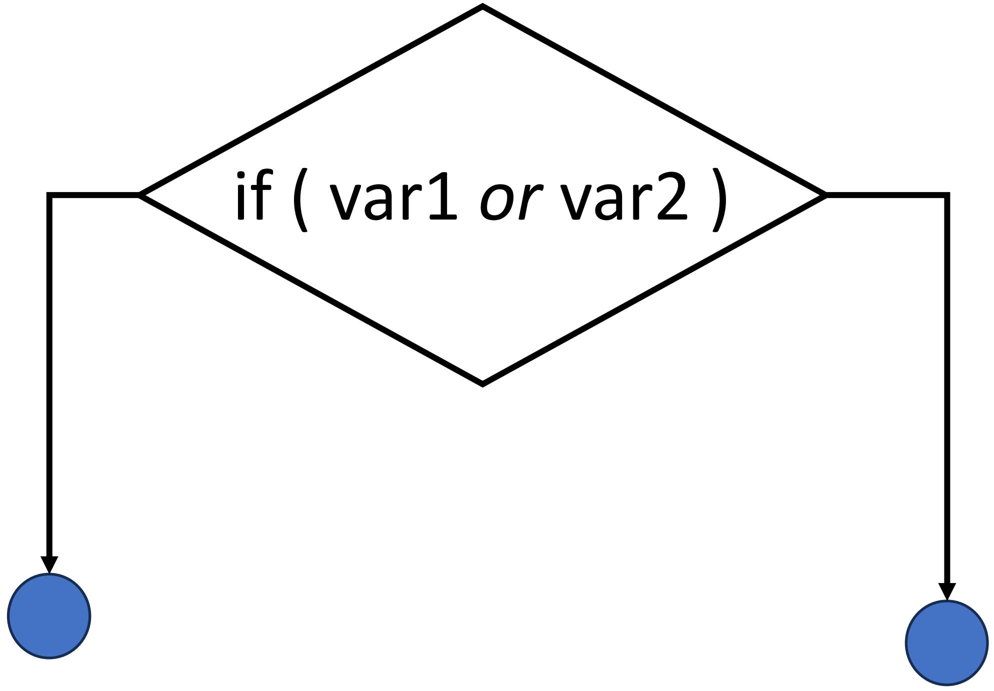
#loops	classical (ODPOR) #Seq	classical (ODPOR) Time	reads-from (RFSC) #Seq	reads-from (RFSC) Time	Reads-value-from (RVF-SMC) #Seq	Reads-value-from (RVF-SMC) Time	View-equivalence (ViEqui) #Seq	View-equivalence (ViEqui) Time
2	1425	0.36	157	0.04	65	0.04	**8**	**0.02**
4	4,931,685	1346.377	99,577	36.32	1,187	0.19	**16**	**0.03**
10	-	Timeout	-	Timeout	3,703,196	705.69	**40**	**0.06**
20	-	Timeout	-	Timeout	-	Timeout	**80**	**0.28**

Timeout: 30mins

Applicable for sound analysis

print (var)

assert (var1 *or* var2)



Applicable for sound analysis

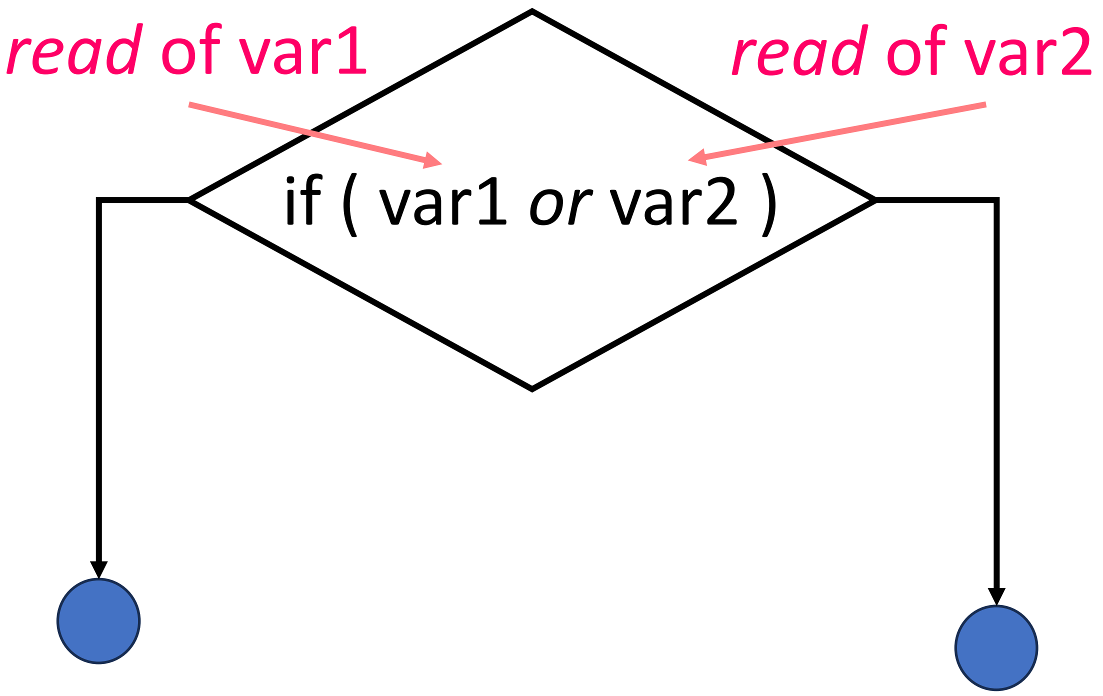
read of var

print (var)

read of var1

assert (var1 *or* var2)

read of var2



Applicable for sound analysis

$\{ v1, v2, \ldots, vn \}$

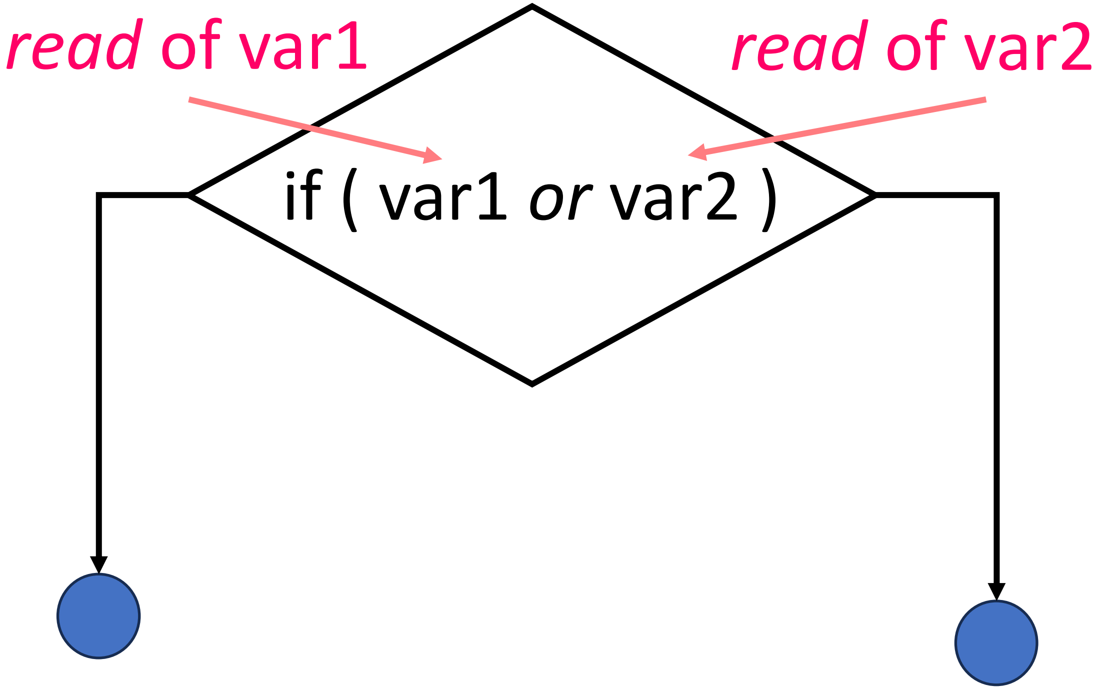
read of var

print (var)

read of var1

assert (var1 *or* var2)

read of var2



Tradeoff of causality

- SMCs discover equivalence classes *on-the-fly*

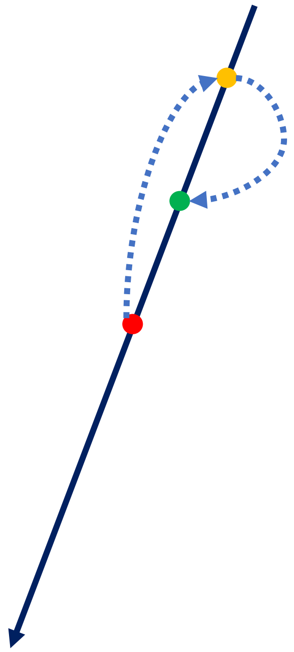
  - explore an execution (eq-1)

Eq-1

Tradeoff of causality

- SMCs discover equivalence classes *on-the-fly*

  - explore an execution (eq-1)
  - manipulate order of events

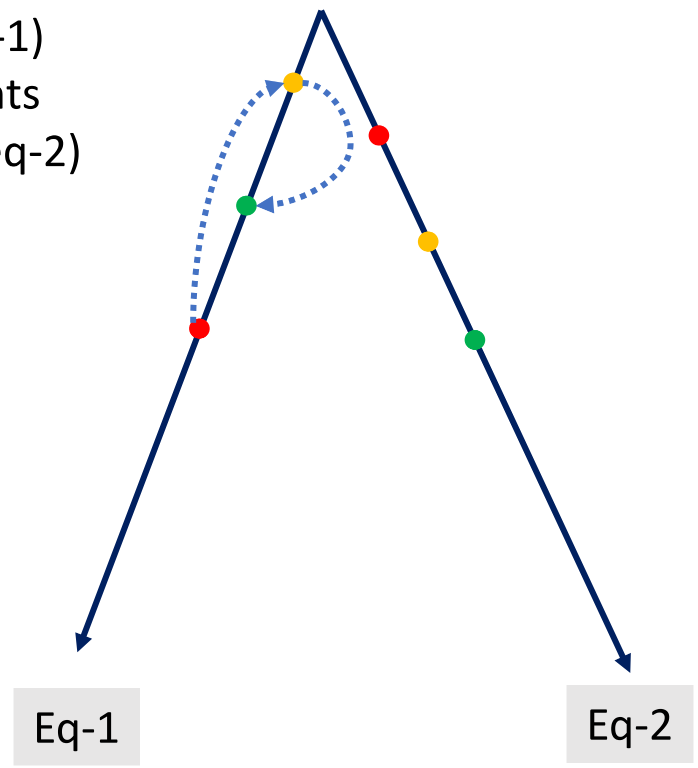


Eq-1

Tradeoff of causality

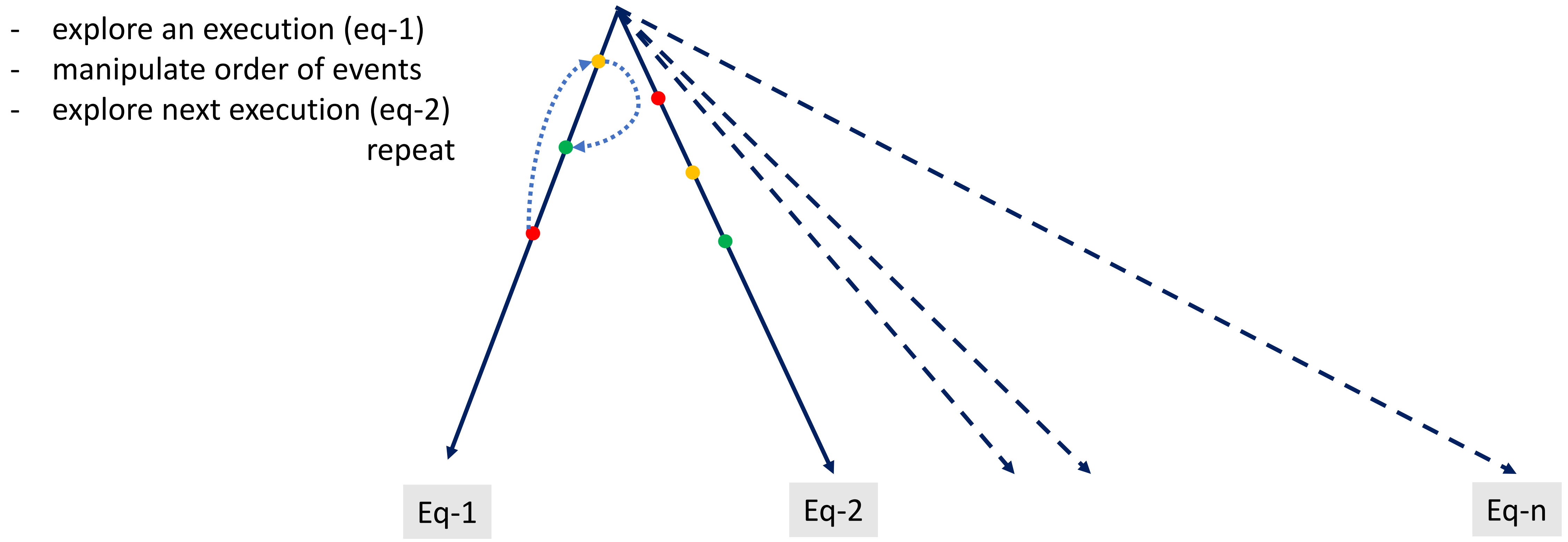
- SMCs discover equivalence classes *on-the-fly*

  - explore an execution (eq-1)
  - manipulate order of events
  - explore next execution (eq-2)

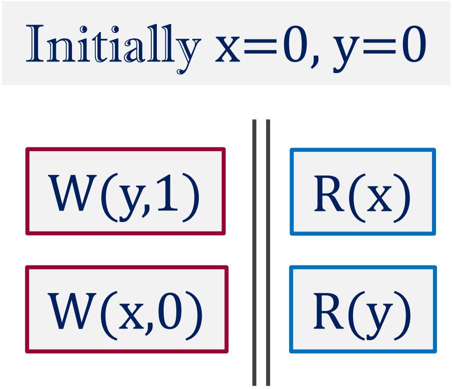


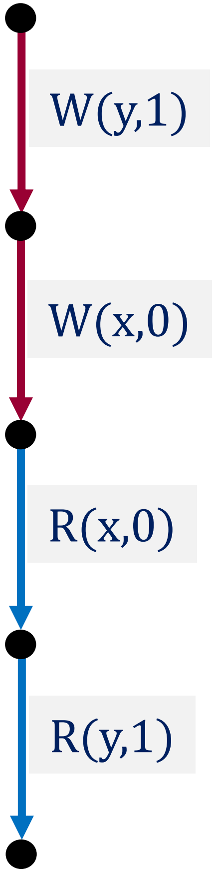
Tradeoff of causality

- SMCs discover equivalence classes *on-the-fly*

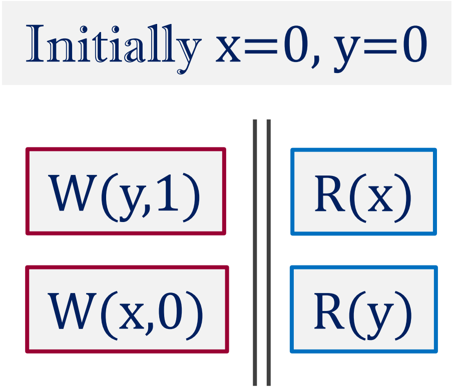


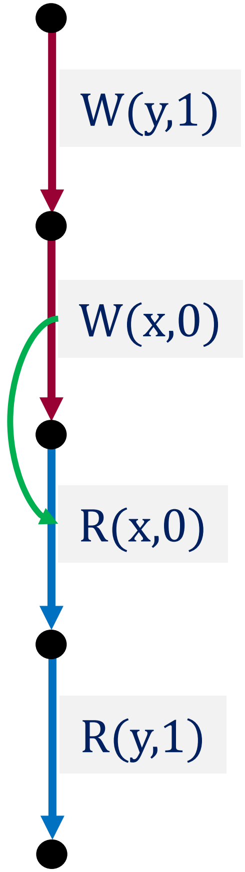
Tradeoff of causality





Tradeoff of causality

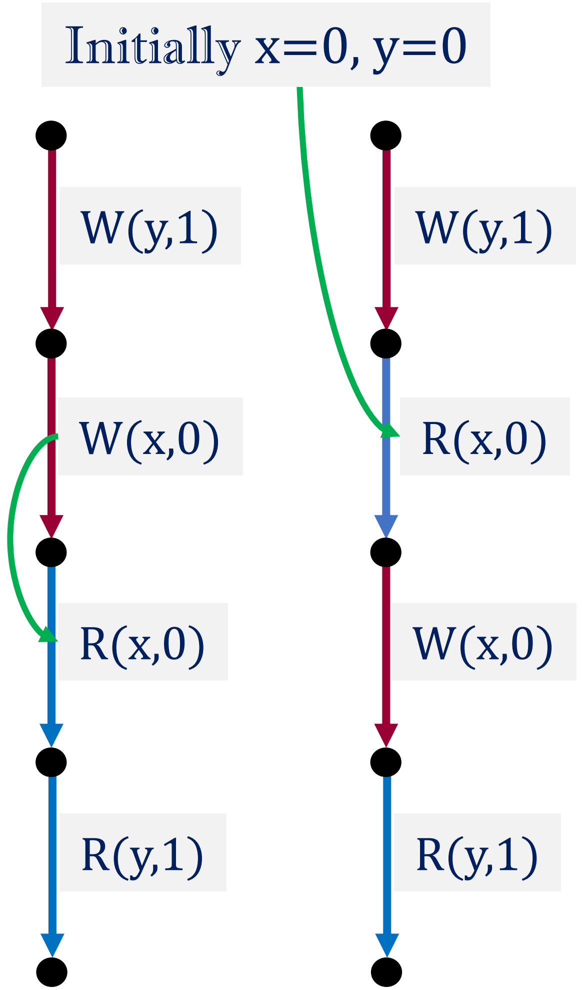




Tradeoff of causality

Initially x=0, y=0

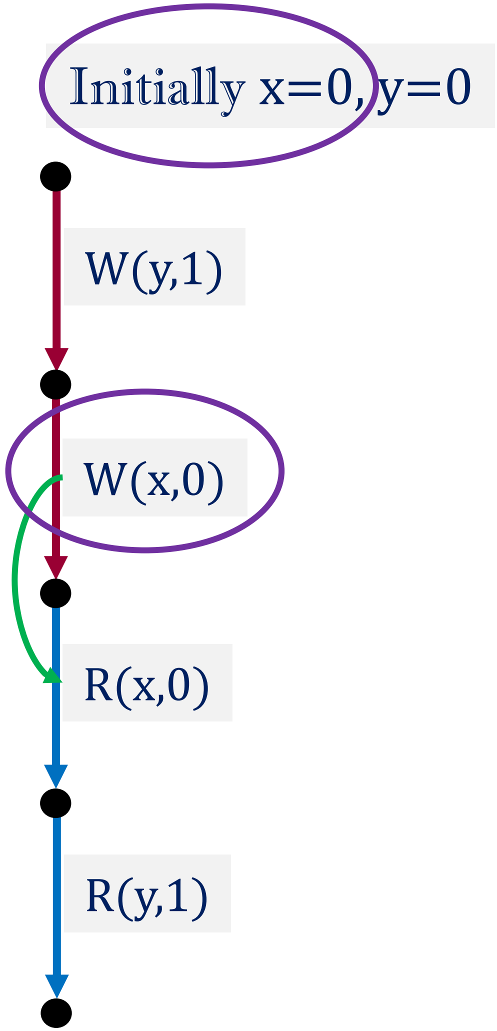
Thread 1		Thread 2
W(y,1)	‖	R(x)
W(x,0)	‖	R(y)



Tradeoff of causality

Initially x=0, y=0

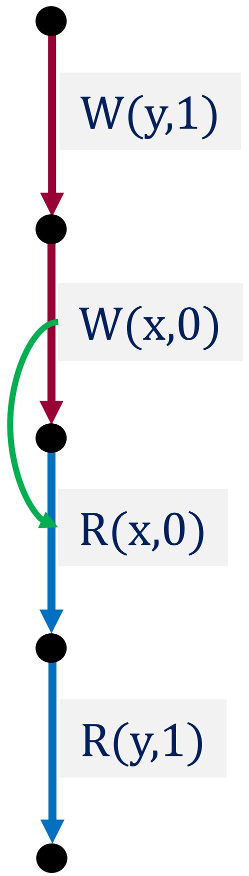
W(y,1)	R(x)
W(x,0)	R(y)



Tradeoff of causality

Initially x=0, y=0

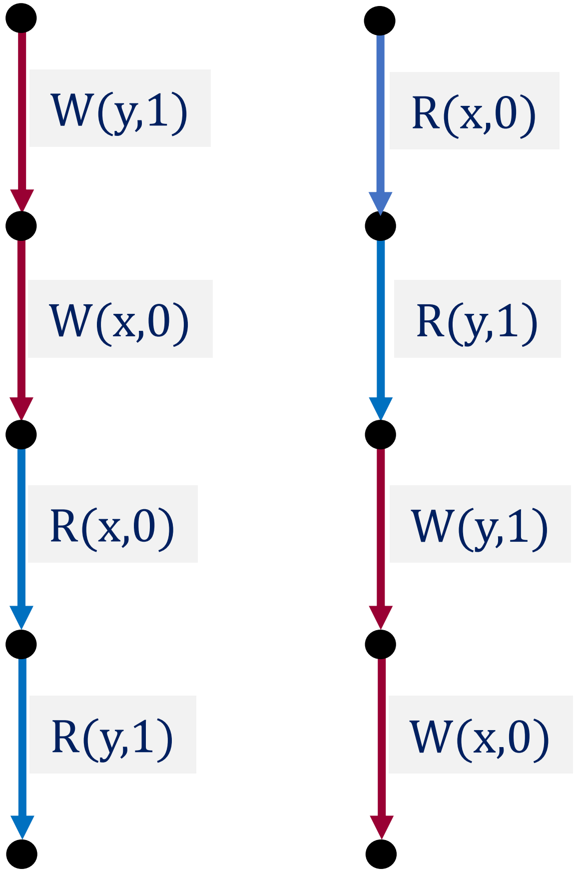
W(y,1)	‖	R(x)	0
W(x,0)	‖	R(y)	1



Tradeoff of causality

Initially x=0, y=0

W(y,1)	R(x)	0
W(x,0)	R(y)	1



Tradeoff of causality

equivalence relation

finer ⟷ coarser

higher ⟷ lower

ordering on events

+ Has necessary information for constructing a coherent ordering

+ can reduce the number of explorations and achieve higher performance
- compute coherence operationally

ViEqui. SMC under view-equivalence

- Deterministic and terminating C/C++ programs
- Single input
- Under sequential consistency

- *Complete*: each maximal sequence represents an equivalence class
- *Sound*: each equivalence class is explored
- *Optimal*: each equivalence class is explored exactly once

- #view-equivalence classes: $|\mathcal{V}|^{|\mathcal{E}^{\mathbb{R}}|}$

ViEqui tool

Implemented over Nidhugg.

available at: https://github.com/nidhugg/nidhugg

Tested over 16,154 litmus tests of concurrent C programs

borrowed from [Abdulla et al., OOPSLA '18]

Performance analysis

ODPOR → classical equivalence [Abdulla et al., POPL '14]
RFSC → reads-from equivalence [Abdulla et al., OOPSLA '19]
RVF-SMC → reads-value-from equivalence [Agarwal et al., CAV '21]

	ODPOR		RFSC		RVF-SMC		ViEqui	
benchmark	#Seq	Time	#Seq	Time	#Seq	Time	#Seq	Time
monabsex(5)	14400	2.56	1296	0.31	6	0.04	**1**	**0.02**
monabsex(100)	-	To	-	To	101	1.20	**1**	**0.09**
monabsex(500)	-	To	-	To	501	195.69	**1**	**2.63**
redundant-co(8)	1969110	338.84	217	0.15	11	0.04	**7**	**0.02**
redundant-co(10)	-	To	331	0.16	11	0.03	**7**	**0.02**
redundant-co(50)	-	To	7651	2.58	11	**0.03**	**7**	0.04
redundant-co(1000)	-	To	-	To	11	**0.16**	**7**	3.24
IBM-incdec(50)	-	To	-	To	-	To	**3**	**7.70**
IBM-incdec(100)	-	To	-	To	-	To	**3**	**34.23**

To: Timeout (30 mins)

Performance analysis

Benchmarks with constant number of view-equivalence classes

	ODPOR		RFSC		RVF-SMC		ViEqui	
benchmark	#Seq	Time	#Seq	Time	#Seq	Time	#Seq	Time
monabsex(5)	14400	2.56	1296	0.31	6	0.04	**1**	**0.02**
monabsex(100)	-	To	-	To	101	1.20	**1**	**0.09**
monabsex(500)	-	To	-	To	501	195.69	**1**	**2.63**
redundant-co(8)	1969110	338.84	217	0.15	11	0.04	**7**	**0.02**
redundant-co(10)	-	To	331	0.16	11	0.03	**7**	**0.02**
redundant-co(50)	-	To	7651	2.58	11	**0.03**	**7**	0.04
redundant-co(1000)	-	To	-	To	11	**0.16**	**7**	3.24
IBM-incdec(50)	-	To	-	To	-	To	**3**	**7.70**
IBM-incdec(100)	-	To	-	To	-	To	**3**	**34.23**

Performance analysis

Benchmarks with many writes but few values and causal dependencies on reads

	ODPOR		RFSC		RVF-SMC		ViEqui		Assert
benchmark	#Seq	Time	#Seq	Time	#Seq	Time	#Seq	Time	violation
nd-array2(4,4)	2616	0.89	292	0.19	534	0.10	**51**	**0.04**	No
nd-array2(6,6)	-	To	75486	21.67	63491	8.34	**2163**	**3.06**	No
nd-array2(14,7)	-	To	1649221	610.68	908984	156.31	**18731**	**120.74**	No
nd-array1(100,100)	**1**	0.22	**1**	12.87	**1**	0.06	**1**	0.19	Yes
nd-array1(1000,500)	**1**	**0.03**	**1**	0.15	**1**	**0.03**	**1**	0.08	Yes

Performance analysis

Benchmarks with writes of different values and no causal dependencies on reads

benchmark	ODPOR #Seq	ODPOR Time	RFSC #Seq	RFSC Time	RVF-SMC #Seq	RVF-SMC Time	ViEqui #Seq	ViEqui Time
swsc-co1(20)	-	To	8040	14.80	8060	17.06	**7240**	**5.71**
swsc-co1(50)	-	To	125100	860.71	125150	1769.71	**120100**	**375.43**
swsc-co1(60)	-	To	-	To	-	To	**208920**	**891.21**
swsc-co10(10)	-	To	**10**	0.04	11	0.04	**10**	**0.02**
swsc-co10(100)	-	To	**100**	2.19	101	7.69	**100**	**0.76**
swsc-co10(250)	-	To	**250**	41.89	251	266.39	**250**	**9.07**
alpha2(10)	-	To	**111**	0.14	123	0.14	**111**	**0.08**
alpha2(100)	-	To	**10101**	218.71	10203	774.59	**10101**	**191.57**
alpha2(150)	-	To	**22651**	1161.76	-	To	**22651**	**1076.26**

Performance analysis

Mutual exclusion benchmarks from SV-Comp [Beyer 2021]

	ODPOR		RFSC		RVF-SMC		ViEqui	
benchmark	#Seq	Time	#Seq	Time	#Seq	Time	#Seq	Time
burns(5)	2353602	1046.92	-	To	17382	6.38	**36**	**0.05**
burns(10)	-	To	-	To	-	To	**121**	**0.32**
burns(40)	-	To	-	To	-	To	**1681**	**150.36**
burns(60)	-	To	-	To	-	To	**3721**	**1060.96**
dekker(10)	739021	420.96	739021	927.133	2713870	865.98	**21**	**0.04**
dekker(100)	-	To	-	To	-	To	**201**	**31.03**
dekker(150)	-	To	-	To	-	To	**301**	**225.01**
dekker(200)	-	To	-	To	-	To	**401**	**1064.42**
peterson(5)	2782162	1432.44	-	To	-	To	**31**	**0.04**
peterson(50)	-	To	-	To	-	To	**301**	**16.26**
peterson(100)	-	To	-	To	-	To	**601**	**385.10**
peterson(120)	-	To	-	To	-	To	**721**	**985.75**
szymanski(4)	396583	198.87	396583	378.50	1444246	419.67	**5335**	**5.15**
szymanski(5)	-	To	-	To	-	To	**19349**	**26.73**
szymanski(7)	-	To	-	To	-	To	**264209**	**674.53**

Performance analysis

Benchmarks with same classes under view-equivalence and classical equivalence

	ODPOR		RFSC		RVF-SMC		ViEqui	
benchmark	#Seq	Time	#Seq	Time	#Seq	Time	#Seq	Time
pgsql(5,5)	**781**	**0.70**	**781**	1.15	19900	3.97	**781**	0.75
pgsql(6,7)	**55987**	**68.57**	**55987**	123.53	2292077	821.00	**55987**	186.18
pgsql(7,7)	**137257**	**171.45**	**137257**	316.25	-	To	**137257**	909.43
unverif(5,5)	**14400**	**2.74**	**14400**	5.01	68890	14.79	**14400**	227.78
unverif(5,10)	**14400**	**2.98**	**14400**	5.24	70890	16.12	**14400**	230.81
unverif(6,5)	**518400**	**110.60**	**518400**	206.56	2625944	818.55	-	To

Future Scope

- view-equivalence based SMC for *weak memory models*
- *coarsening* by considering the assert condition in the equivalence relation
- applicability for database *transactions*
- Richer constructs like *coarse grained synchronization*

Future Scope

- *scalability*

benchmark	ODPOR #Seq	ODPOR Time	rfsc #Seq	rfsc Time	RVF-SMC #Seq	RVF-SMC Time	ViEqui #Seq	ViEqui Time
FreeBSD-abd-kbd	1	0.03	1	0.12	1	0.02	1	0.02
FreeBSD-rdma-addr	1	0.02	1	0.12	1	0.01	1	0.02
NetBSD-sysmon-power	4	0.03	26	0.15	6	0.02	6	0.04
Solaris-space-map	2	0.03	2	0.12	1	0.02	1	0.01

Future Scope

- *scalability*

benchmark	ODPOR #Seq	ODPOR Time	rfsc #Seq	rfsc Time	RVF-SMC #Seq	RVF-SMC Time	ViEqui #Seq	ViEqui Time
FreeBSD-abd-kbd	1	0.03	1	0.12	1	0.02	1	0.02
FreeBSD-rdma-addr	1	0.02	1	0.12	1	0.01	1	0.02
NetBSD-sysmon-power	4	0.03	26	0.15	6	0.02	6	0.04
Solaris-space-map	2	0.03	2	0.12	1	0.02	1	0.01
Safestack		oom		oom		To		To

oom: out of memory

Thank You

Questions?