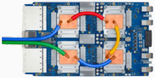
How Hard is Weak Memory Testing?

S. Chakraborty, S. Krishna, U. Mathur and **Andreas Pavlogiannis**

TU Delft
INDIAN INSTITUTE OF TECHNOLOGY BOMBAY
NUS National University of Singapore
AARHUS UNIVERSITY

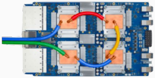
Concurrency is Everywhere

Concurrency is a ubiquitous computing paradigm



Concurrency is Everywhere

Concurrency is a ubiquitous computing paradigm

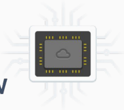


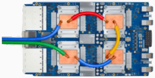
Verification:

- Is my program correct in a given concurrent setting?
- What behaviors are possible in a concurrent environment?

Concurrency is Everywhere

Concurrency is a ubiquitous computing paradigm





Verification:

- Is my program correct in a given concurrent setting?
- What behaviors are possible in a concurrent environment?

Not all concurrency is the same

Store Buffer under x86-TSO

Store Buffer

$$x = 0, y = 0$$

$\texttt{w}(y, 1);$	$\vert\vert$	$\texttt{w}(x, 1);$
$a := \texttt{r}(x, 0)$		$b := \texttt{r}(y, 0)$

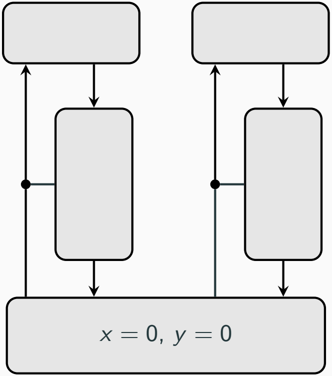
Not sequentially consistent

Store Buffer under x86-TSO

Store Buffer

$$x = 0, y = 0$$

$\texttt{w}(y, 1);$	$\|\|$	$\texttt{w}(x, 1);$
$a := \texttt{r}(x, 0)$		$b := \texttt{r}(y, 0)$

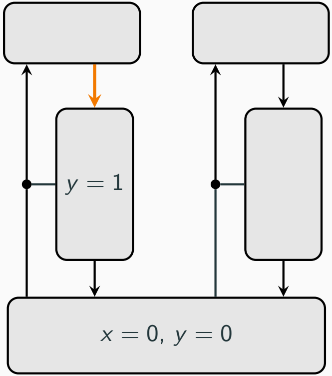


Store Buffer under x86-TSO

Store Buffer

$x = 0, y = 0$

$\mathtt{w}(y, 1)$; ←	‖	$\mathtt{w}(x, 1)$;
$a := \mathtt{r}(x, 0)$	‖	$b := \mathtt{r}(y, 0)$

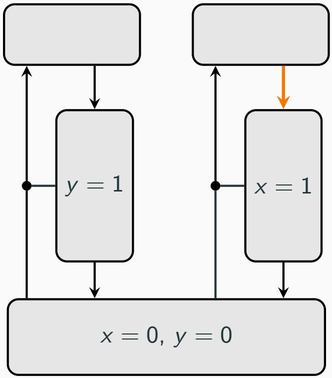


Store Buffer under x86-TSO

Store Buffer

$$x = 0, y = 0$$

$\mathtt{w}(y, 1);$	$\|\|$	$\mathtt{w}(x, 1);$ ←
$a := \mathtt{r}(x, 0)$		$b := \mathtt{r}(y, 0)$

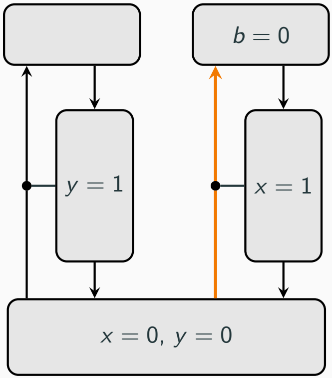


Store Buffer under x86-TSO

Store Buffer

$$x = 0, y = 0$$

$\texttt{w}(y, 1);$	$\|\|$	$\texttt{w}(x, 1);$
$a := \texttt{r}(x, 0)$		$b := \texttt{r}(y, 0)$ ←

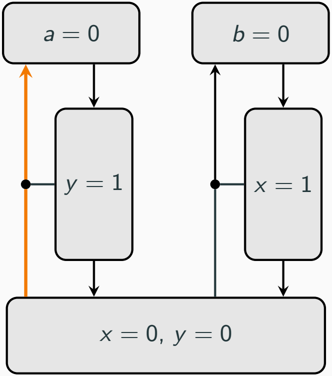


Store Buffer under x86-TSO

Store Buffer

$$x = 0, y = 0$$

$\mathtt{w}(y, 1);$	$\|\|$	$\mathtt{w}(x, 1);$
$a := \mathtt{r}(x, 0) \leftarrow$		$b := \mathtt{r}(y, 0)$

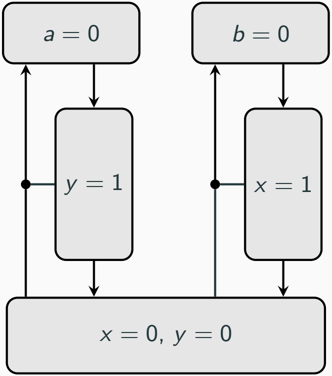


Store Buffer under x86-TSO

Store Buffer

$x = 0, y = 0$

$\mathtt{w}(y, 1);$ $a := \mathtt{r}(x, 0)$	$\parallel$	$\mathtt{w}(x, 1);$ $b := \mathtt{r}(y, 0)$



- **Behavior possible under x86-TSO**

Memory Models

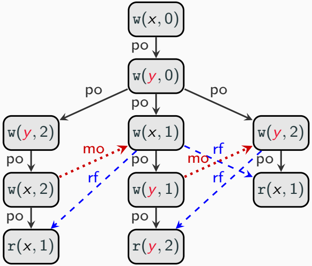
- Formal models of concurrency
- Specifications of all possible communication patterns
  - Buffers/caching
  - Out-of-order execution
  - Speculation
  - Cache coherence protocols
  - Compiler optimizations
  - Message delays
  - ...
- In all cases: **weak data consistency**

Memory Models

SC	Standard Sequential Consistency
TSO	x86-Total Store Order
PSO	Sparc-Partial Store Order
RA	The release-acquire semantics of C11
Relaxed	The relaxed fragment of C11
Relaxed-Acyclic	Relaxed + $(\mathsf{po} \cup \mathsf{rf})$-acyclicity
CC	Causal consistency
CCv	Causal convergence
CM	Causal memory

Execution Graphs

Program executions are represented as execution graphs



Executions on Weak Memory

An **execution** is a tuple $X = (E, po, rf, mo)$, where:

- E is a set of events
- po is the *program order* over E, total on each thread
- rf is a *reads-from* relation on $W \times R$
  - $(\mathtt{w}, \mathtt{r}) \in rf$ means $\mathtt{r}$ reads the value written by $\mathtt{w}$
- mo_x is a total *modification order* over all writes $\mathtt{w}(x)$
  - $mo = \bigcup_x mo_x$

Consistent Executions

A memory model $\mathcal{M}$ defines a set of axioms that every execution must satisfy

Consistency

If an execution $\mathsf{X} = (\mathsf{E}, \mathsf{po}, \mathsf{rf}, \mathsf{mo})$ satisfies all axioms of $\mathcal{M}$, we say that X is **consistent** in $\mathcal{M}$, written as $\mathsf{X} \vDash \mathcal{M}$.

Consistent Executions

A memory model $\mathcal{M}$ defines a set of axioms that every execution must satisfy

Consistency

If an execution $\mathsf{X} = (\mathsf{E}, \mathsf{po}, \mathsf{rf}, \mathsf{mo})$ satisfies all axioms of $\mathcal{M}$, we say that X is **consistent** in $\mathcal{M}$, written as $\mathsf{X} \vDash \mathcal{M}$.

Memory models may be ordered in terms of the behaviors they allow, i.e., the executions they admit

Weak(er) Memory Models

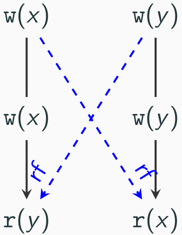
Given two memory models $\mathcal{M}_1$, $\mathcal{M}_2$, we say that $\mathcal{M}_2$ is **weaker than** $\mathcal{M}_1$, written $\mathcal{M}_1 \sqsubseteq \mathcal{M}_2$, if for every execution X, we have

$$\mathsf{X} \vDash \mathcal{M}_1 \Rightarrow \mathsf{X} \vDash \mathcal{M}_2$$

$$\text{Eg, } \mathsf{SC} \sqsubseteq \mathsf{TSO} \sqsubseteq \mathsf{RA} \sqsubseteq \{\mathsf{CC}, \mathsf{Relaxed}\}$$

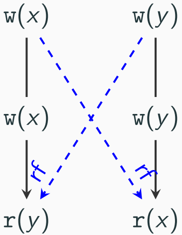
Examples of Consistency

Examples of Consistency

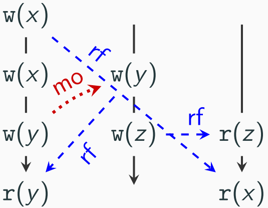


(a) Total Store Order (TSO)

Examples of Consistency

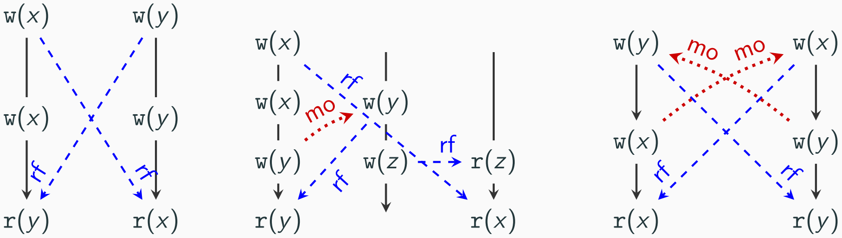


(a) Total Store Order (TSO)



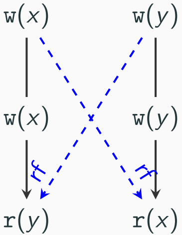
(b) Causal Convergence (CCv)

Examples of Consistency

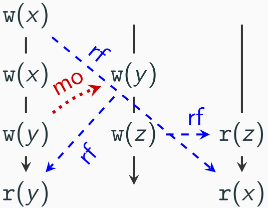


(a) Total Store Order (TSO) **(b)** Causal Convergence (CCv) **(c)** Release/Acquire (RA)

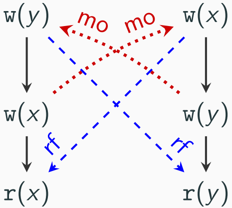
Examples of Consistency



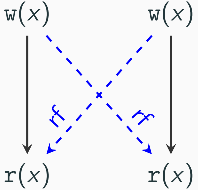
(a) Total Store Order (TSO)



(b) Causal Convergence (CCv)

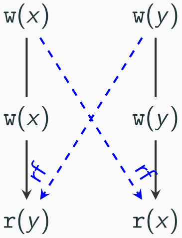


(c) Release/Acquire (RA)

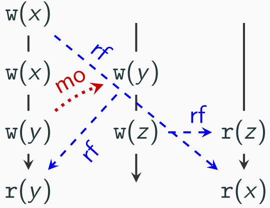


(d) Causal Memory (CM)

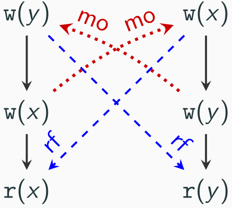
Examples of Consistency



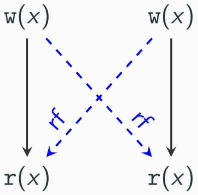
(a) Total Store Order (TSO)



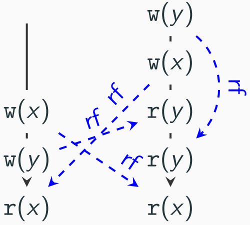
(b) Causal Convergence (CCv)



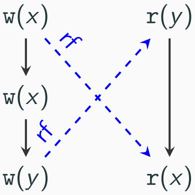
(c) Release/Acquire (RA)



(d) Causal Memory (CM)



(e) Causal Consistency (CC)

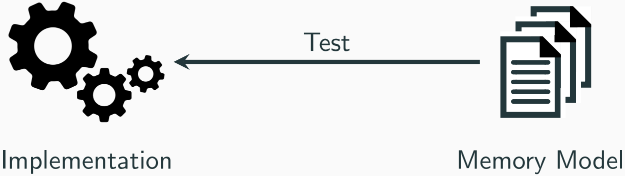


(f) Relaxed (Relaxed)

Testing Weak Memories

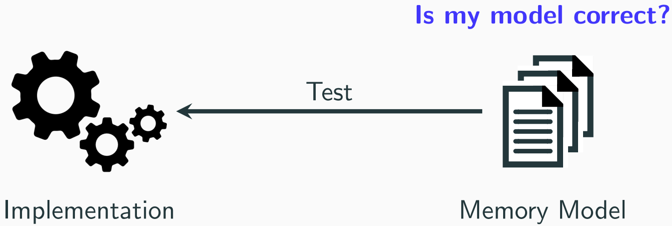
- Aligning a model to an implementation is hard, litmus tests

Is my implementation correct?



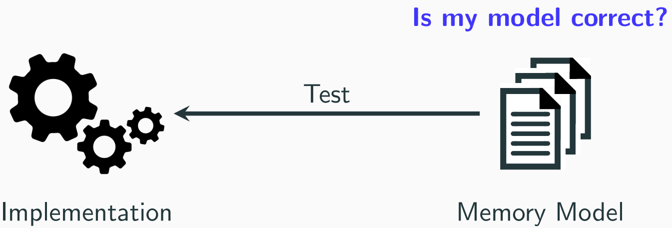
Testing Weak Memories

- Aligning a model to an implementation is hard, litmus tests



Testing Weak Memories

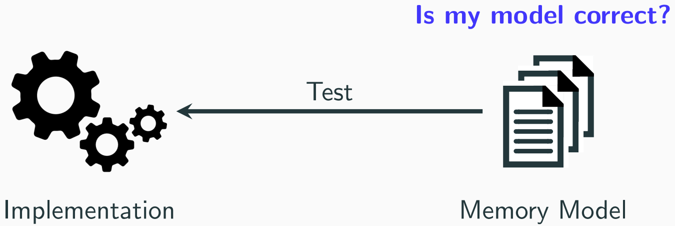
- Aligning a model to an implementation is hard, litmus tests



- Also, model checking, dynamic analyses

Testing Weak Memories

- Aligning a model to an implementation is hard, litmus tests



- Also, model checking, dynamic analyses

- Is the **observed behavior** of the program in alignment with the model?
- Observed behavior is thread-local
  - No rf, no mo

Weak Memory Testing, Formally

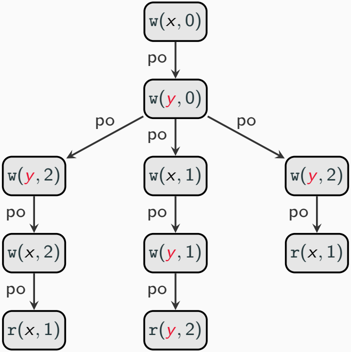
The Testing Problem

Given an abstract execution $\overline{\mathsf{X}} = (\mathsf{E}, \mathsf{po})$ and a memory model $\mathcal{M}$ is there a reads from relation rf and a modification order mo such that $\mathsf{X} = (\mathsf{E}, \mathsf{po}, \mathsf{rf}, \mathsf{mo}) \vDash \mathcal{M}$?

Weak Memory Testing, Formally

The Testing Problem

Given an abstract execution $\overline{\mathsf{X}} = (\mathsf{E}, \mathsf{po})$ and a memory model $\mathcal{M}$ is there a reads from relation rf and a modification order mo such that $\mathsf{X} = (\mathsf{E}, \mathsf{po}, \mathsf{rf}, \mathsf{mo}) \vDash \mathcal{M}$?

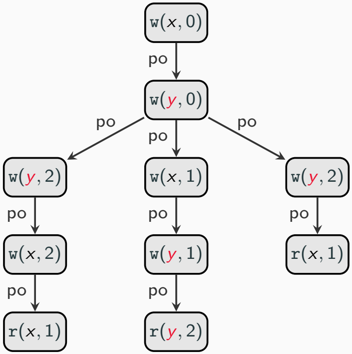


Input $\overline{\mathsf{X}}$

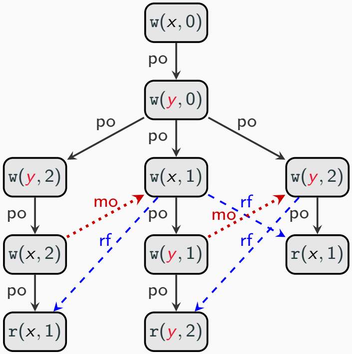
Weak Memory Testing, Formally

The Testing Problem

Given an abstract execution $\overline{X} = (E, po)$ and a memory model $\mathcal{M}$ is there a reads from relation rf and a modification order mo such that $X = (E, po, rf, mo) \models \mathcal{M}$?



Input $\overline{X}$



Output X

How Fast can we Test?

n events, k threads, d memory locations

☹

NP-complete for $k = 3$
NP-complete for $d = 1$

← Sequential Consistency →

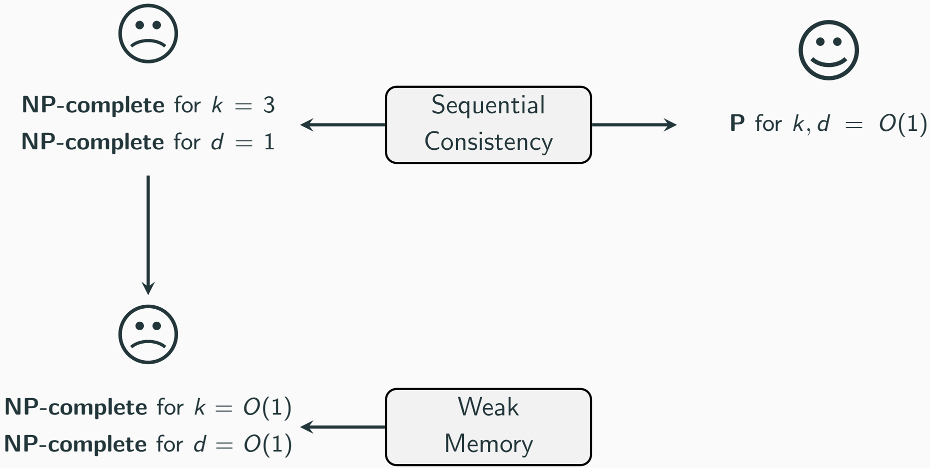
☺

P for $k, d = O(1)$

Weak Memory

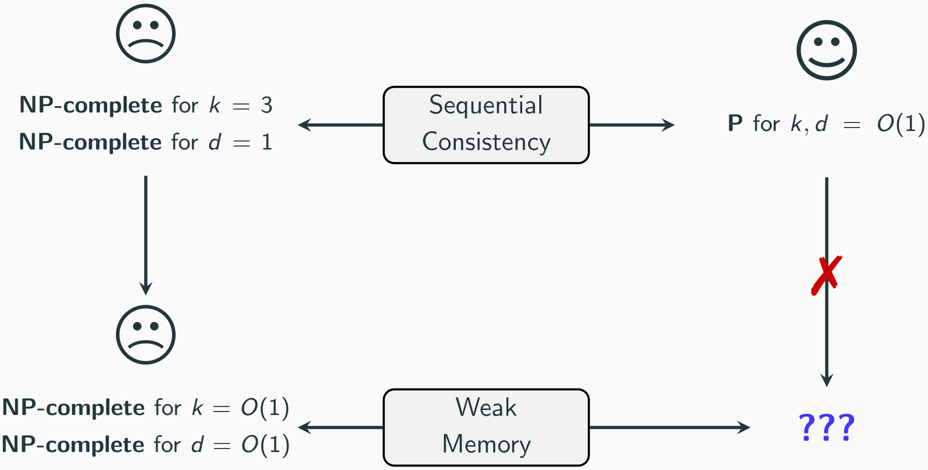
How Fast can we Test?

n events, k threads, d memory locations



How Fast can we Test?

n events, k threads, d memory locations



The Hardness of Weak Memory Testing

Theorem (Hardness of bounded testing)

Testing is **NP-hard** *for any memory model among*

- CCv
- RA
- CM
- CC
- Relaxed-Acyclic

even for abstract executions with bounded

- *threads*
- *memory locations*
- *values read/written*

The Hardness of Weak Memory Testing

Theorem (Hardness of bounded testing)

Testing is **NP-hard** *for any memory model among*

- CCv
- RA
- CM
- CC
- Relaxed-Acyclic

even for abstract executions with bounded

- *threads*
- *memory locations*
- *values read/written*

Non parameterizable!

The Proof is a Bit Involved . . .

Reduction from monotone 1-in-3 SAT $\phi = \mathcal{C}_1 \wedge \mathcal{C}_2 \wedge \cdots \wedge \mathcal{C}_m$

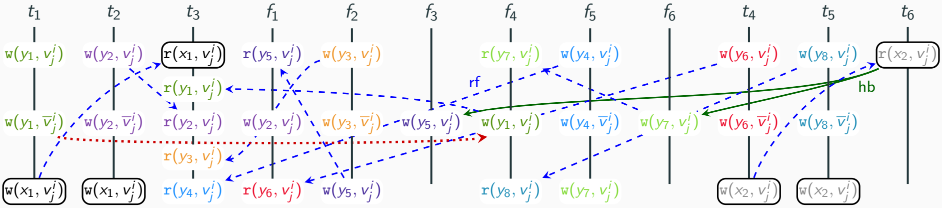
- No negations: $\mathcal{C}_i = x_i^1 \vee x_i^2 \vee x_i^3$
- For each clause, exactly one variable must be true

The Proof is a Bit Involved . . .

Reduction from monotone 1-in-3 SAT $\phi = \mathcal{C}_1 \wedge \mathcal{C}_2 \wedge \cdots \wedge \mathcal{C}_m$

- No negations: $\mathcal{C}_i = x_i^1 \vee x_i^2 \vee x_i^3$
- For each clause, exactly one variable must be true

A Copy Gadget:



How General is this Hardness?

- Hardness proofs are difficult
- Do they generalize?

How General is this Hardness?

- Hardness proofs are difficult
- Do they generalize?

Theorem

For **any** *memory model* $\mathcal{M}$ *with*

- CCv $\sqsubseteq \mathcal{M} \sqsubseteq$ CC, *or*
- CM $\sqsubseteq \mathcal{M} \sqsubseteq$ CC

testing bounded executions is **NP-hard**.

How General is this Hardness?

- Hardness proofs are difficult
- Do they generalize?

Theorem

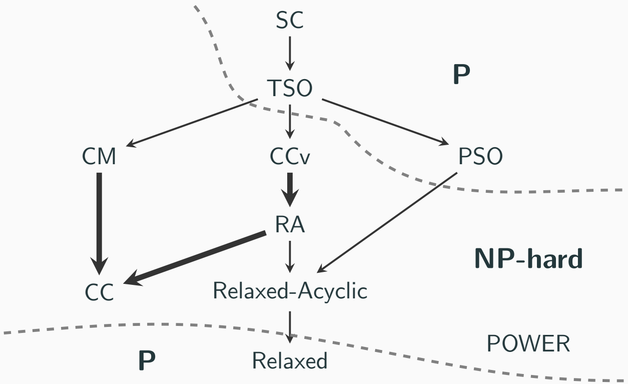
For **any** *memory model* $\mathcal{M}$ *with*

- CCv $\sqsubseteq \mathcal{M} \sqsubseteq$ CC, *or*
- CM $\sqsubseteq \mathcal{M} \sqsubseteq$ CC

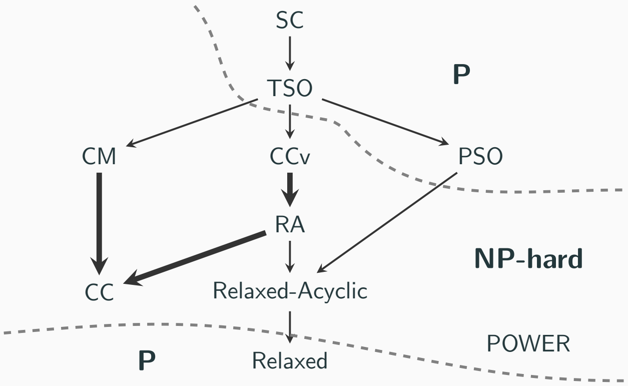
testing bounded executions is **NP-hard**.

However! Bounded testing is in **P** for some weak memory

How Hard is Weak Memory Testing? Very Hard

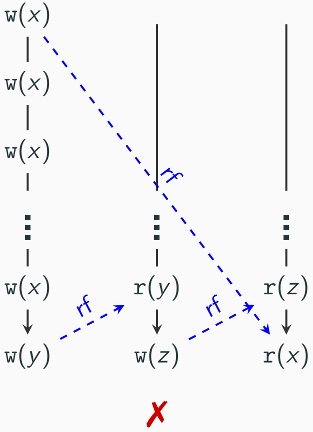


How Hard is Weak Memory Testing? Very Hard

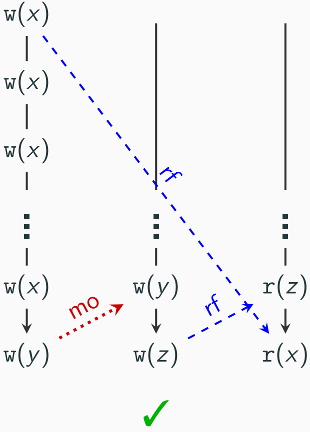


Thank you!
Questions?

Why Should this be Hard?

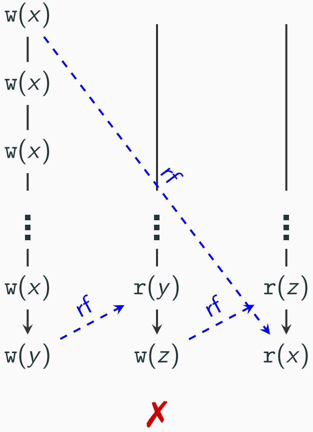


Not Causally Consistent

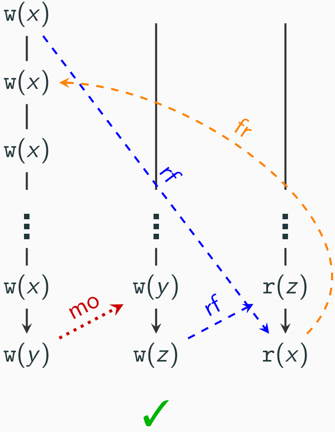


Causally Consistent

Why Should this be Hard?



Not Causally Consistent



Causally Consistent; but not SC

Reads-From (RF) Testing

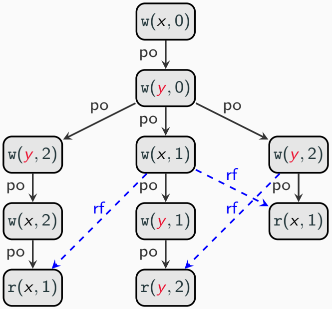
RF-Testing

Given an abstract execution $\overline{\mathsf{X}} = (\mathsf{E}, \mathsf{po}, \mathsf{rf})$ and a memory model $\mathcal{M}$ is there a modification order mo such that $\mathsf{X} = (\mathsf{E}, \mathsf{po}, \mathsf{rf}, \mathsf{mo}) \models \mathcal{M}$?

Reads-From (RF) Testing

RF-Testing

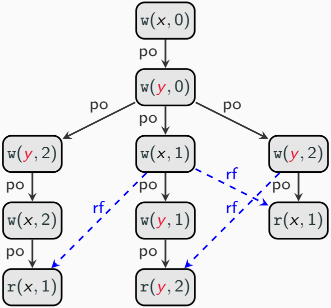
Given an abstract execution $\overline{\mathsf{X}} = (\mathsf{E}, \mathsf{po}, \mathsf{rf})$ and a memory model $\mathcal{M}$ is there a modification order mo such that $\mathsf{X} = (\mathsf{E}, \mathsf{po}, \mathsf{rf}, \mathsf{mo}) \models \mathcal{M}$?

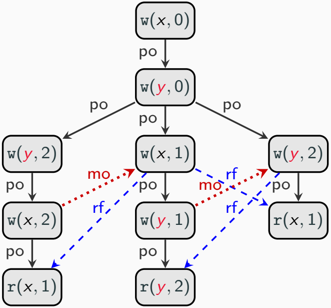


Reads-From (RF) Testing

RF-Testing

Given an abstract execution $\overline{\mathsf{X}} = (\mathsf{E}, \mathsf{po}, \mathsf{rf})$ and a memory model $\mathcal{M}$ is there a modification order mo such that $\mathsf{X} = (\mathsf{E}, \mathsf{po}, \mathsf{rf}, \mathsf{mo}) \models \mathcal{M}$?





How Hard is RF-Testing?

	RF-Testing	**Testing**
SC	**NP-complete**	**P** for $k, d = O(1)$
TSO	**NP-complete**	**P** for $k, d = O(1)$
PSO	**NP-complete**	**P** for $k, d = O(1)$
RA	$O(n \cdot k)$	**NP-complete** for $k, d = O(1)$
CC	$O(n \cdot k)$	**NP-complete** for $k, d = O(1)$
CCv	$O(n \cdot k)$	**NP-complete** for $k, d = O(1)$
CM	$O(n \cdot k)$	**NP-complete** for $k, d = O(1)$
Relaxed-Acyclic	$O(n)$	**NP-complete** for $k, d = O(1)$

How Hard is RF-Testing?

	RF-Testing	Testing
SC	**NP-complete**	**P** for $k, d = O(1)$
TSO	**NP-complete**	**P** for $k, d = O(1)$
PSO	**NP-complete**	**P** for $k, d = O(1)$
RA	$O(n \cdot k)$	**NP-complete** for $k, d = O(1)$
CC	$O(n \cdot k)$	**NP-complete** for $k, d = O(1)$
CCv	$O(n \cdot k)$	**NP-complete** for $k, d = O(1)$
CM	$O(n \cdot k)$	**NP-complete** for $k, d = O(1)$
Relaxed-Acyclic	$O(n)$	**NP-complete** for $k, d = O(1)$

Multi-copy atomicity?