Optimal Prediction of Synchronization-Preserving Races

Umang Mathur, **Andreas Pavlogiannis** and Mahesh Viswanathan

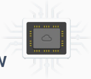
UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

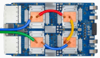
AARHUS UNIVERSITY

Concurrency : Software and Challenges

Ubiquitous computing paradigm

- Back-bone of big-data, AI revolutions

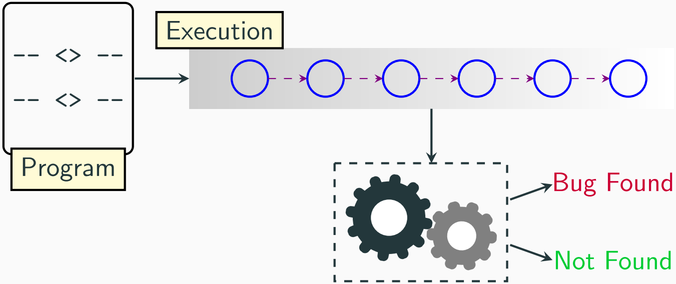




Challenging to write multi-threaded programs

- Large interleaving space
- Concurrency bugs
  - data races, deadlocks, etc.,
  - manifest in production despite rigorous testing
  - hard to reproduce
  - severe outcomes - loss of lives and money

Dynamic Analysis for Detecting Concurrency Bugs



Data Races

	t_1	t_2
1		$\mathsf{acq}(\ell)$
2		$w(x)$
3		$r(x)$
4		$\mathsf{rel}(\ell)$
5	$\mathsf{acq}(\ell)$	
6	$\mathbf{w(x)}$	$\mathbf{r(x)}$

Data Races

	t_1	t_2
1	acq(ℓ)	
2	$\mathbf{w}(\mathbf{x})$	
3	rel(ℓ)	
4		acq(ℓ)
5		$w(x)$
6		$r(x)$
7		rel(ℓ)
8		$\mathbf{r}(\mathbf{x})$

Observed trace σ

reordering $\longrightarrow$

	t_1	t_2
1		acq(ℓ)
2		$w(x)$
3		$r(x)$
4		rel(ℓ)
5	acq(ℓ)	
6	$\mathbf{w}(\mathbf{x})$	$\mathbf{r}(\mathbf{x})$

Witness trace σ^*

Data Races

	t_1	t_2
1	$\mathsf{acq}(\ell)$	
2	$\mathbf{w}(\mathbf{x})$	
3	$\mathsf{rel}(\ell)$	
4		$\mathsf{acq}(\ell)$
5		$w(x)$
6		$r(x)$
7		$\mathsf{rel}(\ell)$
8		$\mathbf{r}(\mathbf{x})$

$\xrightarrow{\text{reordering}}$

	t_1	t_2
1		$\mathsf{acq}(\ell)$
2		$w(x)$
3		$r(x)$
4		$\mathsf{rel}(\ell)$
5	$\mathsf{acq}(\ell)$	
6	$\mathbf{w}(\mathbf{x})$	$\mathbf{r}(\mathbf{x})$

Observed trace σ

Witness trace σ^*

Witness σ^*

- Executes a prefix of each thread
- Every read sees the same write as in σ

Happens-Before Races

Happens-Before Races

Happens-before principle

Do not reorder conflicting critical sections

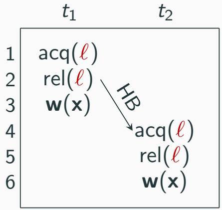
Happens-Before Races

Happens-before principle

Do not reorder conflicting critical sections

	t_1		t_2
1	acq(ℓ)		
2	rel(ℓ)		
3	**w**(**x**)	HB	
4			acq(ℓ)
5			rel(ℓ)
6			**w**(**x**)

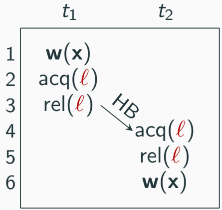
Happens-Before Races



Happens-before principle

Do not reorder conflicting critical sections

But!



- HB not complete for this principle
- Here a race is missed
- Exposed without reordering critical sections

Sync-Preserving Witness Reorderings

Definition (Sync-preserving witnesses)

A sync-preserving witness of trace σ is a witness σ^* such that for every two acquires $\mathsf{acq}_1(\ell), \mathsf{acq}_2(\ell) \in \sigma^*$ we have

$$\mathsf{acq}_1(\ell) <^{\sigma^*} \mathsf{acq}_2(\ell) \implies \mathsf{acq}_1(\ell) <^{\sigma} \mathsf{acq}_2(\ell)$$

Sync-Preserving Witness Reorderings

Definition (Sync-preserving witnesses)

A sync-preserving witness of trace σ is a witness σ^* such that for every two acquires $\mathsf{acq}_1(\ell), \mathsf{acq}_2(\ell) \in \sigma^*$ we have

$$\mathsf{acq}_1(\ell) <^{\sigma^*} \mathsf{acq}_2(\ell) \implies \mathsf{acq}_1(\ell) <^{\sigma} \mathsf{acq}_2(\ell)$$

	t_1	t_2	t_3
1	$\mathsf{acq}(\ell_1)$		
2	$\mathsf{rel}(\ell_1)$		
3		$\mathsf{acq}(\ell_2)$	
4		$\mathsf{rel}(\ell_2)$	
5		$\mathsf{acq}(\ell_1)$	
6		$\mathsf{rel}(\ell_1)$	
7			$\mathsf{acq}(\ell_2)$
8			$\mathsf{rel}(\ell_2)$

Observed trace

$\longrightarrow$

	t_1	t_2	t_3
1		$\mathsf{acq}(\ell_2)$	
2		$\mathsf{rel}(\ell_2)$	
3			$\mathsf{acq}(\ell_2)$
4			$\mathsf{rel}(\ell_2)$
5	$\mathsf{acq}(\ell_1)$		
6	$\mathsf{rel}(\ell_1)$		
7		$\mathsf{acq}(\ell_1)$	
8		$\mathsf{rel}(\ell_1)$	

sync-preserving witness

Sync-Preserving Data Races

Definition (Sync-preserving data races)

A predictable data race (e_1, e_2) of σ is sync-preserving if it has a sync-preserving witness

Sync-Preserving Data Races

Definition (Sync-preserving data races)

A predictable data race (e_1, e_2) of σ is sync-preserving if it has a sync-preserving witness

	t_1	t_2
1	$\mathbf{w}(\mathbf{x})$	
2	acq(ℓ)	
3	rel(ℓ)	
4		acq(ℓ)
5		rel(ℓ)
6		$\mathbf{w}(\mathbf{x})$

$\xrightarrow{\text{sync-preserving data race}}$

	t_1	t_2
1		acq(ℓ)
2		rel(ℓ)
3	$\mathbf{w}(\mathbf{x})$	$\mathbf{w}(\mathbf{x})$

Sync-Preserving Data Races

Definition (Sync-preserving data races)

A predictable data race (e_1, e_2) of σ is sync-preserving if it has a sync-preserving witness

	t_1	t_2
1	$\mathbf{w}(\mathbf{x})$	
2	$\mathsf{acq}(\ell)$	
3	$\mathsf{rel}(\ell)$	
4		$\mathsf{acq}(\ell)$
5		$\mathsf{rel}(\ell)$
6		$\mathbf{w}(\mathbf{x})$

$\xrightarrow{\text{sync-preserving data race}}$

	t_1	t_2
1		$\mathsf{acq}(\ell)$
2		$\mathsf{rel}(\ell)$
3	$\mathbf{w}(\mathbf{x})$	$\mathbf{w}(\mathbf{x})$

	t_1	t_2
1	$\mathsf{acq}(\ell)$	
2	$\mathbf{w}(\mathbf{x})$	
3	$\mathsf{rel}(\ell)$	
4		$\mathsf{acq}(\ell)$
5		$\mathsf{rel}(\ell)$
6		$\mathbf{w}(\mathbf{x})$

$\xrightarrow{\text{non-sync-preserving data race}}$

	t_1	t_2
1		$\mathsf{acq}(\ell)$
2		$\mathsf{rel}(\ell)$
3	$\mathsf{acq}(\ell)$	
4	$\mathbf{w}(\mathbf{x})$	$\mathbf{w}(\mathbf{x})$

How do sync-preserving races compare to HB races?

Sync-Preserving vs HB races

Theorem

Every HB *race is also sync-preserving.*

Sync-Preserving vs HB races

Theorem

Every HB *race is also sync-preserving.*

The opposite is not true

	t_1	t_2
1	$\mathbf{w}(\mathbf{x})$	
2	$\mathsf{acq}(\ell)$	
3	$\mathsf{rel}(\ell)$	
4		$\mathsf{acq}(\ell)$
5		$\mathsf{rel}(\ell)$
6		$\mathbf{w}(\mathbf{x})$

Sync-Preserving vs HB races

Theorem

Every HB *race is also sync-preserving.*

The opposite is not true

	t_1	t_2
1	$\mathbf{w}(\mathbf{x})$	
2	$\mathsf{acq}(\ell)$	
3	$\mathsf{rel}(\ell)$	
4		$\mathsf{acq}(\ell)$
5		$\mathsf{rel}(\ell)$
6		$\mathbf{w}(\mathbf{x})$

Sync-preserving races need not be consecutive

	t_1	t_2
1	$\mathbf{w}(\mathbf{x})$	
2	$\mathsf{acq}(\ell)$	
3	$w(x)$	
4	$\mathsf{rel}(\ell)$	
5		$\mathsf{acq}(\ell)$
6		$\mathbf{w}(\mathbf{x})$
7		$\mathsf{rel}(\ell)$

Sync-Preserving vs HB races

Theorem

Every HB *race is also sync-preserving.*

The opposite is not true

	t_1	t_2
1	$\mathbf{w}(\mathbf{x})$	
2	acq(ℓ)	
3	rel(ℓ)	
4		acq(ℓ)
5		rel(ℓ)
6		$\mathbf{w}(\mathbf{x})$

Sync-preserving races need not be consecutive

	t_1	t_2
1	$\mathbf{w}(\mathbf{x})$	
2	acq(ℓ)	
3	$w(x)$	
4	rel(ℓ)	
5		acq(ℓ)
6		$\mathbf{w}(\mathbf{x})$
7		rel(ℓ)

Sync-preservation is a notion broader than HB

How do sync-preserving races compare to other notions of *predictive* races?

Sync-preserving vs Predictive partial-orders

Sync-preserving vs WCP

1

Sync-preserving vs Predictive partial-orders

Sync-preserving vs WCP

Sync-preserving ✓

WCP ✗

	t_1	t_2
1	$\mathbf{w}(\mathbf{x})$	
2	$\mathrm{acq}(\ell)$	
3	$w(y)$	
4	$\mathrm{rel}(\ell)$	
5		$\mathrm{acq}(\ell)$
6		$w(y)$
7		$\mathrm{rel}(\ell)$
8		$\mathbf{w}(\mathbf{x})$

1

Sync-preserving vs Predictive partial-orders

Sync-preserving vs WCP

Sync-preserving ✓
WCP ✗

	t_1	t_2
1	$\mathbf{w}(\mathbf{x})$	
2	$\mathsf{acq}(\ell)$	
3	$w(y)$	
4	$\mathsf{rel}(\ell)$	
5		$\mathsf{acq}(\ell)$
6		$w(y)$
7		$\mathsf{rel}(\ell)$
8		$\mathbf{w}(\mathbf{x})$

WCP

Sync-preserving ✗
WCP ✓

	t_1	t_2
1	$\mathsf{acq}(\ell)$	
2	$\mathbf{w}(\mathbf{x})$	
3	$\mathsf{rel}(\ell)$	
4		$\mathsf{acq}(\ell)$
5		$\mathsf{rel}(\ell)$
6		$\mathbf{w}(\mathbf{x})$

1

Sync-preserving vs Predictive partial-orders

Sync-preserving vs WCP

Sync-preserving ✓
WCP ✗

	t_1	t_2
1	$\mathbf{w}(\mathbf{x})$	
2	$\mathsf{acq}(\ell)$	
3	$w(y)$	
4	$\mathsf{rel}(\ell)$	
5		$\mathsf{acq}(\ell)$
6		$w(y)$
7		$\mathsf{rel}(\ell)$
8		$\mathbf{w}(\mathbf{x})$

WCP

Sync-preserving ✗
WCP ✓

	t_1	t_2
1	$\mathsf{acq}(\ell)$	
2	$\mathbf{w}(\mathbf{x})$	
3	$\mathsf{rel}(\ell)$	
4		$\mathsf{acq}(\ell)$
5		$\mathsf{rel}(\ell)$
6		$\mathbf{w}(\mathbf{x})$

Other orders — DC, SDP[1] and WDP are unsound

[1] refer to paper for counter-example to soundness of SDP

How fast can we predict sync-preserving races?

SyncP – **An Algorithm For Sync-Preserving Races**

Theorem

SyncP *is a sound and complete algorithm for sync-preserving races. Given a trace* σ *of length* $\mathcal{N}$, SyncP *uses* $\widetilde{O}(\mathcal{N})$ *time and space.*

Theorem

SyncP *is a sound and complete algorithm for sync-preserving races. Given a trace* σ *of length* $\mathcal{N}$, SyncP *uses* $\widetilde{O}(\mathcal{N})$ *time and space.*

Soundness + Completeness + Efficiency

Sync-Preserving Ideals (SPIdeals)

How to decide if (e_1, e_2) is a sync-preserving race?

Sync-Preserving Ideals (SPIdeals)

How to decide if (e_1, e_2) is a sync-preserving race?

Compute the sync-preserving ideal $\mathsf{SPIdeal}(e_1, e_2)$

Sync-Preserving Ideals (SPIdeals)

How to decide if (e_1, e_2) is a sync-preserving race?

Compute the sync-preserving ideal SPIdeal(e_1, e_2)

	t_1	t_2	t_3
1		$\mathsf{acq}(\ell_1)$	
2		$w(x)$	
3		$\mathsf{rel}(\ell_1)$	
4	$\mathsf{acq}(\ell_1)$		
5	$\mathbf{e_1}$		
6	$\mathsf{rel}(\ell_1)$		
7	$\mathsf{acq}(\ell_2)$		
8	$\mathsf{rel}(\ell_2)$		
9			$r(x)$
10			$\mathsf{acq}(\ell_2)$
11			$\mathbf{e_2}$
12			$\mathsf{rel}(\ell_2)$

Sync-Preserving Ideals (SPIdeals)

How to decide if (e_1, e_2) is a sync-preserving race?

Compute the sync-preserving ideal SPIdeal(e_1, e_2)

	t_1	t_2	t_3
1		acq(ℓ_1)	
2		$w(x)$	
3		rel(ℓ_1)	
4	acq(ℓ_1)		
5	$\mathbf{e_1}$		
6	rel(ℓ_1)		
7	acq(ℓ_2)		
8	rel(ℓ_2)		
9			$r(x)$
10			acq(ℓ_2)
11			$\mathbf{e_2}$
12			rel(ℓ_2)

Sync-Preserving Ideals (SPIdeals)

How to decide if (e_1, e_2) is a sync-preserving race?

Compute the sync-preserving ideal $\mathsf{SPIdeal}(e_1, e_2)$

	t_1	t_2	t_3
1		$\mathsf{acq}(\ell_1)$	
2		$w(x)$	
3		$\mathsf{rel}(\ell_1)$	
4	$\mathsf{acq}(\ell_1)$		
5	$\mathbf{e_1}$		
6	$\mathsf{rel}(\ell_1)$		
7	$\mathsf{acq}(\ell_2)$		
8	$\mathsf{rel}(\ell_2)$		
9			$r(x)$
10			$\mathsf{acq}(\ell_2)$
11			$\mathbf{e_2}$
12			$\mathsf{rel}(\ell_2)$

Sync-Preserving Ideals (SPIdeals)

How to decide if (e_1, e_2) is a sync-preserving race?

Compute the sync-preserving ideal SPIdeal(e_1, e_2)

	t_1	t_2	t_3
1		acq(ℓ_1)	
2		$w(x)$	
3		rel(ℓ_1)	
4	acq(ℓ_1)		
5	$\mathbf{e_1}$		
6	rel(ℓ_1)		
7	acq(ℓ_2)		
8	rel(ℓ_2)		
9			$r(x)$
10			acq(ℓ_2)
11			$\mathbf{e_2}$
12			rel(ℓ_2)

Sync-Preserving Ideals (SPIdeals)

How to decide if (e_1, e_2) is a sync-preserving race?

Compute the sync-preserving ideal $\mathsf{SPIdeal}(e_1, e_2)$

	t_1	t_2	t_3
1		$\mathsf{acq}(\ell_1)$	
2		$w(x)$	
3		$\mathsf{rel}(\ell_1)$	
4	$\mathsf{acq}(\ell_1)$		
5	$\mathbf{e_1}$		
6	$\mathsf{rel}(\ell_1)$		
7	$\mathsf{acq}(\ell_2)$		
8	$\mathsf{rel}(\ell_2)$		
9			$r(x)$
10			$\mathsf{acq}(\ell_2)$
11			$\mathbf{e_2}$
12			$\mathsf{rel}(\ell_2)$

Sync-preserving race!

Sync-Preserving Ideals (SPIdeals)

How to decide if (e_1, e_2) is a sync-preserving race?

Compute the sync-preserving ideal SPIdeal(e_1, e_2)

	t_1	t_2	t_3
1		$acq(\ell_1)$	
2		$w(x)$	
3		$rel(\ell_1)$	
4	$acq(\ell_1)$		
5	$\mathbf{e_1}$		
6	$rel(\ell_1)$		
7	$acq(\ell_2)$		
8	$rel(\ell_2)$		
9			$r(x)$
10			$acq(\ell_2)$
11			$\mathbf{e_2}$
12			$rel(\ell_2)$

Sync-preserving race!

	t_1	t_2	t_3
1		$acq(\ell_1)$	
2		$w(x)$	
3	$r(x)$		
4	$\mathbf{e_1}$		
5	$w(y)$		
6		$r(y)$	
7		$rel(\ell_1)$	
8			$acq(\ell_1)$
9			$\mathbf{e_2}$
10			$rel(\ell_1)$

Sync-Preserving Ideals (SPIdeals)

How to decide if (e_1, e_2) is a sync-preserving race?

Compute the sync-preserving ideal SPIdeal(e_1, e_2)

	t_1	t_2	t_3
1		acq(ℓ_1)	
2		$w(x)$	
3		rel(ℓ_1)	
4	acq(ℓ_1)		
5	$\mathbf{e_1}$		
6	rel(ℓ_1)		
7	acq(ℓ_2)		
8	rel(ℓ_2)		
9			$r(x)$
10			acq(ℓ_2)
11			$\mathbf{e_2}$
12			rel(ℓ_2)

Sync-preserving race!

	t_1	t_2	t_3
1		acq(ℓ_1)	
2		$w(x)$	
3	$r(x)$		
4	$\mathbf{e_1}$		
5	$w(y)$		
6		$r(y)$	
7		rel(ℓ_1)	
8			acq(ℓ_1)
9			$\mathbf{e_2}$
10			rel(ℓ_1)

Sync-Preserving Ideals (SPIdeals)

How to decide if (e_1, e_2) is a sync-preserving race?

Compute the sync-preserving ideal SPIdeal(e_1, e_2)

	t_1	t_2	t_3
1		acq(ℓ_1)	
2		$w(x)$	
3		rel(ℓ_1)	
4	acq(ℓ_1)		
5	$\mathbf{e_1}$		
6	rel(ℓ_1)		
7	acq(ℓ_2)		
8	rel(ℓ_2)		
9			$r(x)$
10			acq(ℓ_2)
11			$\mathbf{e_2}$
12			rel(ℓ_2)

Sync-preserving race!

	t_1	t_2	t_3
1		acq(ℓ_1)	
2		$w(x)$	
3	$r(x)$		
4	$\mathbf{e_1}$		
5	$w(y)$		
6		$r(y)$	
7		rel(ℓ_1)	
8			acq(ℓ_1)
9			$\mathbf{e_2}$
10			rel(ℓ_1)

Sync-Preserving Ideals (SPIdeals)

How to decide if (e_1, e_2) is a sync-preserving race?

Compute the sync-preserving ideal SPIdeal(e_1, e_2)

	t_1	t_2	t_3
1		$\mathsf{acq}(\ell_1)$	
2		$w(x)$	
3		$\mathsf{rel}(\ell_1)$	
4	$\mathsf{acq}(\ell_1)$		
5	$\mathbf{e_1}$		
6	$\mathsf{rel}(\ell_1)$		
7	$\mathsf{acq}(\ell_2)$		
8	$\mathsf{rel}(\ell_2)$		
9			$r(x)$
10			$\mathsf{acq}(\ell_2)$
11			$\mathbf{e_2}$
12			$\mathsf{rel}(\ell_2)$

Sync-preserving race!

	t_1	t_2	t_3
1		$\mathsf{acq}(\ell_1)$	
2		$w(x)$	
3	$r(x)$		
4	$\mathbf{e_1}$		
5	$w(y)$		
6		$r(y)$	
7		$\mathsf{rel}(\ell_1)$	
8			$\mathsf{acq}(\ell_1)$
9			$\mathbf{e_2}$
10			$\mathsf{rel}(\ell_1)$

Sync-Preserving Ideals (SPIdeals)

How to decide if (e_1, e_2) is a sync-preserving race?

Compute the sync-preserving ideal SPIdeal(e_1, e_2)

	t_1	t_2	t_3
1		acq(ℓ_1)	
2		$w(x)$	
3		rel(ℓ_1)	
4	acq(ℓ_1)		
5	$\mathbf{e_1}$		
6	rel(ℓ_1)		
7	acq(ℓ_2)		
8	rel(ℓ_2)		
9			$r(x)$
10			acq(ℓ_2)
11			$\mathbf{e_2}$
12			rel(ℓ_2)

Sync-preserving race!

	t_1	t_2	t_3
1		acq(ℓ_1)	
2		$w(x)$	
3	$r(x)$		
4	$\mathbf{e_1}$		
5	$w(y)$		
6		$r(y)$	
7		rel(ℓ_1)	
8			acq(ℓ_1)
9			$\mathbf{e_2}$
10			rel(ℓ_1)

Sync-Preserving Ideals (SPIdeals)

How to decide if (e_1, e_2) is a sync-preserving race?

Compute the sync-preserving ideal $\text{SPIdeal}(e_1, e_2)$

	t_1	t_2	t_3
1		$\text{acq}(\ell_1)$	
2		$w(x)$	
3		$\text{rel}(\ell_1)$	
4	$\text{acq}(\ell_1)$		
5	$\mathbf{e_1}$		
6	$\text{rel}(\ell_1)$		
7	$\text{acq}(\ell_2)$		
8	$\text{rel}(\ell_2)$		
9			$r(x)$
10			$\text{acq}(\ell_2)$
11			$\mathbf{e_2}$
12			$\text{rel}(\ell_2)$

Sync-preserving race!

	t_1	t_2	t_3
1		$\text{acq}(\ell_1)$	
2		$w(x)$	
3	$r(x)$		
4	$\mathbf{e_1}$		
5	$w(y)$		
6		$r(y)$	
7		$\text{rel}(\ell_1)$	
8			$\text{acq}(\ell_1)$
9			$\mathbf{e_2}$
10			$\text{rel}(\ell_1)$

No sync-preserving race!

Sync-Preserving Ideals (SPIdeals)

How to decide if (e_1, e_2) is a sync-preserving race?

Compute the sync-preserving ideal $\mathsf{SPIdeal}(e_1, e_2)$

	t_1	t_2	t_3
1		$\mathsf{acq}(\ell_1)$	
2		$w(x)$	
3		$\mathsf{rel}(\ell_1)$	
4	$\mathsf{acq}(\ell_1)$		
5	$\mathbf{e_1}$		
6	$\mathsf{rel}(\ell_1)$		
7	$\mathsf{acq}(\ell_2)$		
8	$\mathsf{rel}(\ell_2)$		
9			$r(x)$
10			$\mathsf{acq}(\ell_2)$
11			$\mathbf{e_2}$
12			$\mathsf{rel}(\ell_2)$

Sync-preserving race!

	t_1	t_2	t_3
1		$\mathsf{acq}(\ell_1)$	
2		$w(x)$	
3	$r(x)$		
4	$\mathbf{e_1}$		
5	$w(y)$		
6		$r(y)$	
7		$\mathsf{rel}(\ell_1)$	
8			$\mathsf{acq}(\ell_1)$
9			$\mathbf{e_2}$
10			$\mathsf{rel}(\ell_1)$

No sync-preserving race!

Theorem

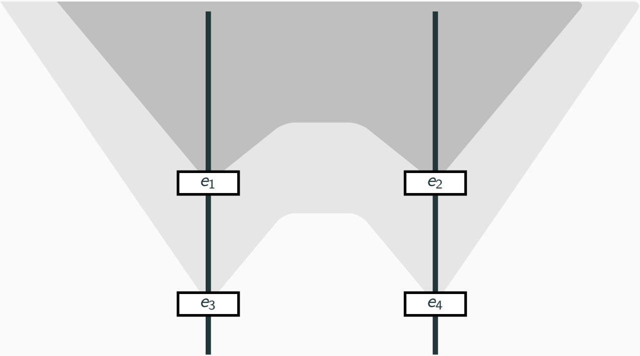
(e_1, e_2) *is a sync-preserving race iff* $\mathsf{SPIdeal}(e_1, e_2) \cap \{e_1, e_2\} = \emptyset$

Monotonicity of SPIdeals

How to test for all racy events?

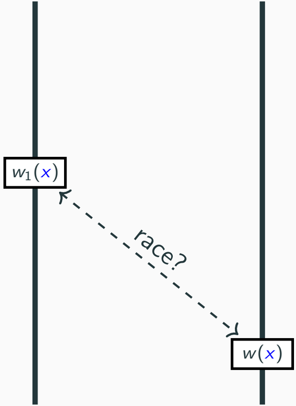
Monotonicity of SPIdeals

How to test for all racy events?

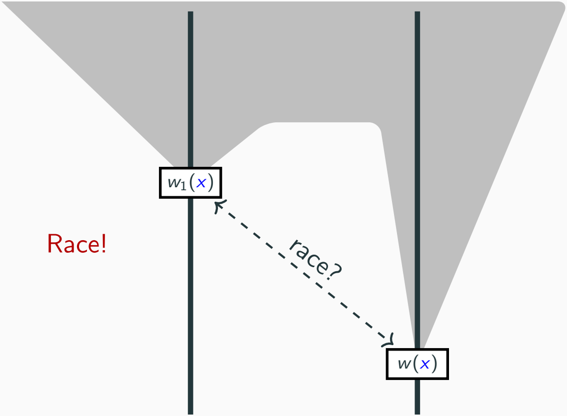


$$\mathrm{SPIdeal}(e_1, e_2) \subseteq \mathrm{SPIdeal}(e_3, e_4)$$

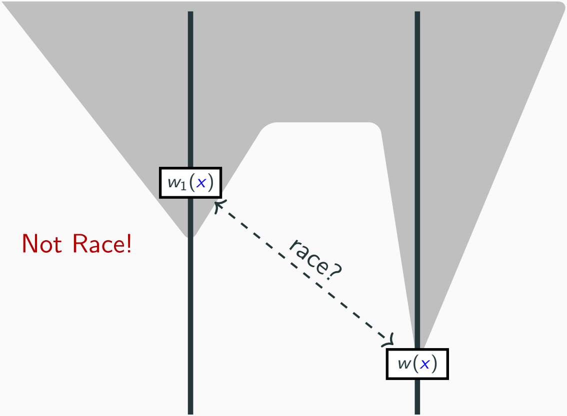
Detecting All Racy Events



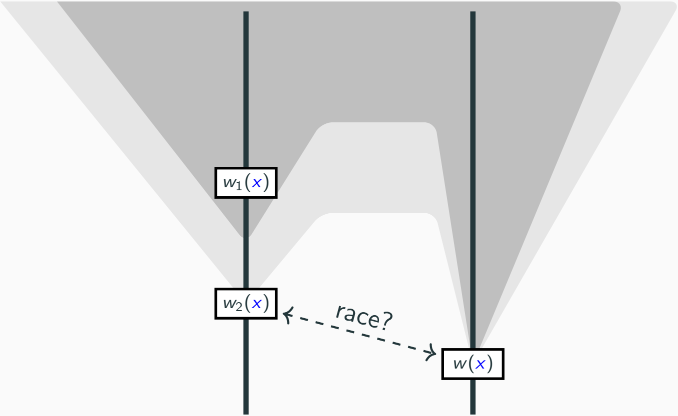
Detecting All Racy Events



Detecting All Racy Events

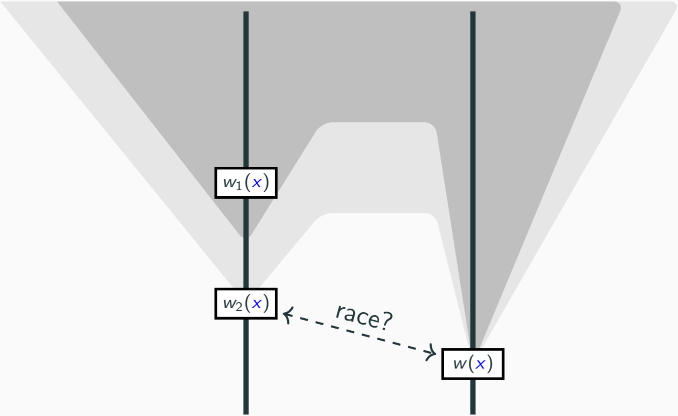


Detecting All Racy Events



Pay the cost for SPIdeal($w_2(x), w(x)$) $\setminus$ SPIdeal($w_1(x), w(x)$)

Detecting All Racy Events



Pay the cost for $\mathrm{SPIdeal}(w_2(x), w(x)) \setminus \mathrm{SPIdeal}(w_1(x), w(x))$

Repeat for all thread pairs and variables

What is the time/space complexity of sync-preserving races?

Optimality

Theorem

SyncP *runs in* $\widetilde{O}(\mathcal{N})$ *time and space*

- Clearly, time is almost optimal (i.e., almost linear)

- Can space be improved?
- E.g., HB takes only $\widetilde{O}(1)$ space

Optimality

Theorem

SyncP *runs in* $\widetilde{O}(\mathcal{N})$ *time and space*

- Clearly, time is almost optimal (i.e., almost linear)

- Can space be improved?
- E.g., HB takes only $\widetilde{O}(1)$ space

Theorem

Any streaming (i.e., one-pass) algorithm that decides whether there is a sync-preserving race must use $\Omega(\mathcal{N} / \log^2 \mathcal{N})$ *space.*

Optimality

Theorem

SyncP *runs in* $\widetilde{O}(\mathcal{N})$ *time and space*

- Clearly, time is almost optimal (i.e., almost linear)

- Can space be improved?
- E.g., HB takes only $\widetilde{O}(1)$ space

Theorem

Any streaming (i.e., one-pass) algorithm that decides whether there is a sync-preserving race must use $\Omega(\mathcal{N} / \log^2 \mathcal{N})$ *space.*

Optimality

SyncP is nearly optimal for both time and space.

What happens beyond sync-preserving races?

Parametric On Sync-Reversals

- Sync-preserving: "Do not try to reorder any critical section on σ"
- What if we try to reorder only a few?

Parametric On Sync-Reversals

- Sync-preserving: "Do not try to reorder any critical section on σ"
- What if we try to reorder only a few?

k-sync-reversal races

Exposed by reordering at most k critical sections of σ

- Sync-preserving = 0-sync-reversals

Parametric On Sync-Reversals

- Sync-preserving: "Do not try to reorder any critical section on σ"
- What if we try to reorder only a few?

k-sync-reversal races

Exposed by reordering at most k critical sections of σ

- Sync-preserving $=$ 0-sync-reversals

Question

What is the cost of k-sync-reversal races when k is small (e.g., $k \leq 5$)?

- $\widetilde{O}(\mathcal{N})$ when $k = 0$
- Hopefully small when $k = 1$ (?)

Hardness of 1 Sync-Reversals

Theorem

Dynamic race prediction on traces with a single lock and two critical sections is W[1]*-hard parameterized by the number of threads.*

Hardness of 1 Sync-Reversals

Theorem

Dynamic race prediction on traces with a single lock and two critical sections is W[1]*-hard parameterized by the number of threads.*

- NP complete
- No algorithm in time $2^{\mathcal{T}} \cdot \mathcal{N}^{O(1)}$
    - $\mathcal{N}$ events, $\mathcal{T}$ threads

Hardness of 1 Sync-Reversals

Theorem

Dynamic race prediction on traces with a single lock and two critical sections is W[1]*-hard parameterized by the number of threads.*

- NP complete
- No algorithm in time $2^{\mathcal{T}} \cdot \mathcal{N}^{O(1)}$
  - $\mathcal{N}$ events, $\mathcal{T}$ threads

Moral

The smallest possible synchronization (just 2 critical sections!) makes the problem as hard as in the general case.

Hardness of 1 Sync-Reversals - Intuition

acq(ℓ)

$\downarrow$

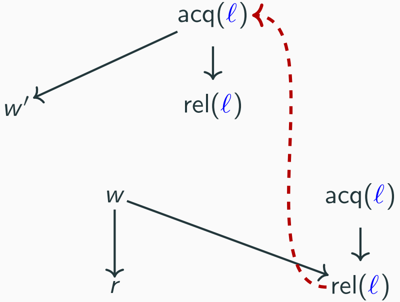
rel(ℓ)

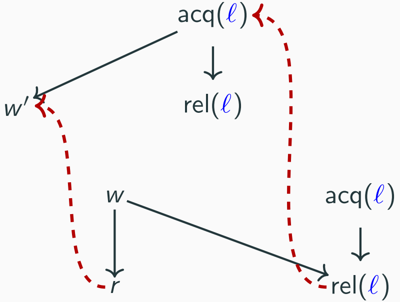
acq(ℓ)

$\downarrow$

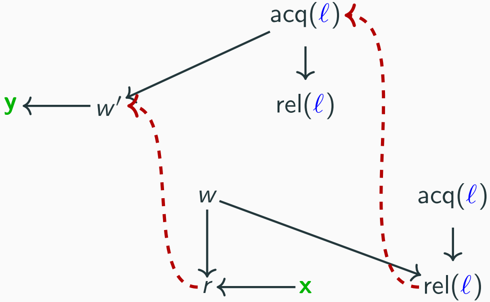
rel(ℓ)

Hardness of 1 Sync-Reversals - Intuition

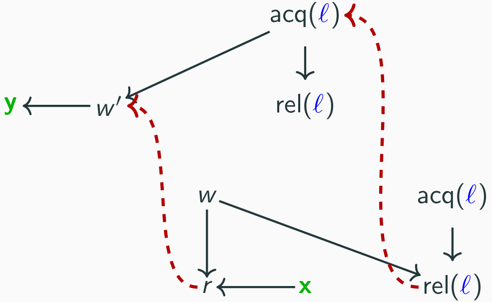



acq(ℓ)
w'
rel(ℓ)
w
acq(ℓ)
r
rel(ℓ)

Hardness of 1 Sync-Reversals - Intuition



Hardness of 1 Sync-Reversals - Intuition



- A *single* sync-reversal can imply *arbitrary* many
- Problem as hard as general race prediction
- U. Mathur, A. Pavlogiannis, and M. Viswanathan, "The Complexity of Dynamic Data Race Prediction" LICS '20 (2020)

Experiments

Setup

- Implementation of SyncP in Rapid[2]
- Comparison with SHB, WCP, M2
- Counted sound racy events reported
- 1h timeout

[2] https://github.com/umangm/rapid

Results

Benchmark	SHB		WCP		M2		SyncP	
	Races	**Time**	**Races**	**Time**	**Races**	**Time**	**Races**	**Time**
...	...	...	...	...	...	...	...	...
`wronglock`	2	0.02s	2	0.09s	2	0.43s	**2**	0.15s
`clean`	4	0.04s	4	0.14s	4	0.85s	**4**	0.17s
`mergesort`	1	11m10s	1	0.12s	2	0.96s	1	0.13s
`readswrites`	4	0.12s	4	0.41s	4	12.74s	**4**	0.77s
`ftpserver`	21	6.91s	21	1.34s	21	4.11s	**21**	4.69s
`derby`	10	0.01s	10	16.48s	11	22.49s	10	24.07s
`jigsaw`	4	0.41s	4	19.53s	6	11.69s	**6**	17.30s
`sunflow`	6	39.66s	6	47.14s	7	50.24s	**7**	55.30s
`linkedlist`	4	7.25s	3	27.07s	TO	TO	**4**	5m19s
`cryptorsa`	5	3m4s	5	6m35s	TO	TO	**7**	9m42s
`xalan`	10	0.15s	7	15m30s	TO	TO	**12**	10m44s
`luindex`	1	24m40s	2	31m6s	TO	TO	**15**	31m46s
Totals	157	1h51m	134	3h15m	131	8h30m	**178**	2h40m

Overall

Observation

Most races are sync-preserving and they can be detected fast.

Conclusion

Sync-preserving races:

- Subsume HB races
- Need not be consecutive
- Can be detected in $\widetilde{O}(\mathcal{N})$ time and space
- Are abundant in practice, more than HB races

Conclusion

Sync-preserving races:

- Subsume HB races
- Need not be consecutive
- Can be detected in $\widetilde{O}(\mathcal{N})$ time and space
- Are abundant in practice, more than HB races

- The $\widetilde{O}(\mathcal{N})$ bound is optimal for both time and space
- Relaxing the definition even a bit makes race prediction intractable
    - Races with only 1 sync-reversal are NP-hard to detect

Conclusion

Sync-preserving races:

- Subsume HB races
- Need not be consecutive
- Can be detected in $\widetilde{O}(\mathcal{N})$ time and space
- Are abundant in practice, more than HB races

- The $\widetilde{O}(\mathcal{N})$ bound is optimal for both time and space
- Relaxing the definition even a bit makes race prediction intractable
  - Races with only 1 sync-reversal are NP-hard to detect

Thank you!