Reasoning about Transactional Memory

Transitioning ideas from academia to industry

Nathan Chong, John Wickerson and Tyler Sorensen

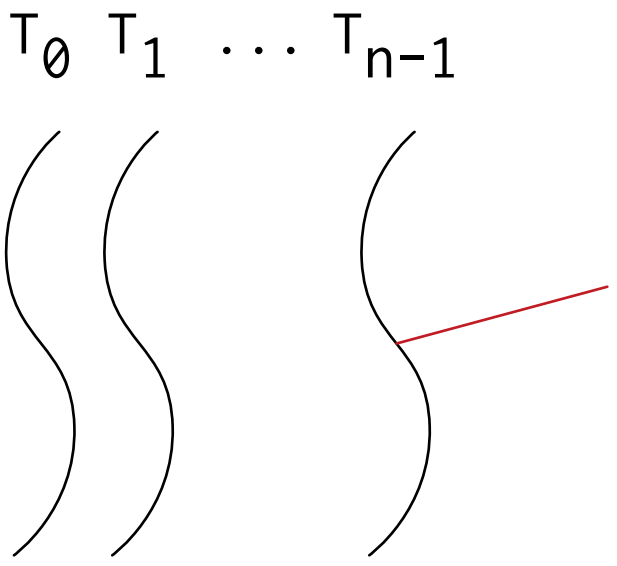
FMATS-5

21 September 2017

Architecture: fundamental contract between hardware/software

Memory Models

Multiprocessor program



a sequence of assembly instructions (ADD, CMP, B, LDR, STR, …)

Essential Question: What values can this load return?

Weaker

Alpha

Power ARM RISC-V

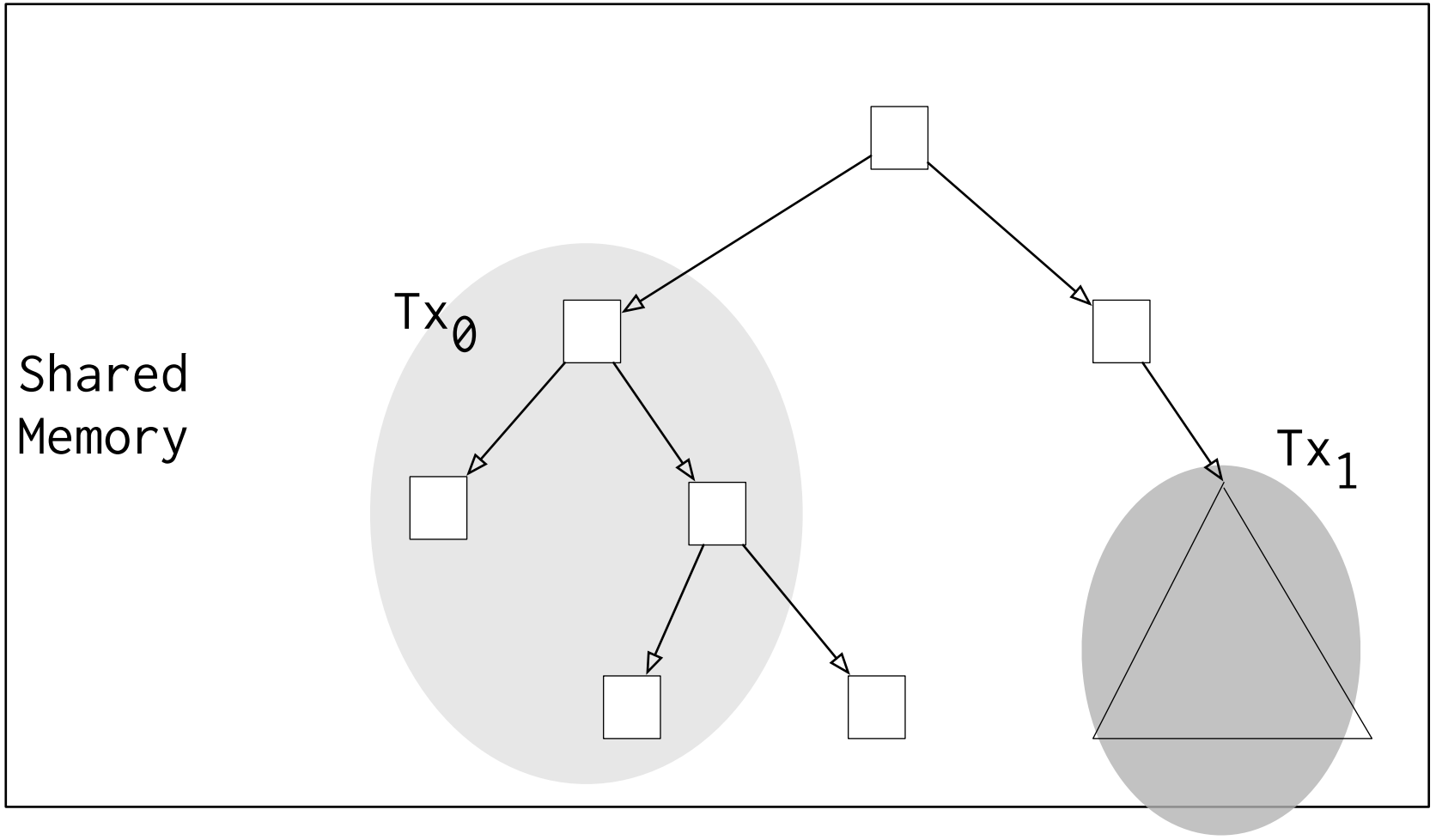
ARM v8.0

x86 (TSO)

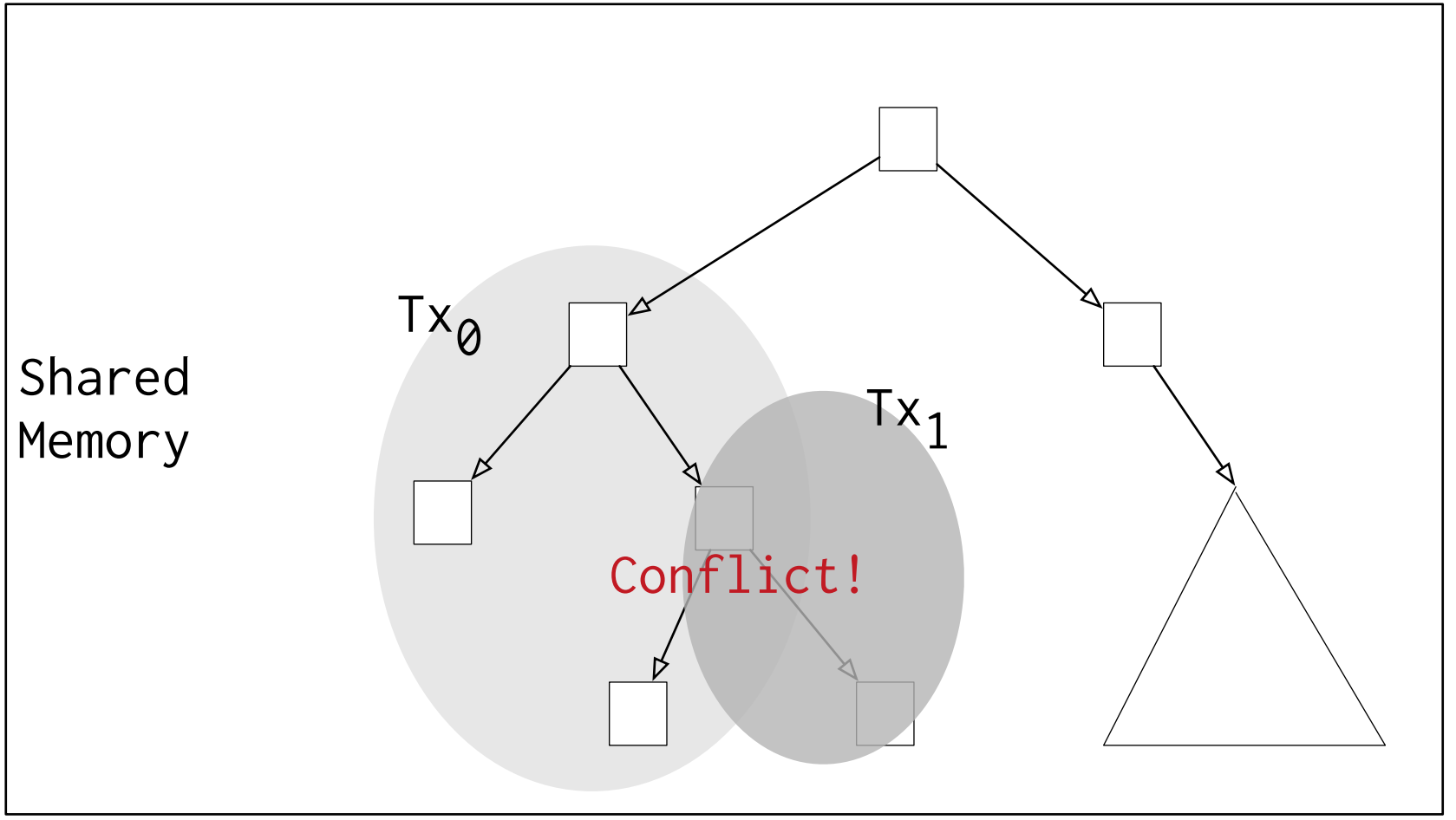
Stronger

Sequential Consistency

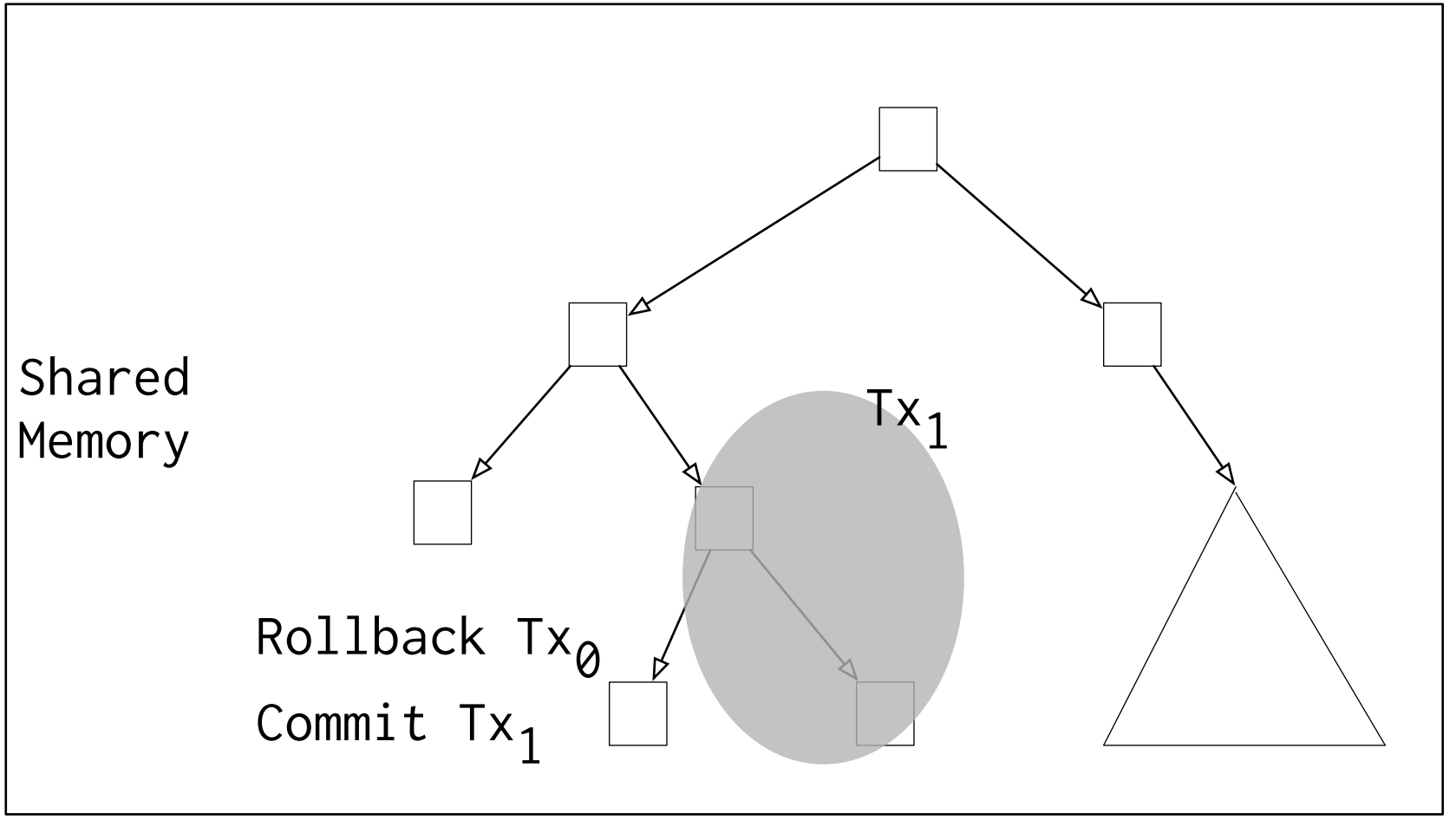
Transactional Memory: Optimistic Concurrency



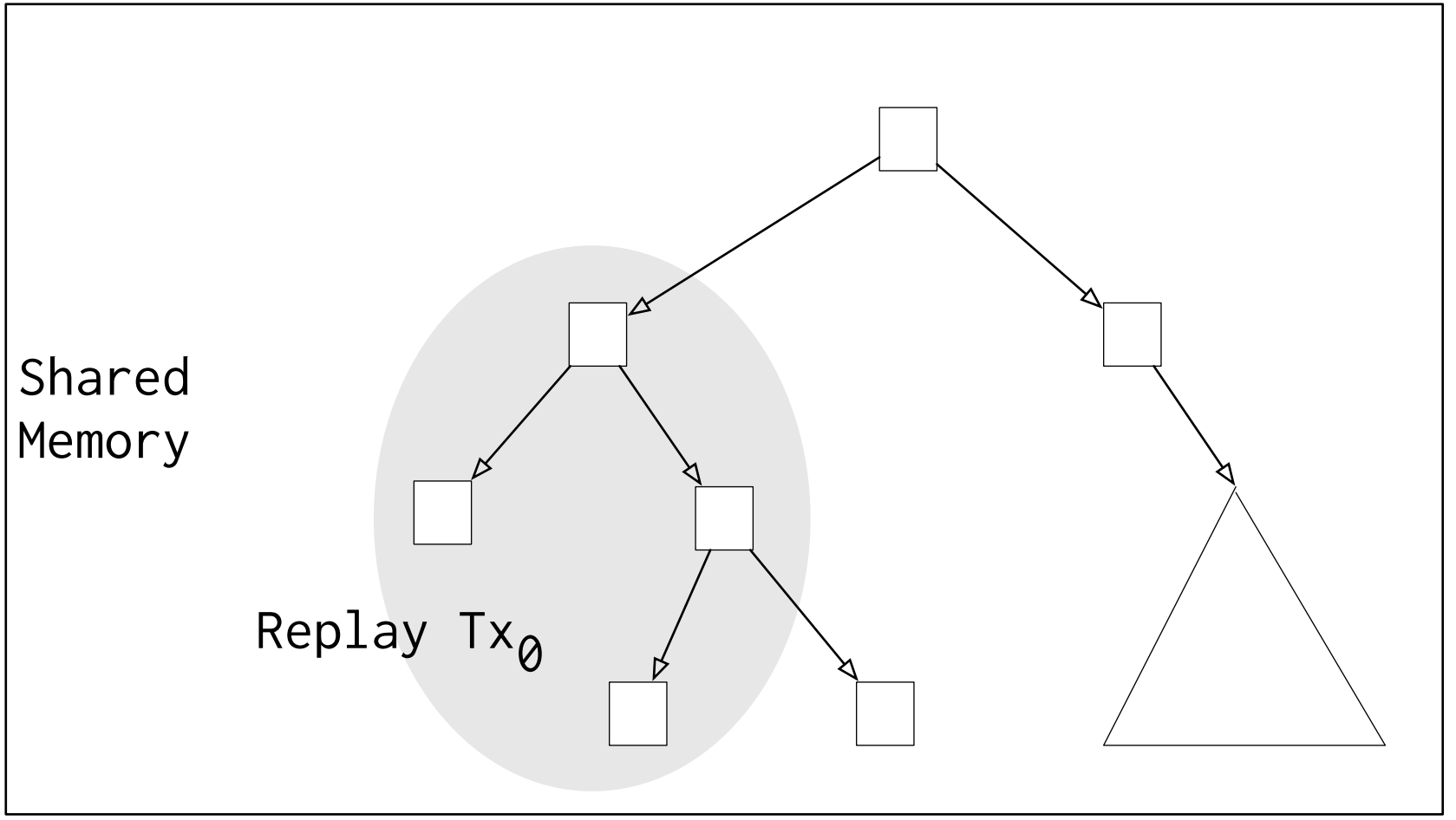
Transactional Memory: Optimistic Concurrency

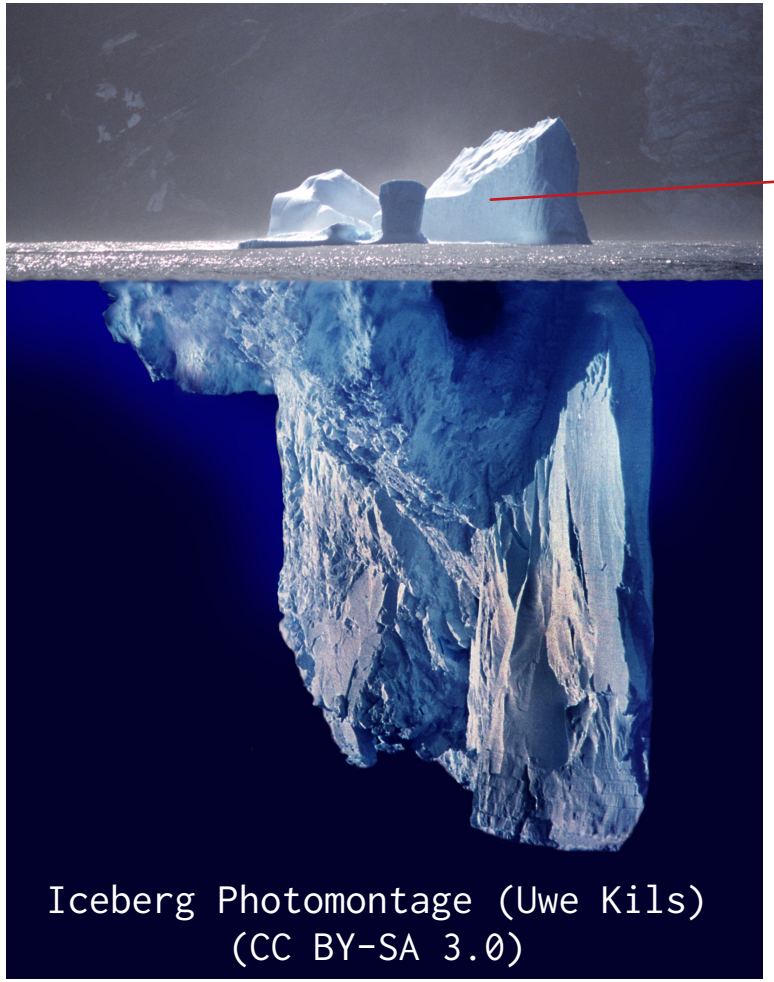


Transactional Memory: Optimistic Concurrency



Transactional Memory: Optimistic Concurrency





Iceberg Photomontage (Uwe Kils)
(CC BY-SA 3.0)

ISA Changes: TXSTART, TXCOMMIT, TXABORT, TXTEST

Impact on Memory Model

Interaction with exceptions, virtualisation, vector-extensions, debug, …

Unintended consequences
- KASLR attack [JLK16]
- Prime+Abort [DKP+17]

This Talk

- Principled method for refining TM models
  - x86, Power, Armv8, C++
  - Automatic generation of minimal conformance testsuites
  - Transferring this technique to engineers

- *The tricky case of aborting transactions*

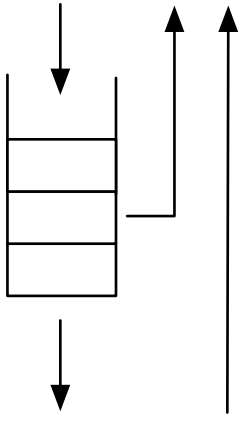
```
x=1;     ||     y=1;
r0=y;    ||     r1=x;
```

	Final values of r0,r1			
	1,1	1,0	0,1	0,0
SC	✓	✓	✓	✗
x86	✓	✓	✓	✓

```
x=1;          ||     y=1;
r0=y;         ||     r1=x;
```

Final values of r0,r1

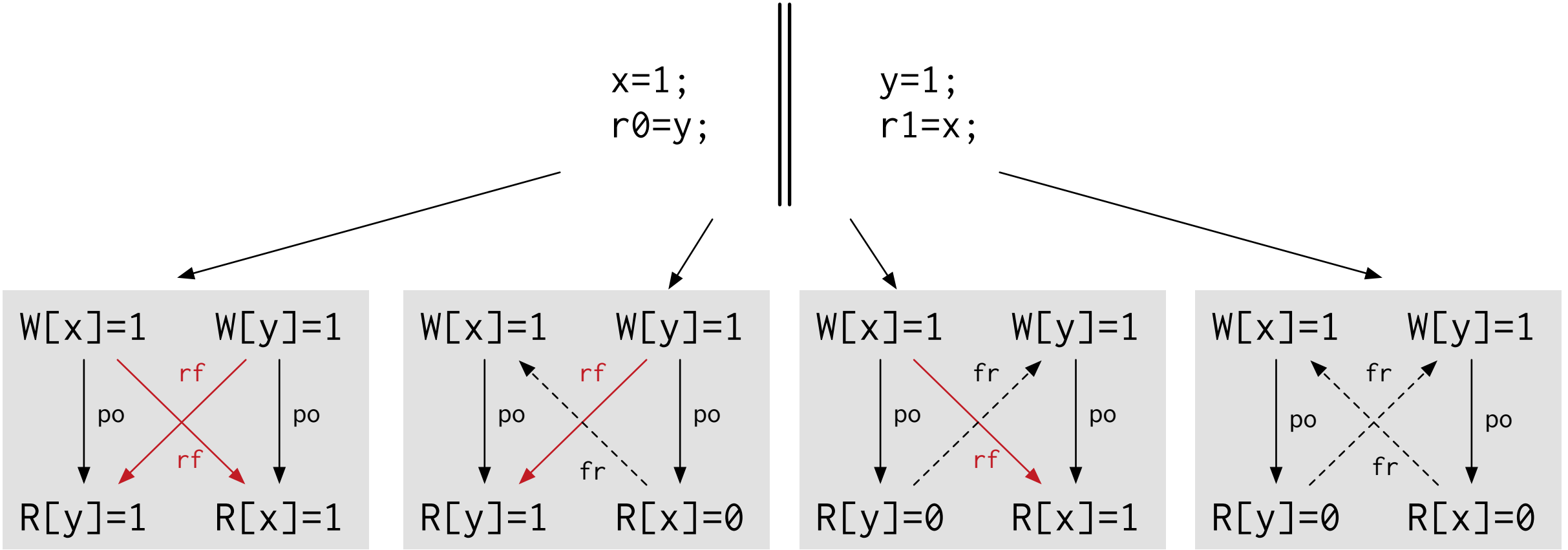
	1,1	1,0	0,1	0,0
SC	✓	✓	✓	✗
x86	✓	✓	✓	✓



Store Buffer

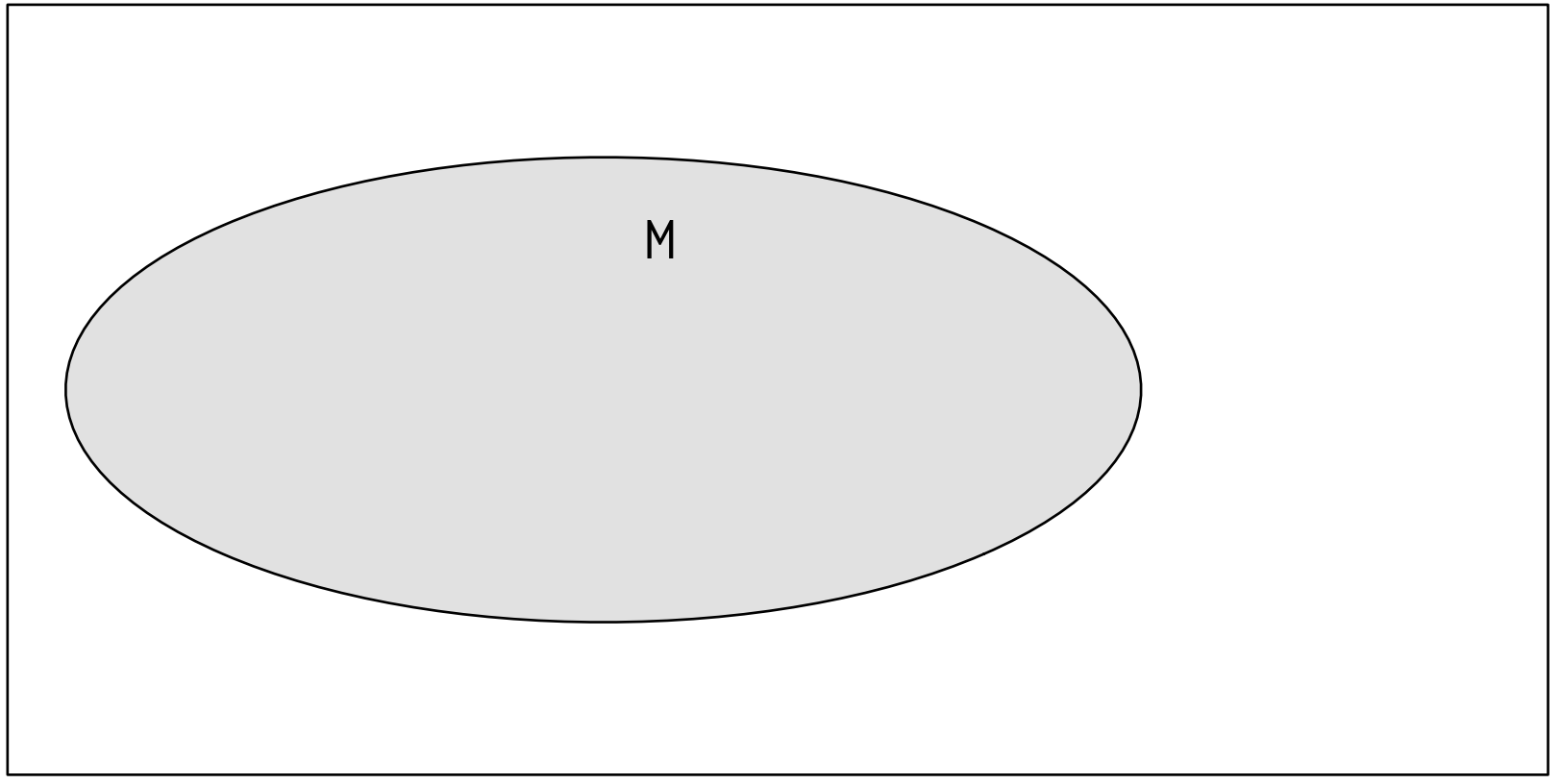
```
tx {          ||  tx {
  x=1;        ||    y=1;
  r0=y;       ||    r1=x;
}             ||  }
```

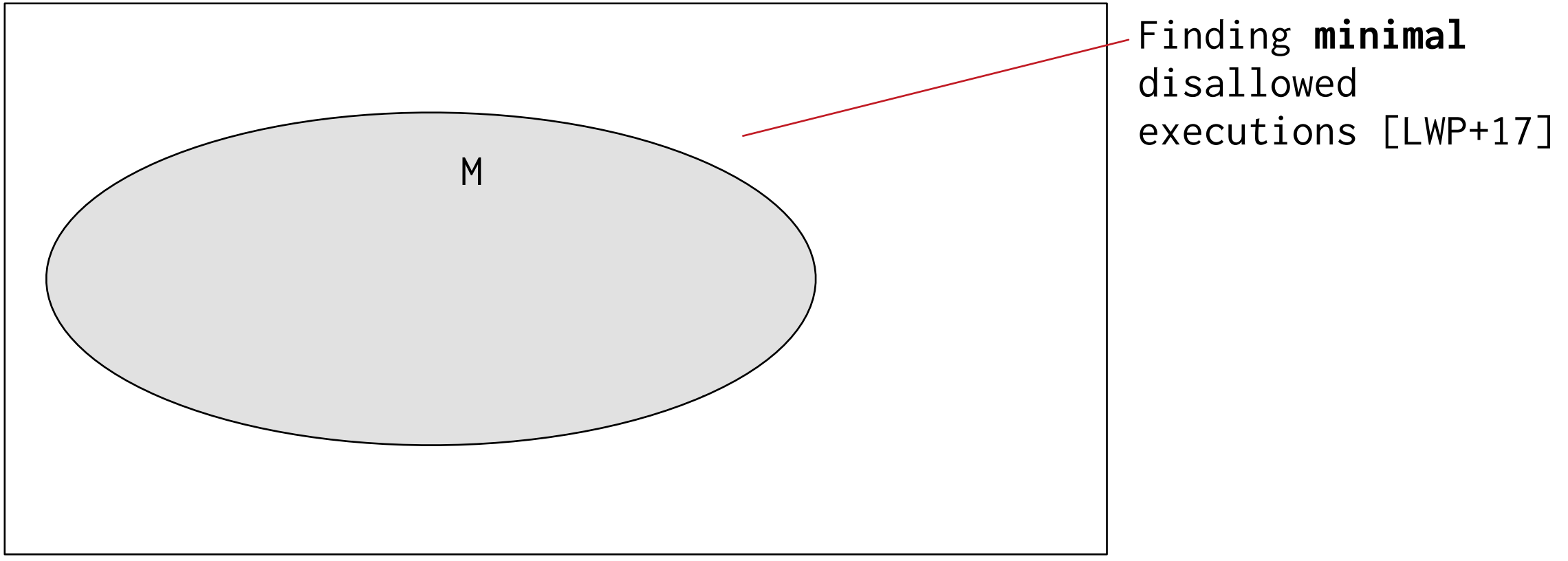
	Final values of r0,r1			
	1,1	1,0	0,1	0,0
SC	✓	✓	✓	✗
x86	✓	✓	✓	✓
x86+TM	✗	✓	✓	✗

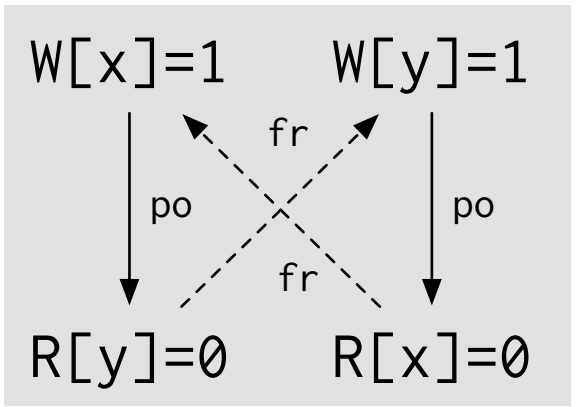

x=1;
r0=y;
y=1;
r1=x;
W[x]=1
W[y]=1
rf
po
po
rf
R[y]=1
R[x]=1
W[x]=1
W[y]=1
rf
po
po
fr
R[y]=1
R[x]=0
W[x]=1
W[y]=1
fr
po
po
rf
R[y]=0
R[x]=1
W[x]=1
W[y]=1
fr
po
po
fr
R[y]=0
R[x]=0

SC	✓	✓	✓	✗
x86	✓	✓	✓	✓

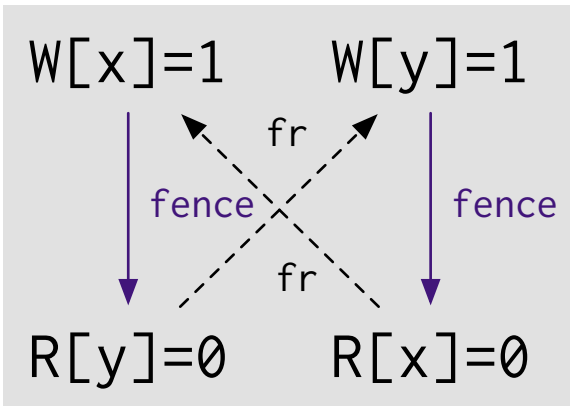
The set of all executions



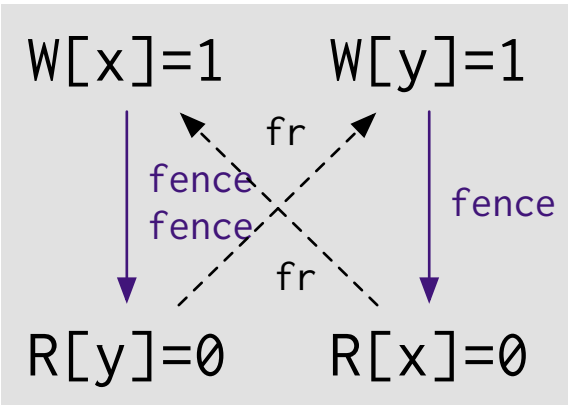

The set of all executions
M
Finding **minimal** disallowed executions [LWP+17]

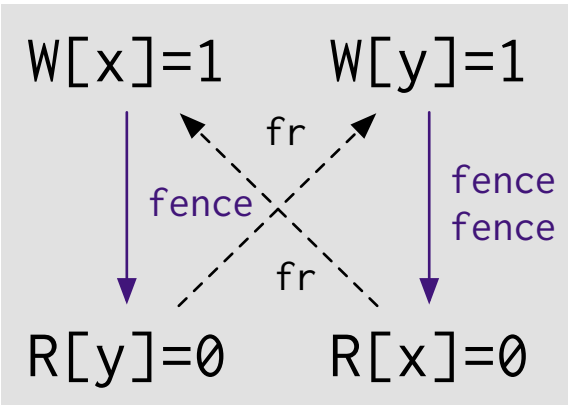


$\sqsubset$



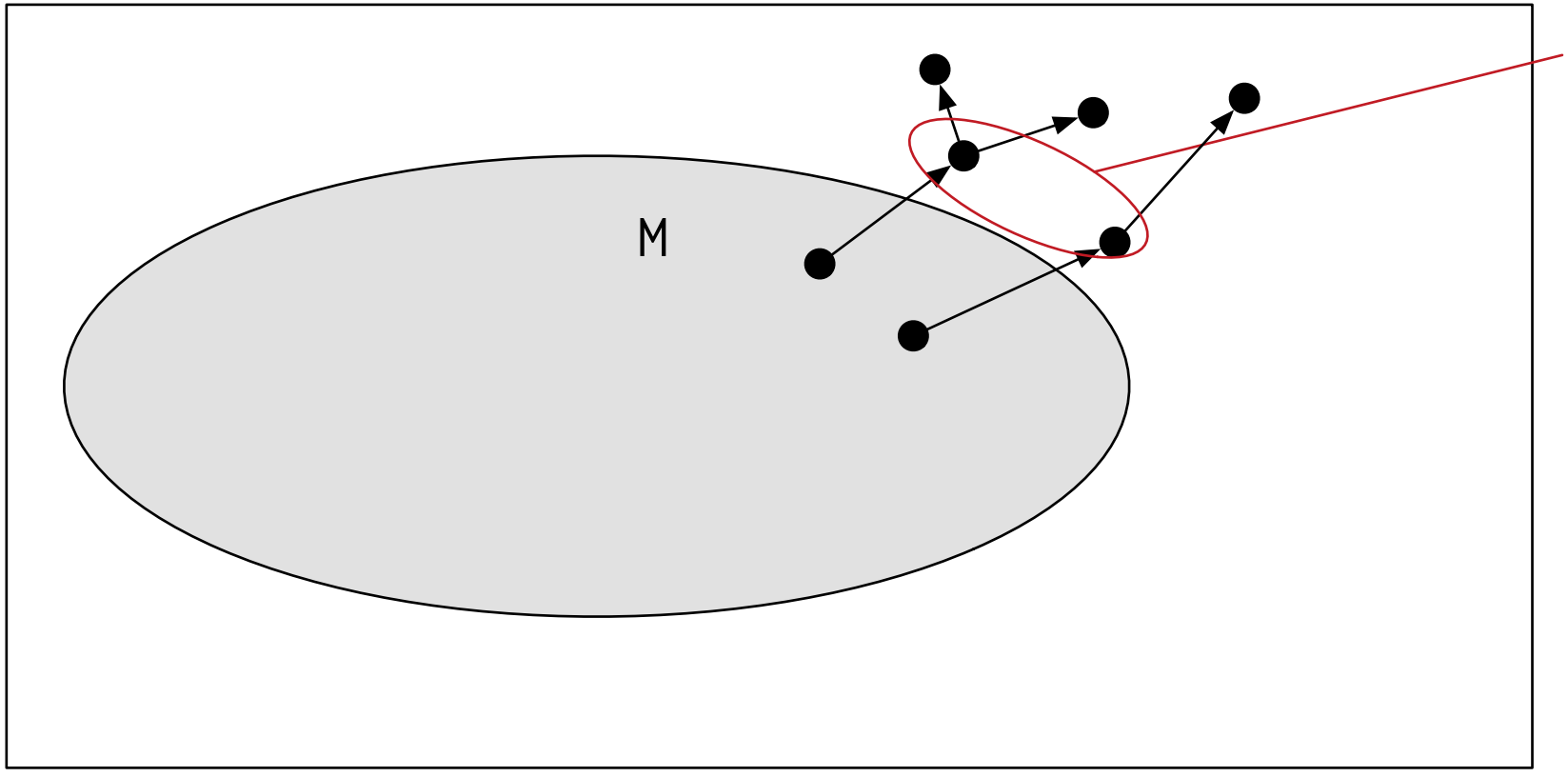
$\sqsubset$ $\sqsubset$





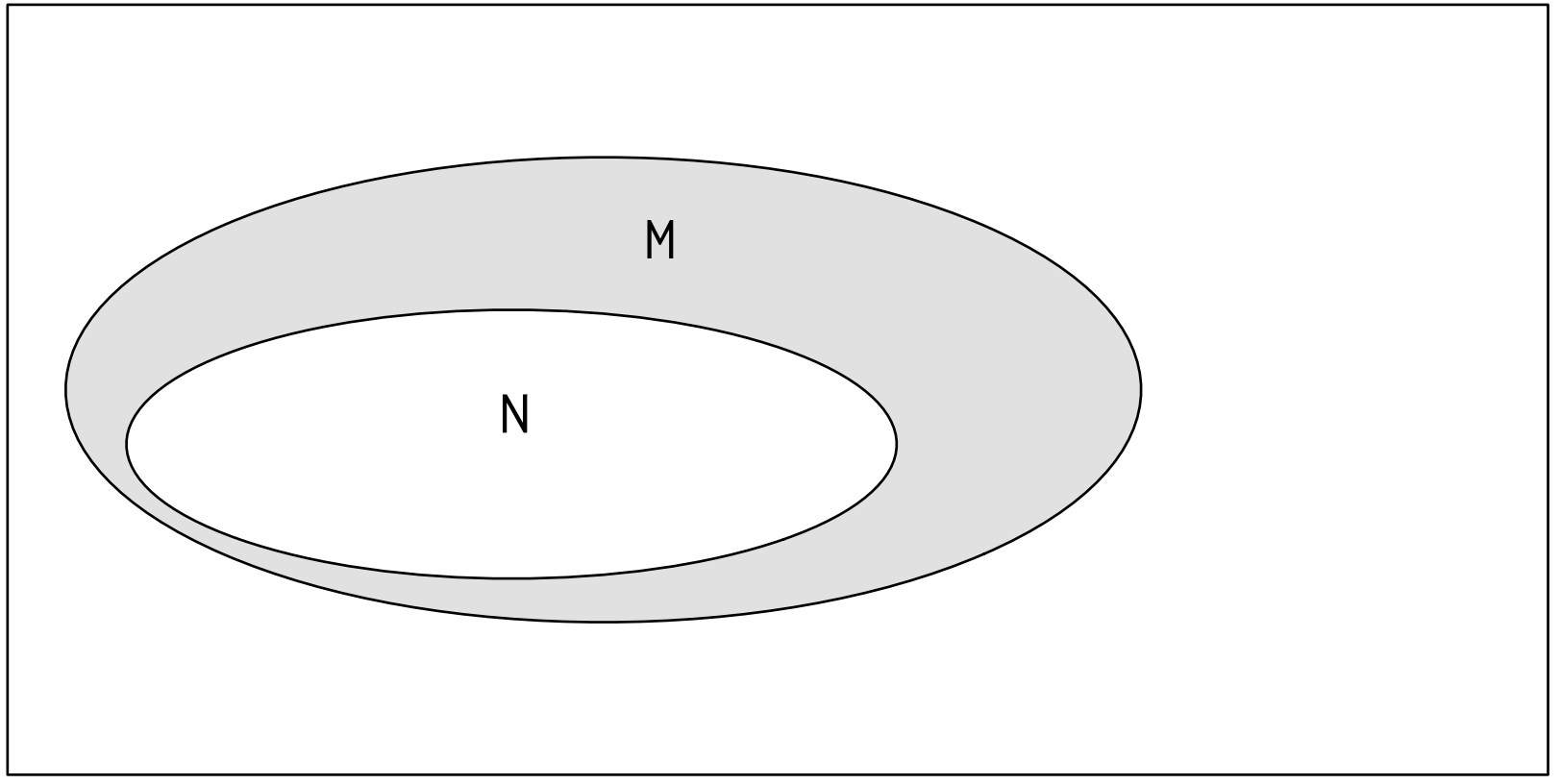
Allow | Forbid

The set of all executions



Finding **minimal** disallowed executions [LWP+17]

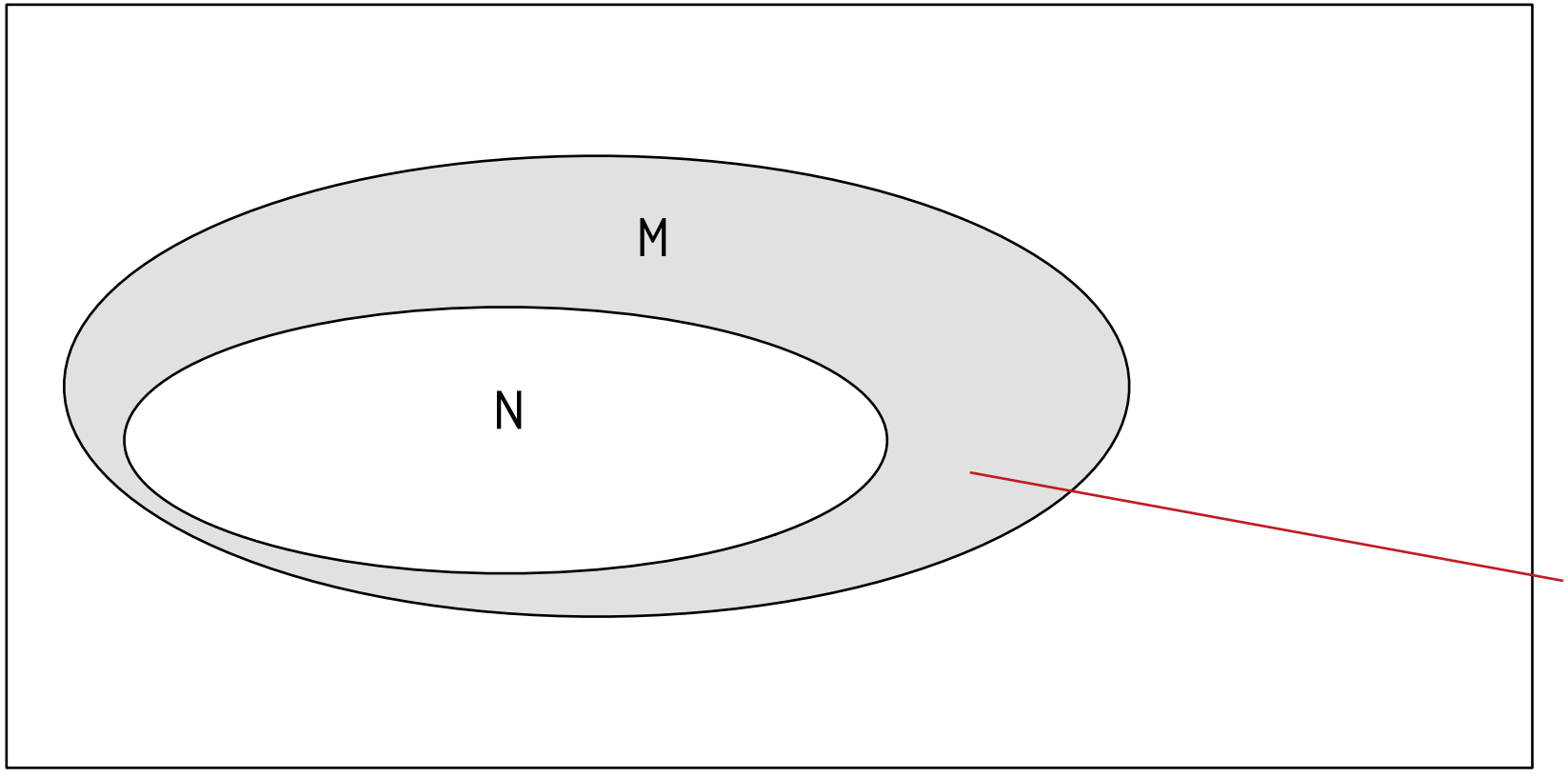
The set of all executions



Finding **minimal** disallowed executions [LWP+17]

N is stronger-than M

M is weaker-than N



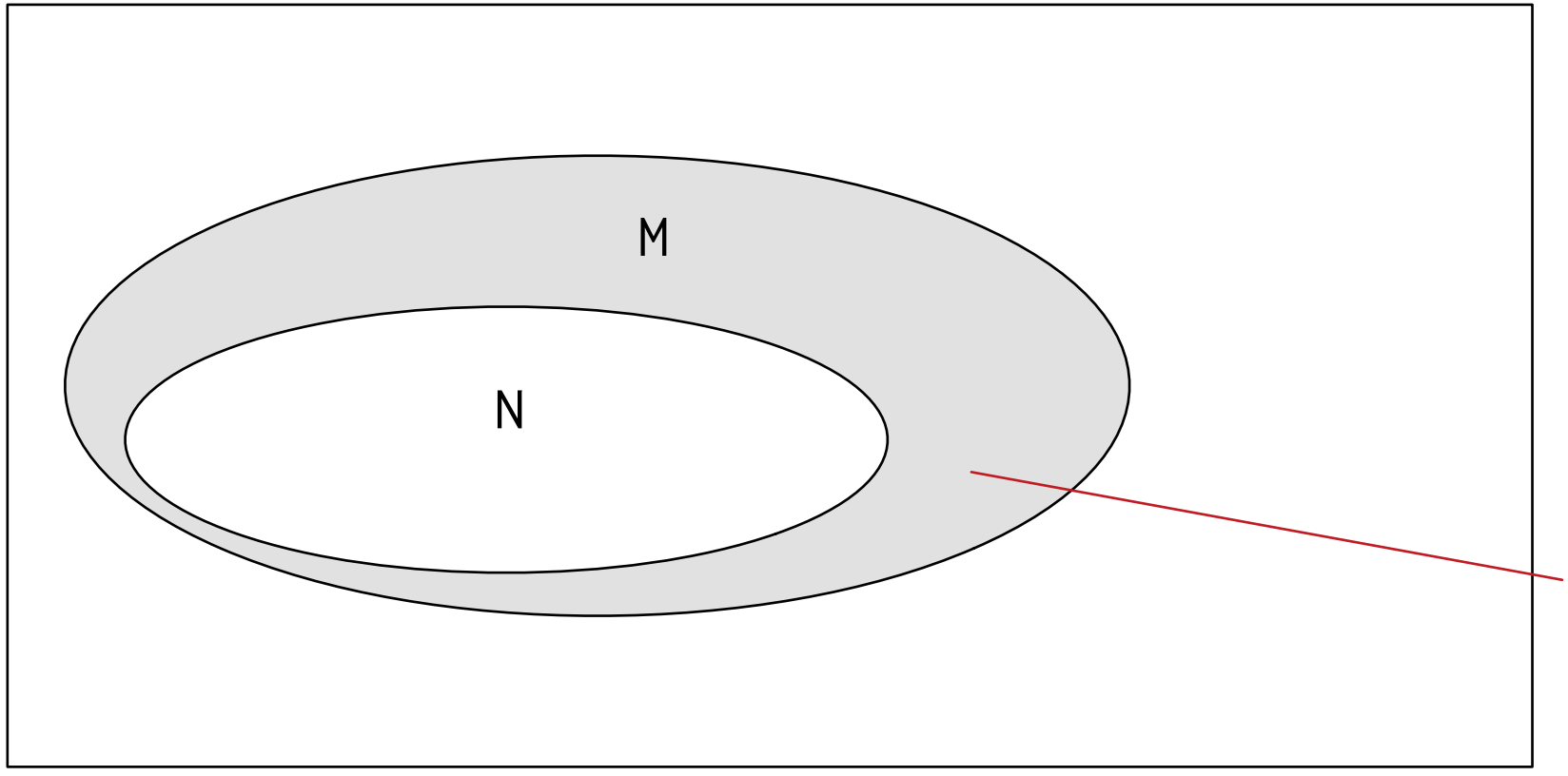
Finding **minimal** disallowed executions [LWP+17]

N is stronger-than M

M is weaker-than N

Finding a **distinguishing** execution in this set M\N [WBS+17]

The set of all executions



Finding **minimal** disallowed executions [LWP+17]

N is stronger-than M

M is weaker-than N

Finding a **distinguishing** execution in this set M\N [WBS+17]

E.g., Let M=x86 and N=SC
Then M\N includes the store-buffering execution

Combine [LWP+17, WBS+17]

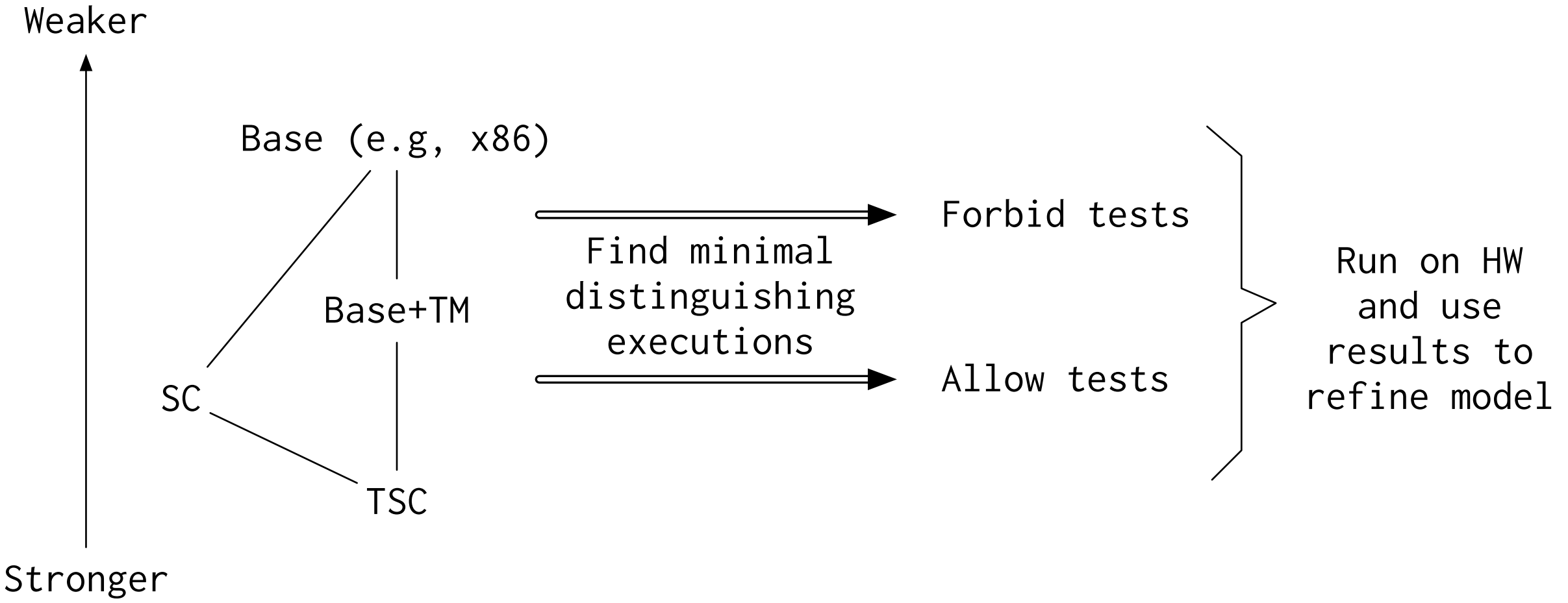


Table 2. Empirical testing of our transactional x86 model on Intel Haswell and Broadwell machines

$\|E\|$	**Forbid**				**Allow**			
	Solve (Sec)	T	S	¬S	Solve (Sec)	T	S	¬S
2	1	2	0	2	1	0	0	0
3	2	6	0	6	1	0	0	0
4	6	26	0	26	3	6	2	4
5	191	45	0	45	47	10	4	6
6	3600*	167	0	167	3600*	38	16	22
7	3600*	372	0	372	3600*	79	4	75
8	3600*	514	0	514	3600*	124	4	120
9	3600*	37	0	37	3600*	21	0	21
10	3600*	9	0	9	3600*	4	0	4
Sum		1178	0	1178		282	30	252

Results

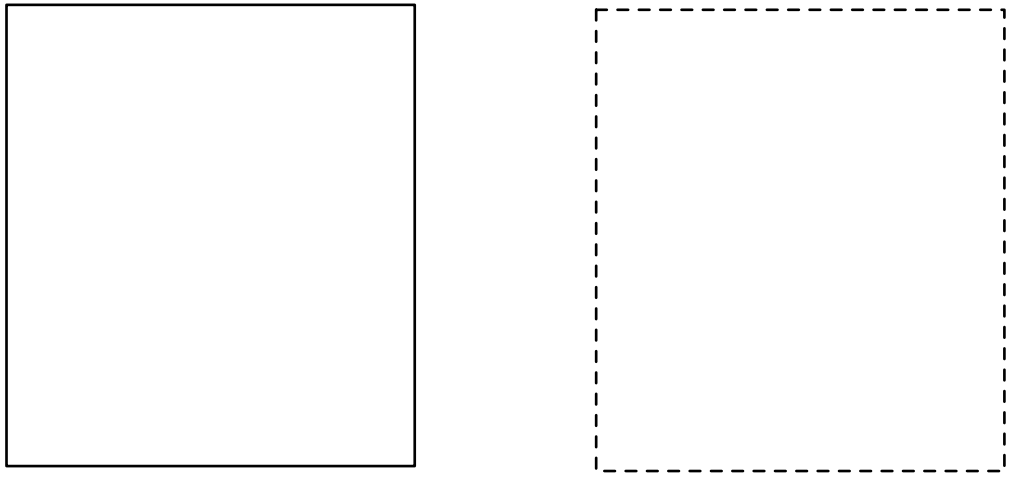
- Experimentally validated x86 TSX and Power TM models

- Proposals for Armv8 and C++ TM extensions

- Small additions to each model
  - Strong isolation
  - Transaction ordering (including implicit barriers)
  - Transaction propagation (Power-only)

- **Methodology transferred to architecture-validation team in Arm**

Failing Transactions



`successful`

`failing`

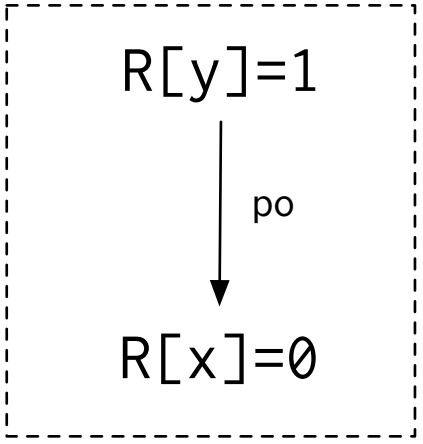
```
  TXSTART fail
  // …
  // Body
  // …
  TXCOMMIT
```

fail => rollback state
and branch to handler

```
fail:
  // …
  // Fail handler
  // …
```

Essential Problem

Abort causes state rollback: how do we get visibility inside a failing tx?



```
  TXSTART fail
  // …
  // Body
  // …
  TXCOMMIT
```

fail => rollback state
and branch to handler

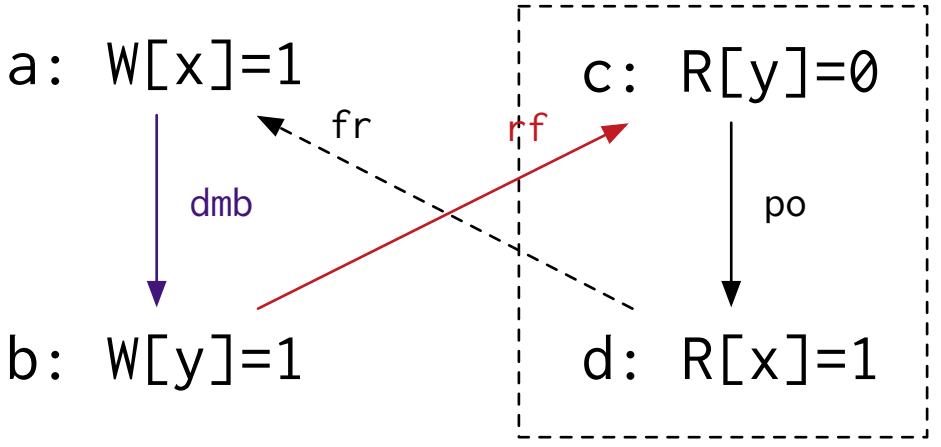
```
fail:
  // …
  // Fail handler
  // …
```

```
  TXSTART fail
  // …
  // Body
  // …
  TXABORT #VAL
```

fail => rollback state
and branch to handler

```
fail:
  // …
  // Fail handler (TXSTATUS.reason==VAL)
  // …
```

Failing Transactions




```
  TXSTART fail
  LDR W0, [X1] // c
  LDR W2, [X3] // d
  // if W0==0 && W2==1
  TXABORT #1
  // else
  TXABORT #0
fail:
  // TXSTATUS.reason==1
```

Future Work

- Transactional lock elision correctness

- Specifying TM operationally

- Fairness and forward-progress

- Interaction with PTW, exceptions, ...

- What about Opacity?

Reflection

- Automatic generation of minimal conformance testsuites
  - Minimality (close to the boundary)
  - Distinguishing
  - Automated
  - Output is very understandable

- Value of not-observing a forbidden test?
- Value of not-observing an allowed test?

Finally

We're hiring!

The security group is interested in the design, implementation and application of testing and verification at all levels of the system stack

Senior Formal Verification Researcher
Specifying and verifying real-world systems
www.arm.com/careers (search: 10720)

Alastair Reid

References

[AMT14] Herding cats: Modelling, Simulation, Testing, and Data-mining for Weak Memory (Alglave, Maranget, Tautschnig)
[CMF+13] Robust Architectural Support for Transactional Memory in the Power Architecture (Cain et al.)
[Deacon17] ARMv8.0 Application Level Memory Model (Deacon)
[DS09] Strong Isolation is a Weak Idea (Dalessandro, Scott)
[HM93] Transactional Memory: Architectural Support for Lock-Free Data Structures (Herlihy, Moss)
[LWP+17] Automatic Synthesis of Comprehensive Memory Model Litmus Test Suites (Lustig et al.)
[MBL06] Subtleties of Transactional Memory Atomicity Semantics (Martin, Blundell, Lewis)
[WBS+17] Automatically Comparing Memory Consistency Models (Wickerson et al.)

Reasoning about Transactional Memory

Transitioning ideas from academia to industry

Nathan Chong, John Wickerson and Tyler Sorensen

FMATS-5

21 September 2017