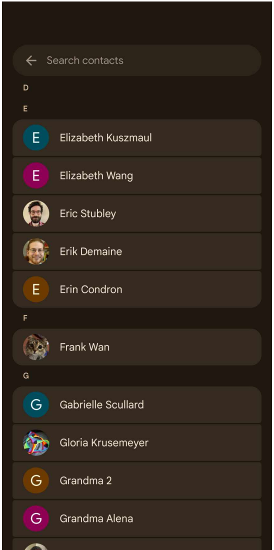
History Independent Data Structures

History Independence: "If an adversary were to see the state of the data structure, they would learn only the current set of elements, and nothing else about the history of past operations."

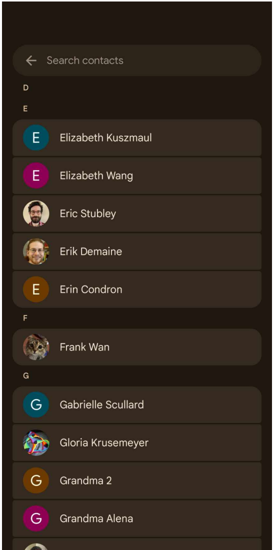
[Micciancio '97], [Naor, Teague '01]

History vs Content



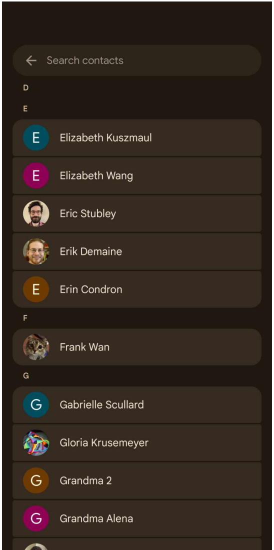
- If someone hacks my phone, they can learn my contacts list.

History vs Content



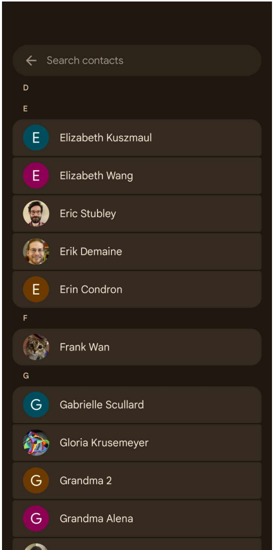
- If someone hacks my phone, they can learn my contacts list.
- But can they learn who my contacts were in the past?

History vs Content



- If someone hacks my phone, they can learn my contacts list.
- But can they learn who my contacts were in the past?
- What about the order in which contacts were added?

History vs Content



- If someone hacks my phone, they can learn my contacts list.
- But can they learn who my contacts were in the past?
- What about the order in which contacts were added?
- A history independent data structure protects this kind of information.

History Independent is a Security Guarantee

History Independence: "If an adversary were to see the state of the data structure, they would learn only the current set of elements, and nothing else about the history of past operations."

[Micciancio '97], [Naor, Teague '01]

History Independent is a Security Guarantee

History Independence: "If an adversary were to see the state of the data structure, they would learn only the current set of elements, and nothing else about the history of past operations."

[Micciancio '97], [Naor, Teague '01]

Lost of successes: Hash tables, trees, memory allocation, PMAs, graph algorithms, B-trees, cache-oblivious data structures... [Micciancio '97], [Naor, Teague '01], [Buchbinder, Petrank '03], [Molnar, Kohno, Sastry, Wagner '06], [Blelloch, Golovin '07], [Moran, Naor, Segev '07] [Naor, Segev, Wieder '08], [Golovin '08 '09 '10], [Tzouramanis '12], [Bajaj, Sion '13] [Bajaj, Chakrabati, Sion '15], [Roche, Aviv, Choi '15], [Bender, Berry, Johnson, Kroeger, McCauley, Phillips, Simon, Singh, Zage '16], ...

History Independent is a Security Guarantee

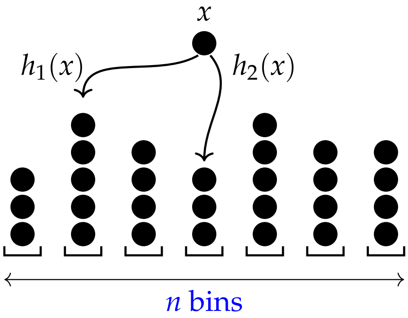
History Independence: “If an adversary were to see the state of the data structure, they would learn only the current set of elements, and nothing else about the history of past operations.”

[Micciancio ’97], [Naor, Teague ’01]

Lost of successes: Hash tables, trees, memory allocation, PMAs, graph algorithms, B-trees, cache-oblivious data structures... [Micciancio ’97], [Naor, Teague ’01], [Buchbinder, Petrank ’03], [Molnar, Kohno, Sastry, Wagner ’06], [Blelloch, Golovin ’07], [Moran, Naor, Segev ’07] [Naor, Segev, Wieder ’08], [Golovin ’08 ’09 ’10], [Tzouramanis ’12], [Bajaj, Sion ’13] [Bajaj, Chakrabati, Sion ’15], [Roche, Aviv, Choi ’15], [Bender, Berry, Johnson, Kroeger, McCauley, Phillips, Simon, Singh, Zage ’16], ...

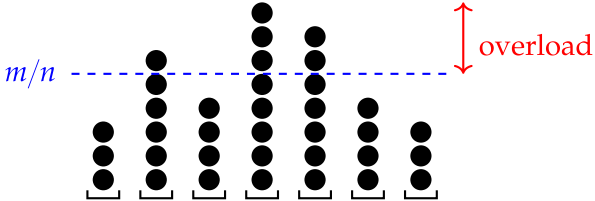
But... some very basic questions also remain open.

Two-Choice Load Balancing



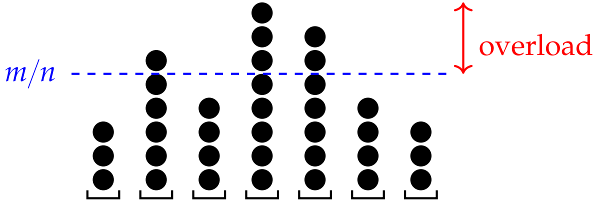
- Balls are inserted/deleted, with up to m present at a time.
- Each ball has two random bins where it can go.
- We must maintain a valid assignment of balls to bins.

Two Goals



Minimize Overload:
The amount by which the fullest bin exceeds m/n is small.

Two Goals



Minimize Overload:
The amount by which the fullest bin exceeds m/n is small.

Minimize Recourse:
On any given insertion/deletion, the number of balls moved around is small.

This Paper

Question: Does there exist a history-independent solution with small recourse and overload?

This Paper

Question: Does there exist a history-independent solution with small recourse and overload?

Theorem: There exists a history-independent solution with:

- Overload $O(1)$, with high probability.
- Expected recourse $O(\log \log(m/n))$.

What About Non-History-Independent Solutions?

Lots of work on the insertion-only case.
[Azar, Broder, Karlin and Upfal '94] [Berenbrink, Czumaj, Steger, and Vöcking '00][Dietzfelbinger and Weidling '07] [Frieze and Petti '18] . . .

What About Non-History-Independent Solutions?

Lots of work on the insertion-only case.
[Azar, Broder, Karlin and Upfal ’94] [Berenbrink, Czumaj, Steger, and Vöcking ’00][Dietzfelbinger and Weidling ’07] [Frieze and Petti ’18] . . .

But the fully dynamic case has remained largely open.
[Vöcking ’99] [Dietzfelbinger and Weidling ’07] [Bender, Conway, Farach-Colton, Kuszmaul, Tagliavini ’21] [Bansal, Kuszmaul ’22] . . .

What About Non-History-Independent Solutions?

Lots of work on the insertion-only case.
[Azar, Broder, Karlin and Upfal '94] [Berenbrink, Czumaj, Steger, and Vöcking '00][Dietzfelbinger and Weidling '07] [Frieze and Petti '18] . . .

But the fully dynamic case has remained largely open.
[Vöcking '99] [Dietzfelbinger and Weidling '07] [Bender, Conway, Farach-Colton, Kuszmaul, Tagliavini '21] [Bansal, Kuszmaul '22] . . .

Open Question:
Is there a fully dynamic solution with recourse $o(m/n)$ and overload $O(1)$?

What About Non-History-Independent Solutions?

Lots of work on the insertion-only case.
[Azar, Broder, Karlin and Upfal '94] [Berenbrink, Czumaj, Steger, and Vöcking '00][Dietzfelbinger and Weidling '07] [Frieze and Petti '18] ...

But the fully dynamic case has remained largely open.
[Vöcking '99] [Dietzfelbinger and Weidling '07] [Bender, Conway, Farach-Colton, Kuszmaul, Tagliavini '21] [Bansal, Kuszmaul '22] ...

Open Question:
Is there a fully dynamic solution with recourse $o(m/n)$ and overload $O(1)$?

Answer:
Yes! We get recourse $O(\log\log(m/n))$ and overload $O(1)$!

This Paper

Question: Does there exist a history-independent solution with small recourse and overload?

Theorem: There exists a history-independent solution with:

- Overload $O(1)$, with high probability.
- Expected recourse $O(\log \log(m/n))$.

Rest of Talk:

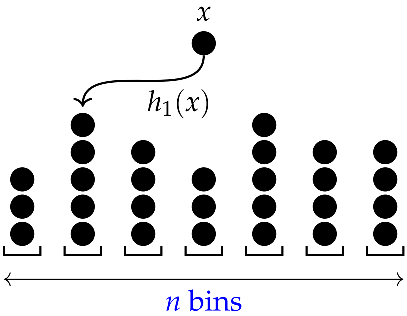
Outlining a Solution with

Overload $O(\log \log n)$

and Expected Recourse $O(m/n)$.

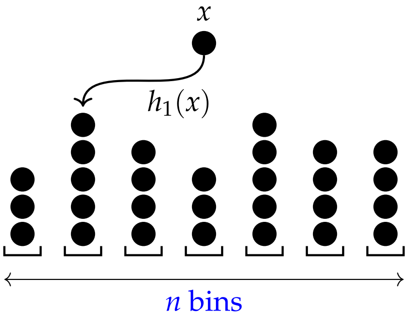
Warmup 1: The Single-Choice Strategy

To insert a ball x, just put it in bin $h_1(x)$:



Warmup 1: The Single-Choice Strategy

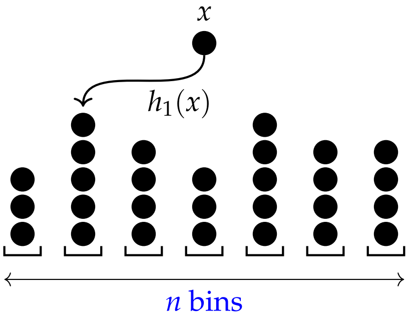
To insert a ball x, just put it in bin $h_1(x)$:



- This is history-independent ✓

Warmup 1: The Single-Choice Strategy

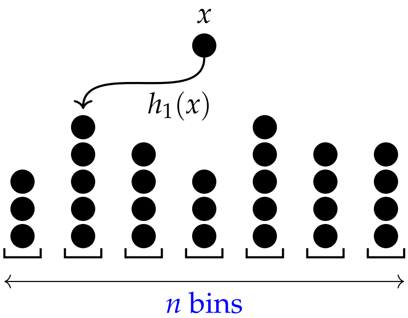
To insert a ball x, just put it in bin $h_1(x)$:



- This is history-independent ✓
- The recourse is 0 ✓

Warmup 1: The Single-Choice Strategy

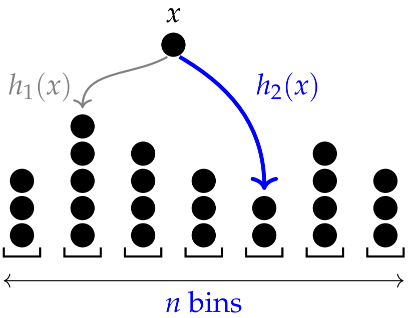
To insert a ball x, just put it in bin $h_1(x)$:



- This is history-independent ✓
- The recourse is 0 ✓
- But... the overload is huge, roughly $\sqrt{m/n}$ ✗

Warmup 2: Greedy Insertions

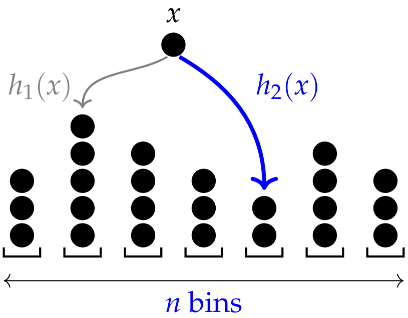
To insert a ball x, put it in the emptier of its choices:



- This is **not** history-independent ✗

Warmup 2: Greedy Insertions

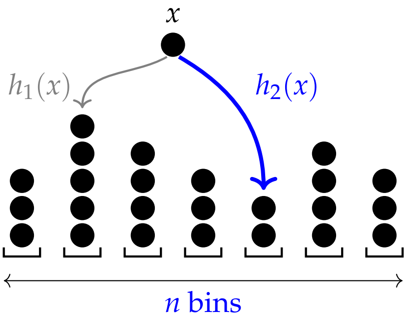
To insert a ball x, put it in the emptier of its choices:



- This is **not** history-independent ✗
- The recourse is 0 ✓

Warmup 2: Greedy Insertions

To insert a ball x, put it in the emptier of its choices:

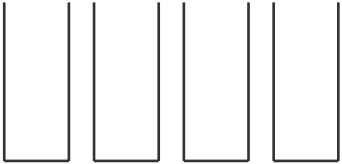


- This is **not** history-independent ✗
- The recourse is 0 ✓
- In the insertion-only case, the overload is $O(\log \log n)$ ✓

[Azar, Broder, Karlin and Upfal '94]

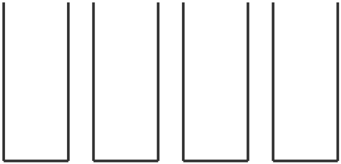
Turning Greedy into a History-Independent Solution

$S_0 = x_1, x_2, x_3, x_4, \ldots$



World 0

$S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$

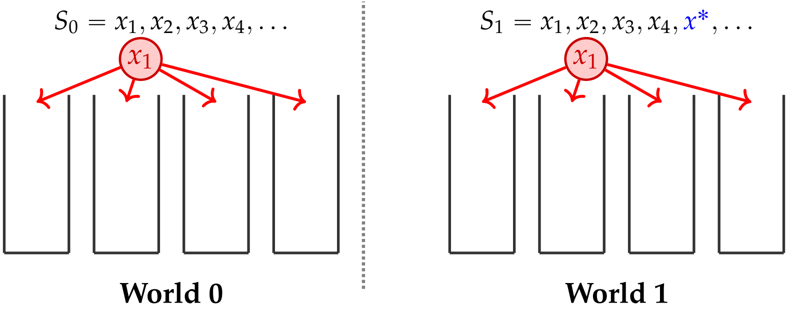


World 1

World 0: Insert balls from S_0

World 1: Insert balls from

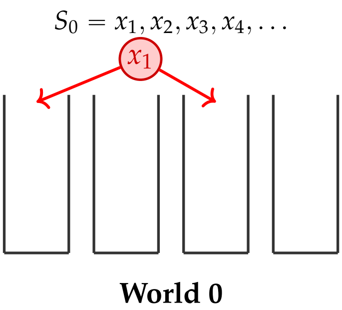
$S_1 = S_0 \cup \{x^*\}$



World 0: Insert balls from S_0

World 1: Insert balls from

$S_1 = S_0 \cup \{x^*\}$



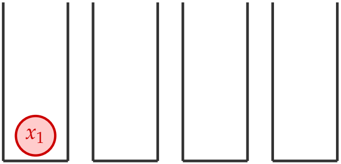
World 0: Insert balls from S_0

World 1: Insert balls from

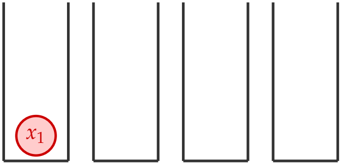
$S_1 = S_0 \cup \{x^*\}$

$S_0 = x_1, x_2, x_3, x_4, \ldots$

$S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$



World 0



World 1

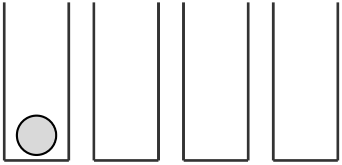
World 0: Insert balls from S_0

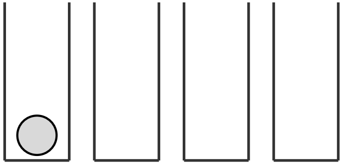
$S_1 = S_0 \cup \{x^*\}$

World 1: Insert balls from

$$S_0 = x_1, x_2, x_3, x_4, \ldots$$

$$S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$$

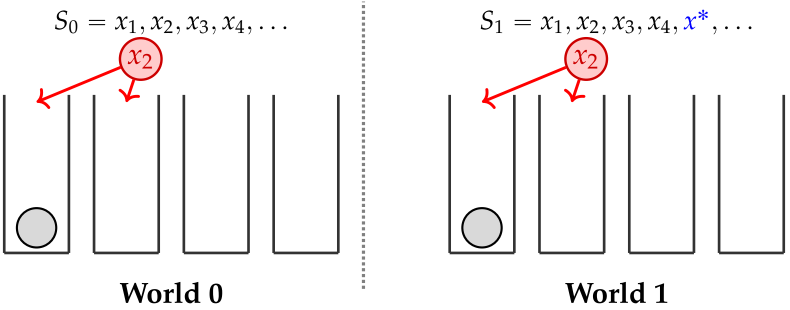




World 0: Insert balls from S_0

World 1: Insert balls from

$S_1 = S_0 \cup \{x^*\}$

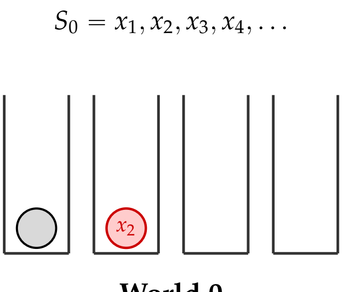


World 0: Insert balls from S_0

$S_1 = S_0 \cup \{x^*\}$

World 1: Insert balls from

$$S_0 = x_1, x_2, x_3, x_4, \ldots \qquad S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$$



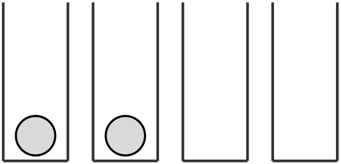
World 0 **World 1**

World 0: Insert balls from S_0

$S_1 = S_0 \cup \{x^*\}$

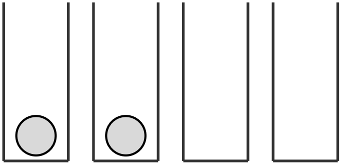
World 1: Insert balls from

$$S_0 = x_1, x_2, x_3, x_4, \ldots$$



World 0

$$S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$$

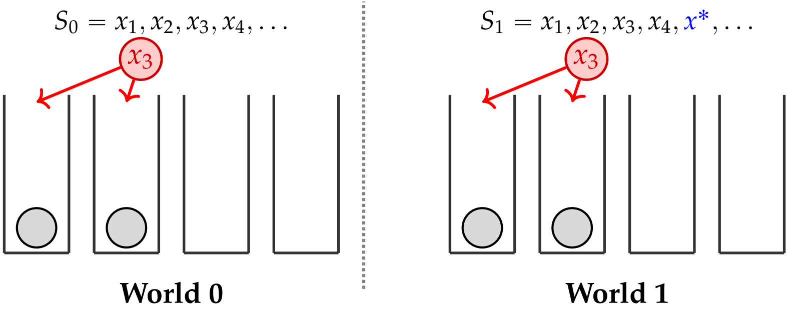


World 1

World 0: Insert balls from S_0

$S_1 = S_0 \cup \{x^*\}$

World 1: Insert balls from



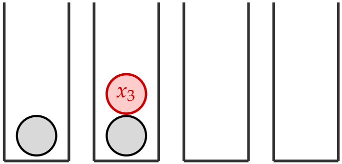
World 0: Insert balls from S_0

World 1: Insert balls from

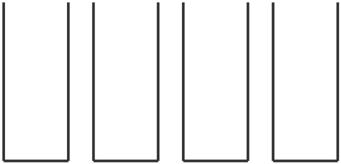
$S_1 = S_0 \cup \{x^*\}$

$S_0 = x_1, x_2, x_3, x_4, \ldots$

$S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$



World 0



World 1

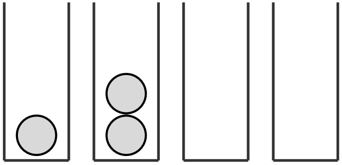
World 0: Insert balls from S_0

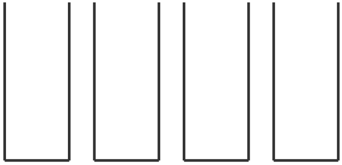
$S_1 = S_0 \cup \{x^*\}$

World 1: Insert balls from

$S_0 = x_1, x_2, x_3, x_4, \ldots$

$S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$

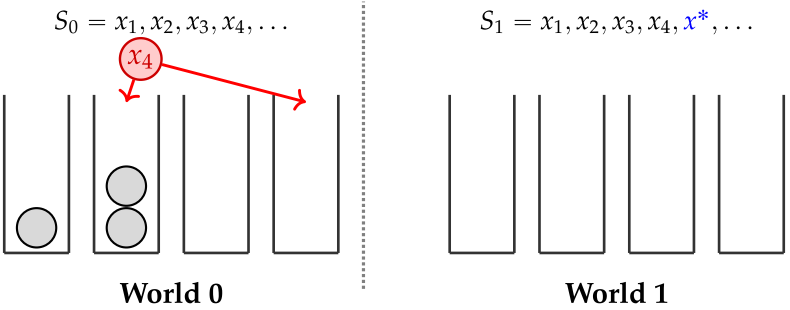




World 0: Insert balls from S_0

World 1: Insert balls from

$S_1 = S_0 \cup \{x^*\}$



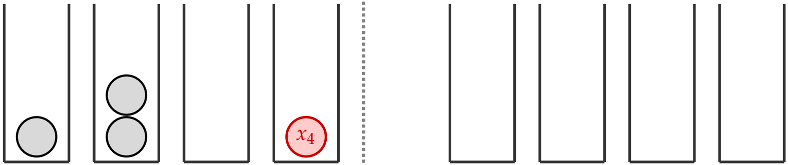
World 0: Insert balls from S_0

World 1: Insert balls from

$S_1 = S_0 \cup \{x^*\}$

$S_0 = x_1, x_2, x_3, x_4, \ldots$

$S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$



World 0

World 1

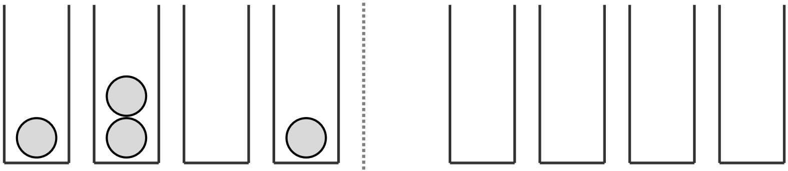
World 0: Insert balls from S_0

$S_1 = S_0 \cup \{x^*\}$

World 1: Insert balls from

$S_0 = x_1, x_2, x_3, x_4, \ldots$

$S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$



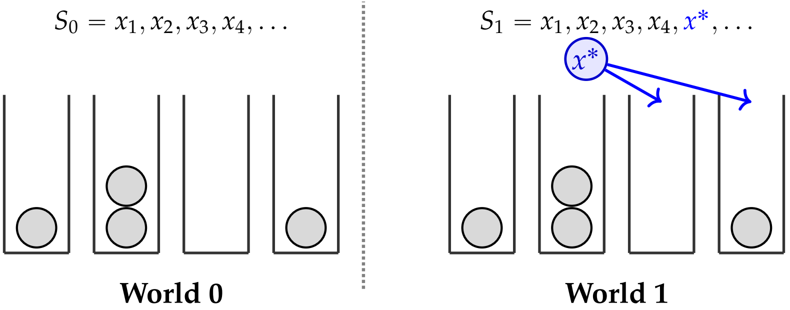
World 0

World 1

World 0: Insert balls from S_0

World 1: Insert balls from

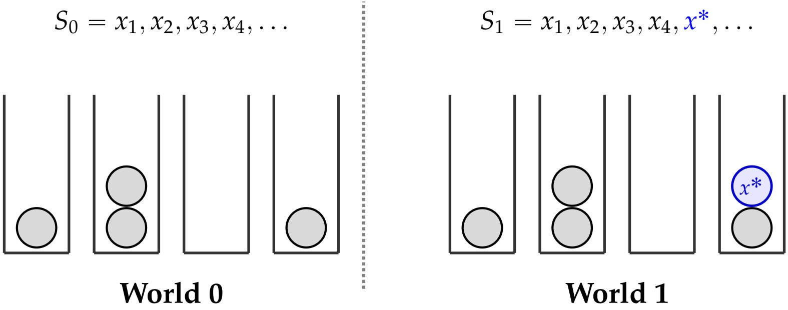
$S_1 = S_0 \cup \{x^*\}$



World 0: Insert balls from S_0

World 1: Insert balls from

$S_1 = S_0 \cup \{x^*\}$

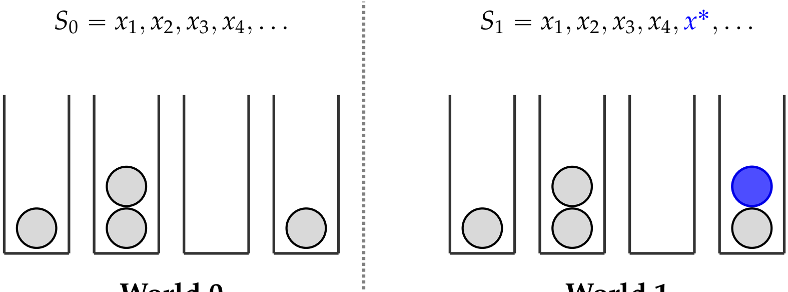


World 0: Insert balls from S_0

World 1: Insert balls from

$S_1 = S_0 \cup \{x^*\}$

$S_0 = x_1, x_2, x_3, x_4, \ldots$ | $S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$



World 0 | **World 1**

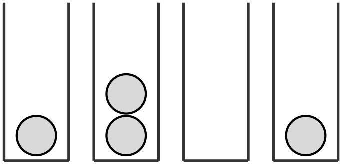
World 0: Insert balls from S_0

World 1: Insert balls from

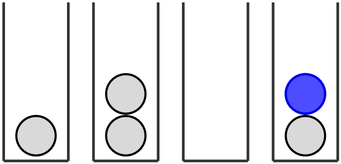
$S_1 = S_0 \cup \{x^*\}$

$S_0 = x_1, x_2, x_3, x_4, \ldots$

$S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$

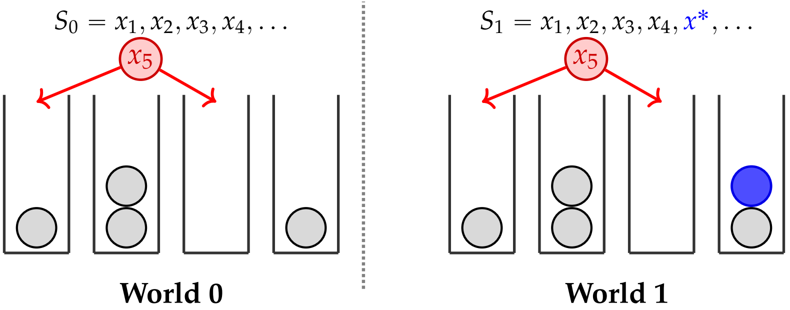


World 0



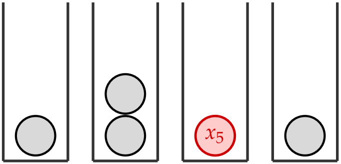
World 1

Future insertions: (1) No recourse



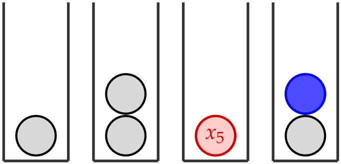
Future insertions: (1) No recourse

$$S_0 = x_1, x_2, x_3, x_4, \ldots$$



World 0

$$S_1 = x_1, x_2, x_3, x_4, {\color{blue}x^*}, \ldots$$

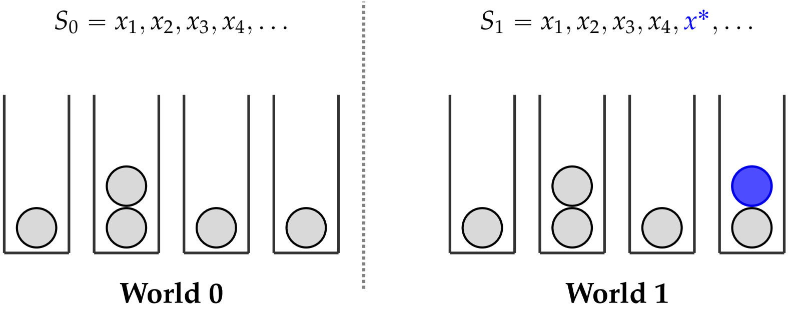


World 1

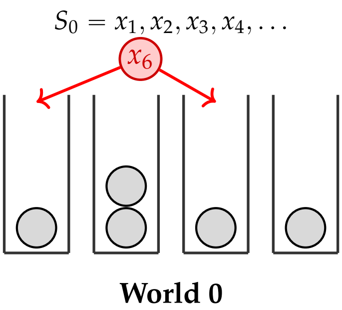
Future insertions: (1) No recourse

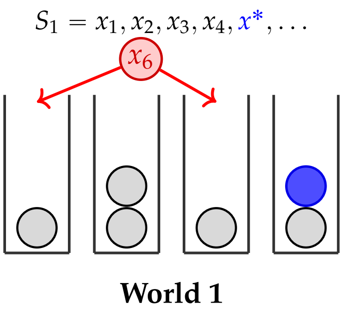
$S_0 = x_1, x_2, x_3, x_4, \ldots$

$S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$



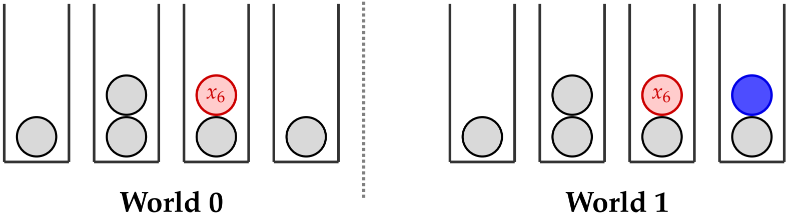
Future insertions: (1) No recourse





Future insertions: (1) No recourse

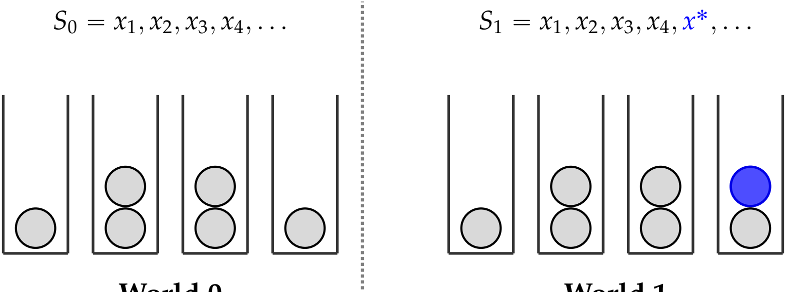
$S_0 = x_1, x_2, x_3, x_4, \ldots$ $\qquad$ $S_1 = x_1, x_2, x_3, x_4, {\color{blue}x^*}, \ldots$



Future insertions: (1) No recourse

$S_0 = x_1, x_2, x_3, x_4, \ldots$

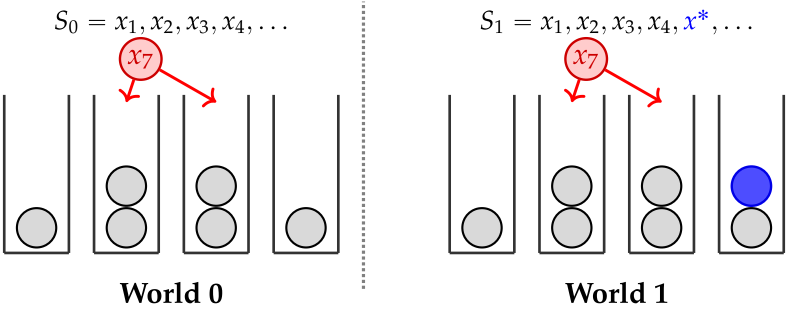
$S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$



World 0

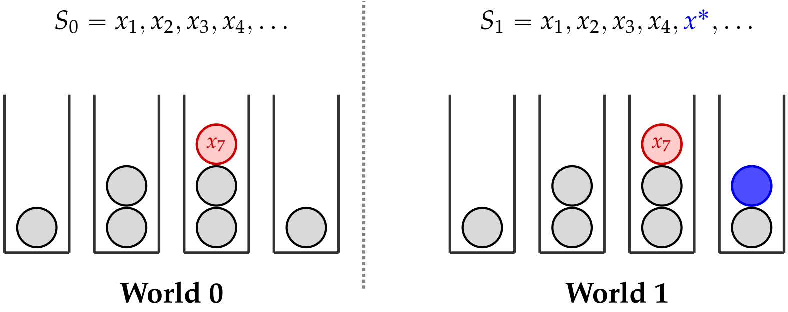
World 1

Future insertions: (1) No recourse



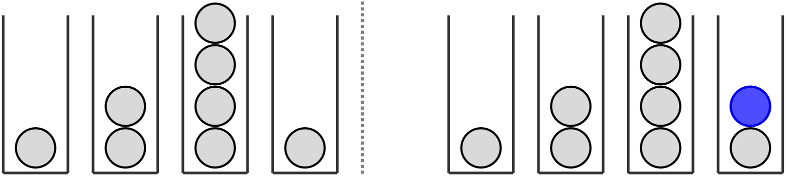
Future insertions: (1) No recourse

$S_0 = x_1, x_2, x_3, x_4, \ldots$ $S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$



Future insertions: (1) No recourse

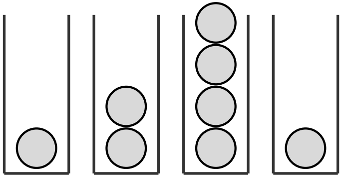
$S_0 = x_1, x_2, x_3, x_4, \ldots$ $S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$



World 0 **World 1**

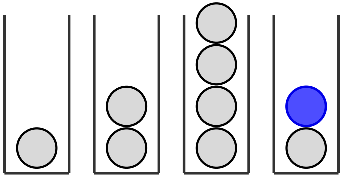
Future insertions: (1) No recourse

$S_0 = x_1, x_2, x_3, x_4, \ldots$



World 0

$S_1 = x_1, x_2, x_3, x_4, x^*, \ldots$



World 1

Future insertions: (1) No recourse (2) Recourse