Imperial College London

SOFTWARE RELIABILITY GROUP

ASE 2019

FINE-GRAINED MEMORY OBJECT REPRESENTATION IN SYMBOLIC EXECUTION

MARTIN NOWACK

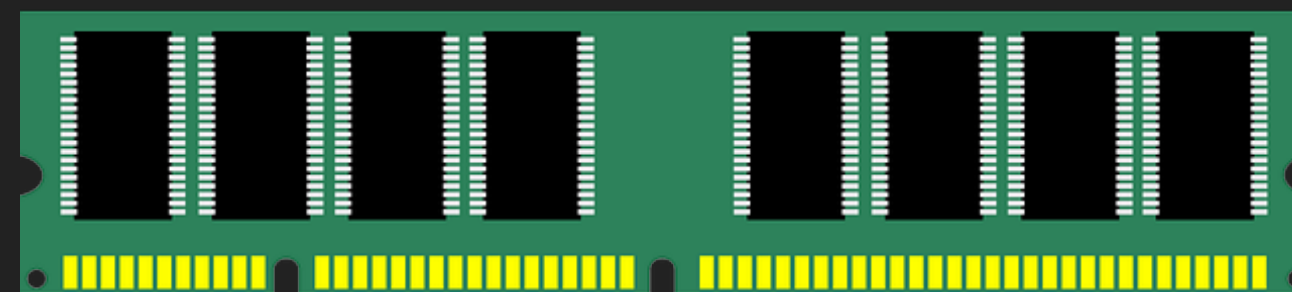
M.NOWACK@IMPERIAL.AC.UK

PROGRAMS

```c
char * a = malloc(1024);
int32 i = 10;

a[i]++;
if (i != 12345)
{
  a[i-2] = a[i] * 2;
} else {
  a[i+2] = a[i] - 2;
}
```

MEMORY REPRESENTATION



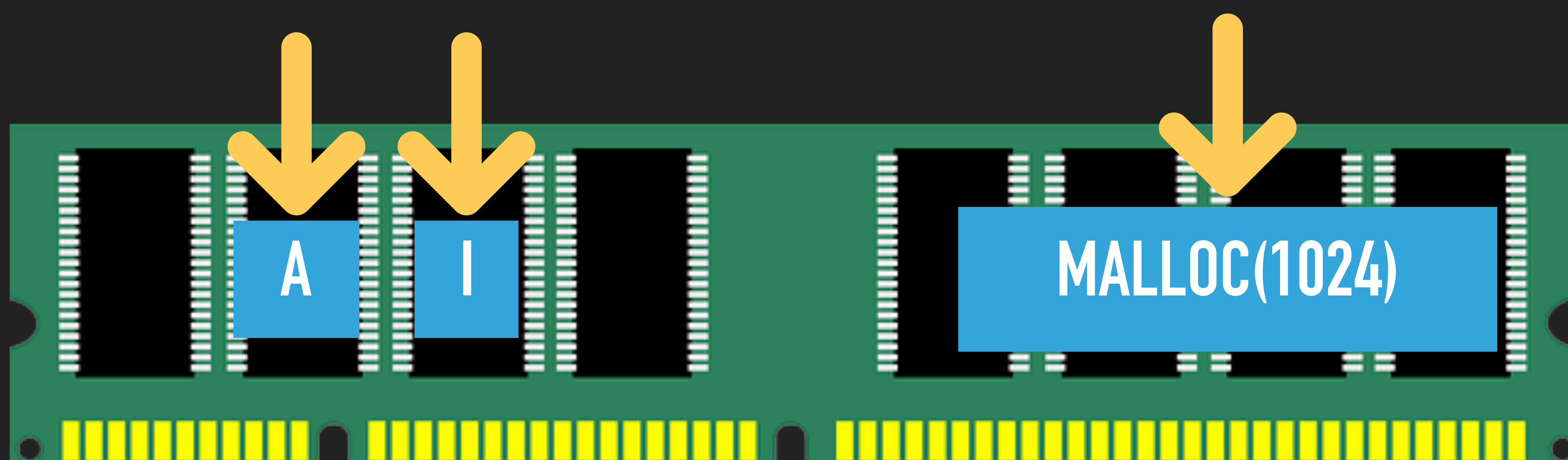
SYMBOLIC EXECUTION

```
char * a = malloc(1024);
int32 i = 10;

a[i]++;
if (i != 12345) {
  a[i-2] = a[i] * 2;
} else {
  a[i+2] = a[i] - 2;
}
```

```
char * a = malloc(1024);
int32 i = 10;

a[i]++;
if (i != 12345) {
  a[i-2] = a[i] * 2;
} else {
  a[i+2] = a[i] - 2;
}
```

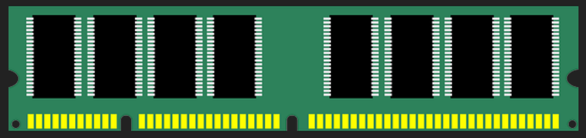


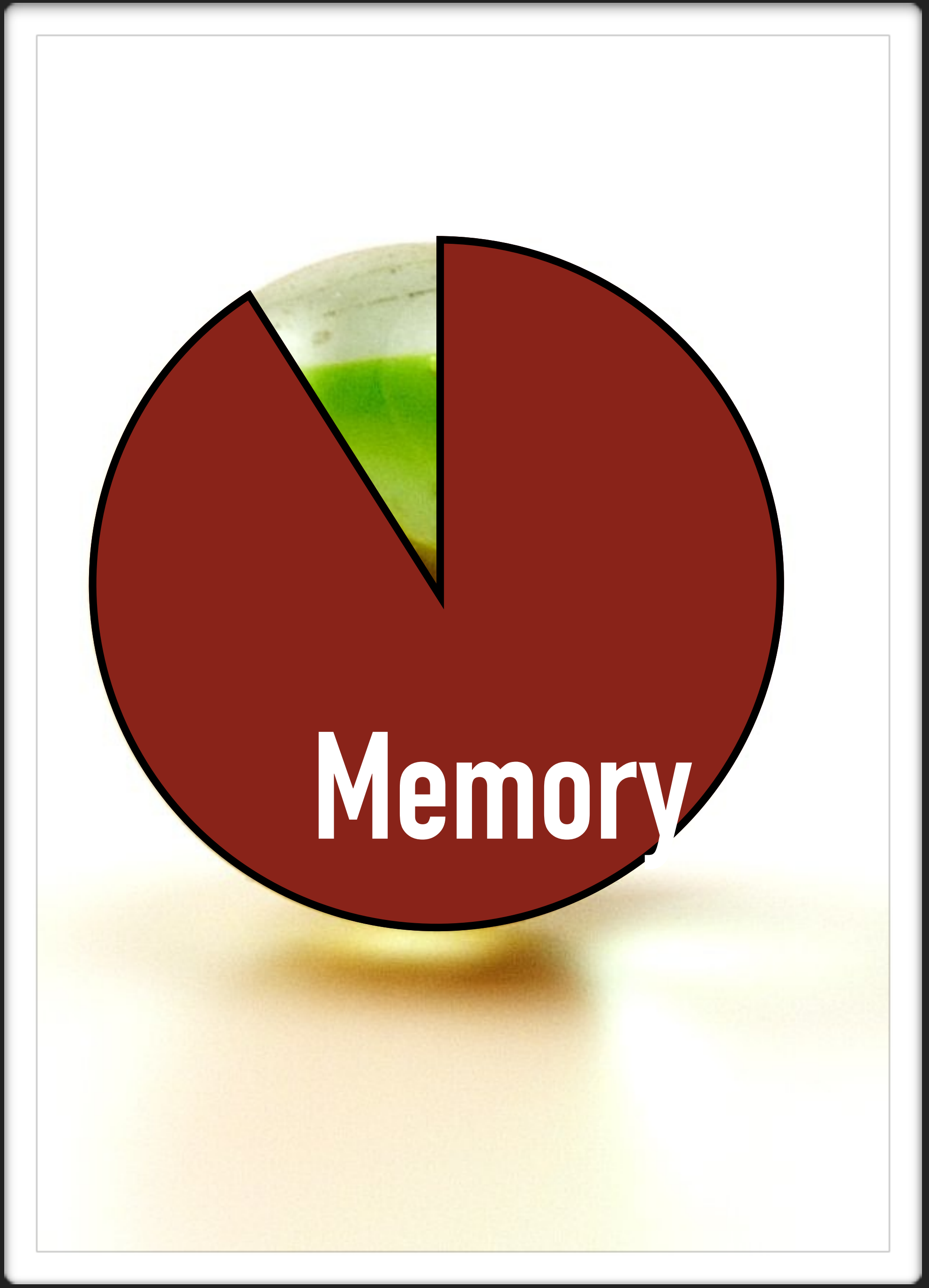
(DYNAMIC) SYMBOLIC EXECUTION

```
char * a = malloc(1024);
int32 i = symbolic;

a[i]++;
if (i != 12345)
{
  a[i-2] = a[i] * 2;
} else {
  a[i+2] = a[i] - 2;
}
```

STATE - A SIMPLIFIED VIEW

- Path Constraints
- Registers (i.e., program counter)
- Allocated Memory 
  - Stack-local
  - Heap



THE MANY STATES . . .

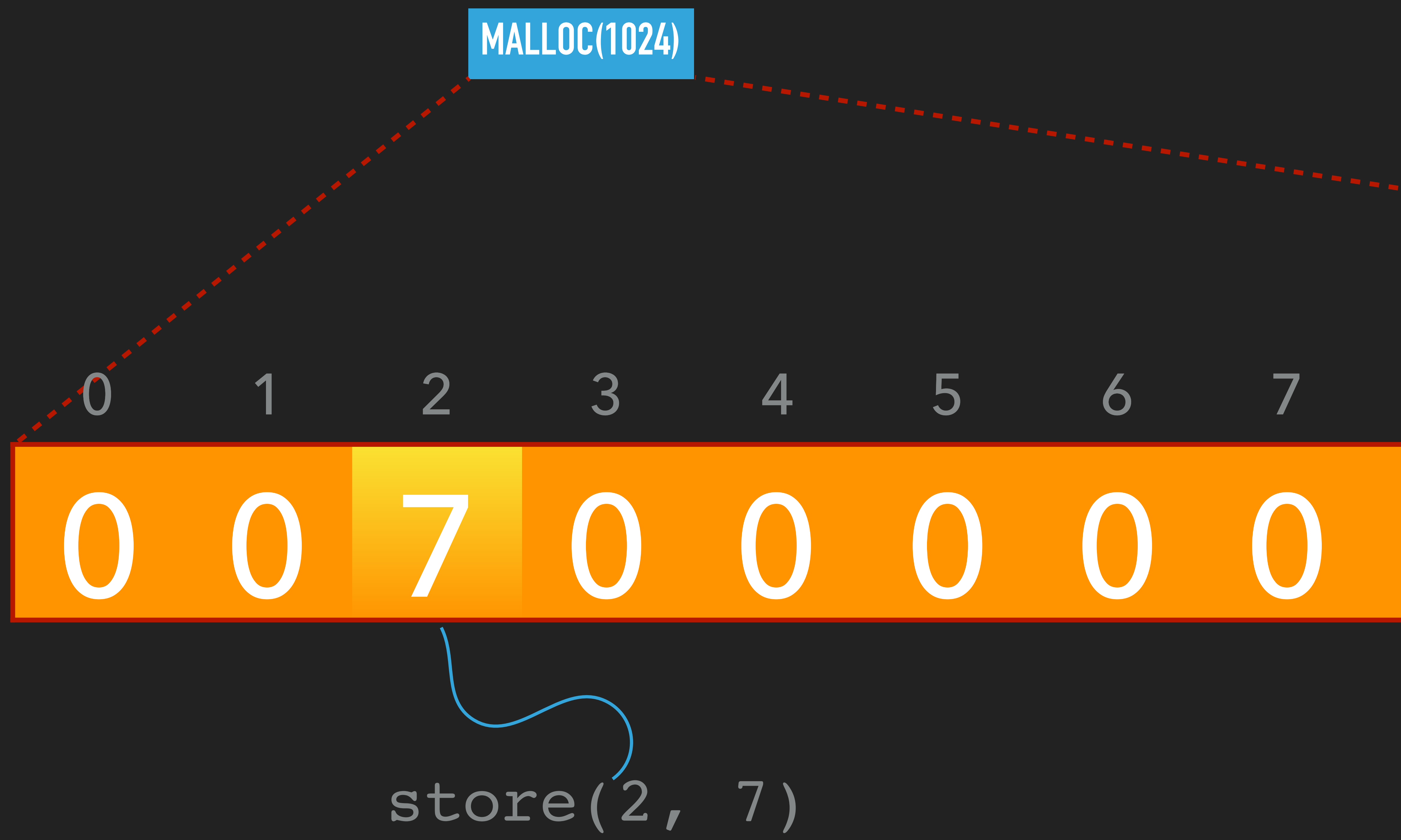
THE MANY STATES . . .

GOAL

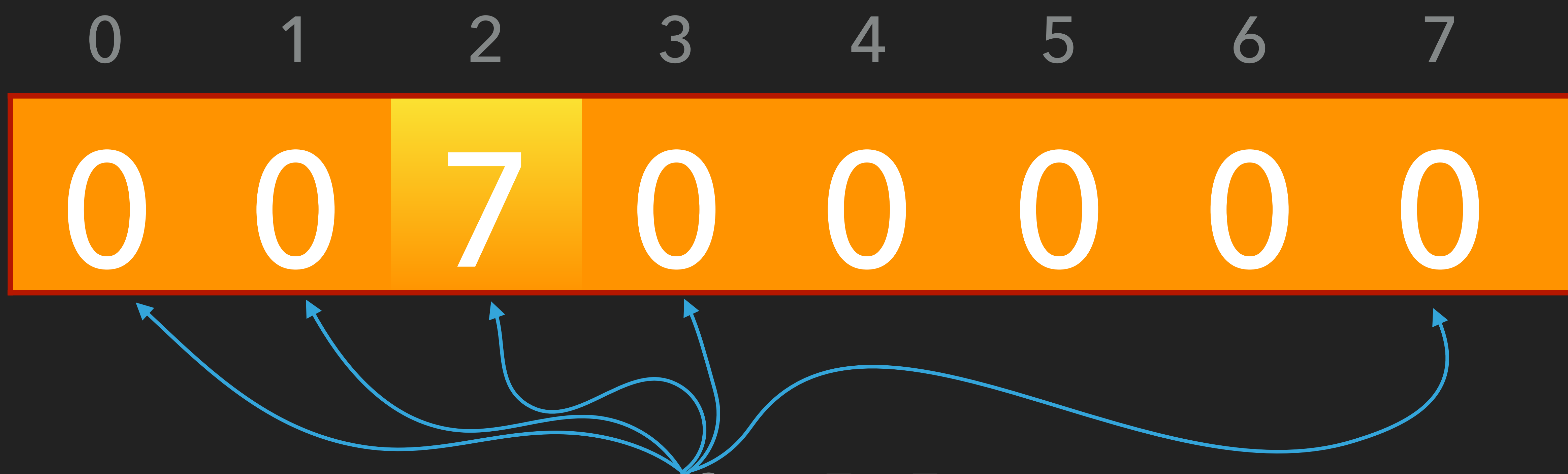
- Scale symbolic execution
- Avoid premature termination of states
- Sort/Reason about states

STATE OF THE ART

Copy on Write (CoW)


MALLOC(1024)
0 1 2 3 4 5 6 7
0 0 7 0 0 0 0 0
store(2, 7)

HANDLING SYMBOLICS



store(sym, 7)

load(5)

FINE-GRAINED MEMORY OBJECT REPRESENTATION

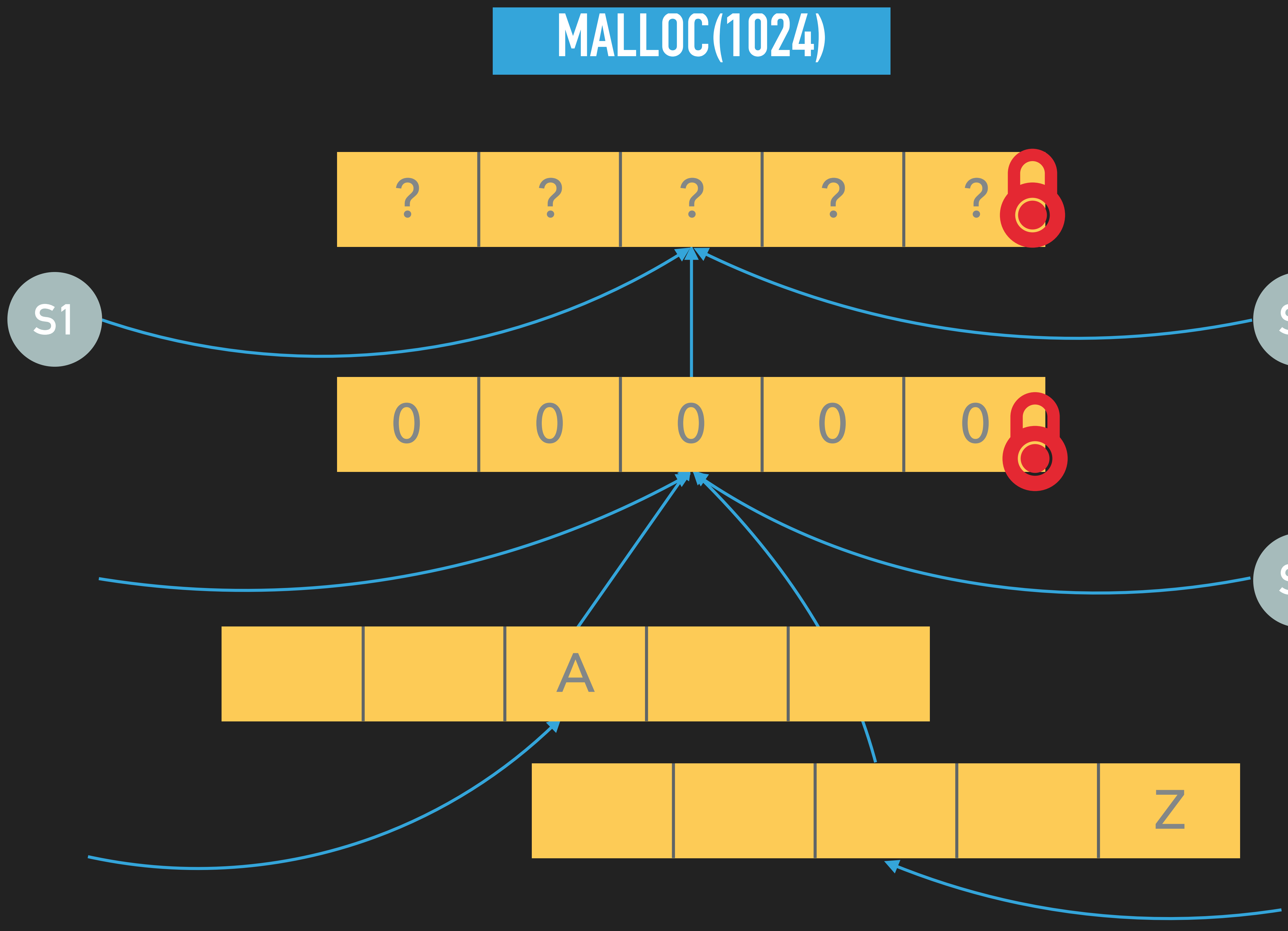
INSIGHT I:
CHANGES ARE (OFTEN) SMALL; SHARE COMMON PARTS

INSIGHT II:
CHANGES ARE (OFTEN) LOCAL AND OF SIMILAR TYPE

EVERYTHING IS A LAYER

BASICS

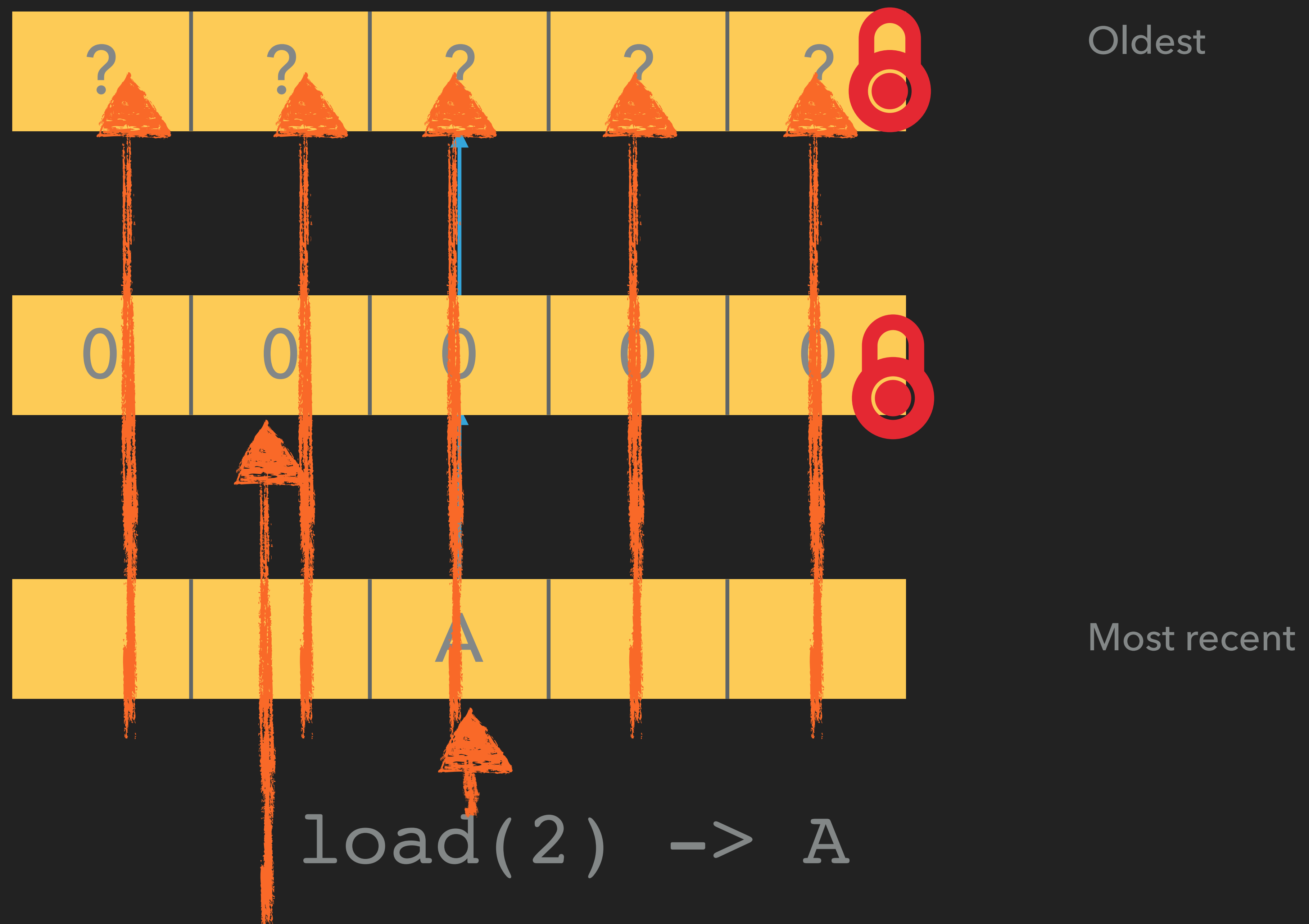
EXAMPLE SCENARIO



OPTIMISATIONS

INDEX-BASED ACCESS

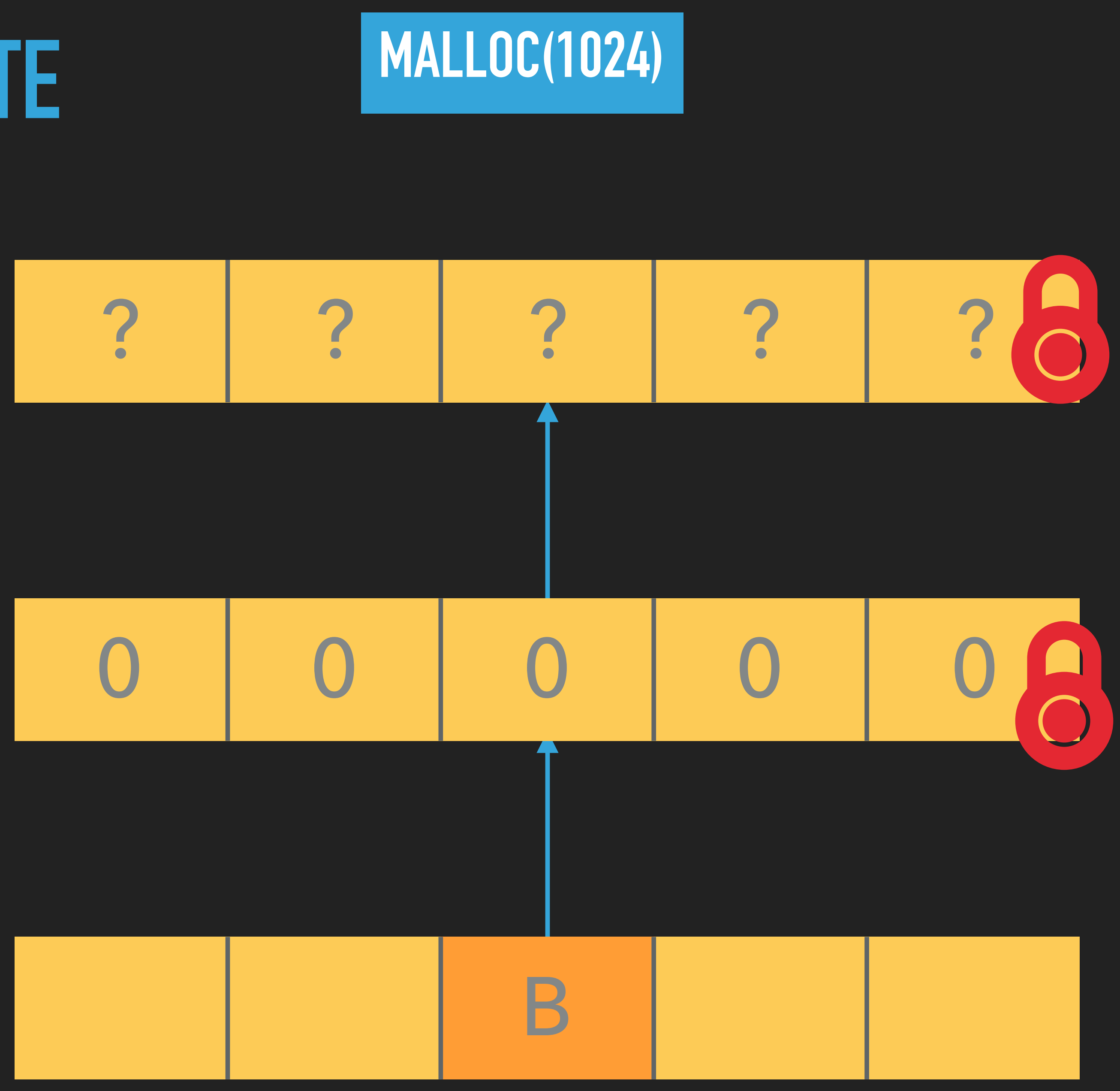
MALLOC(1024)



`load(2) -> A`

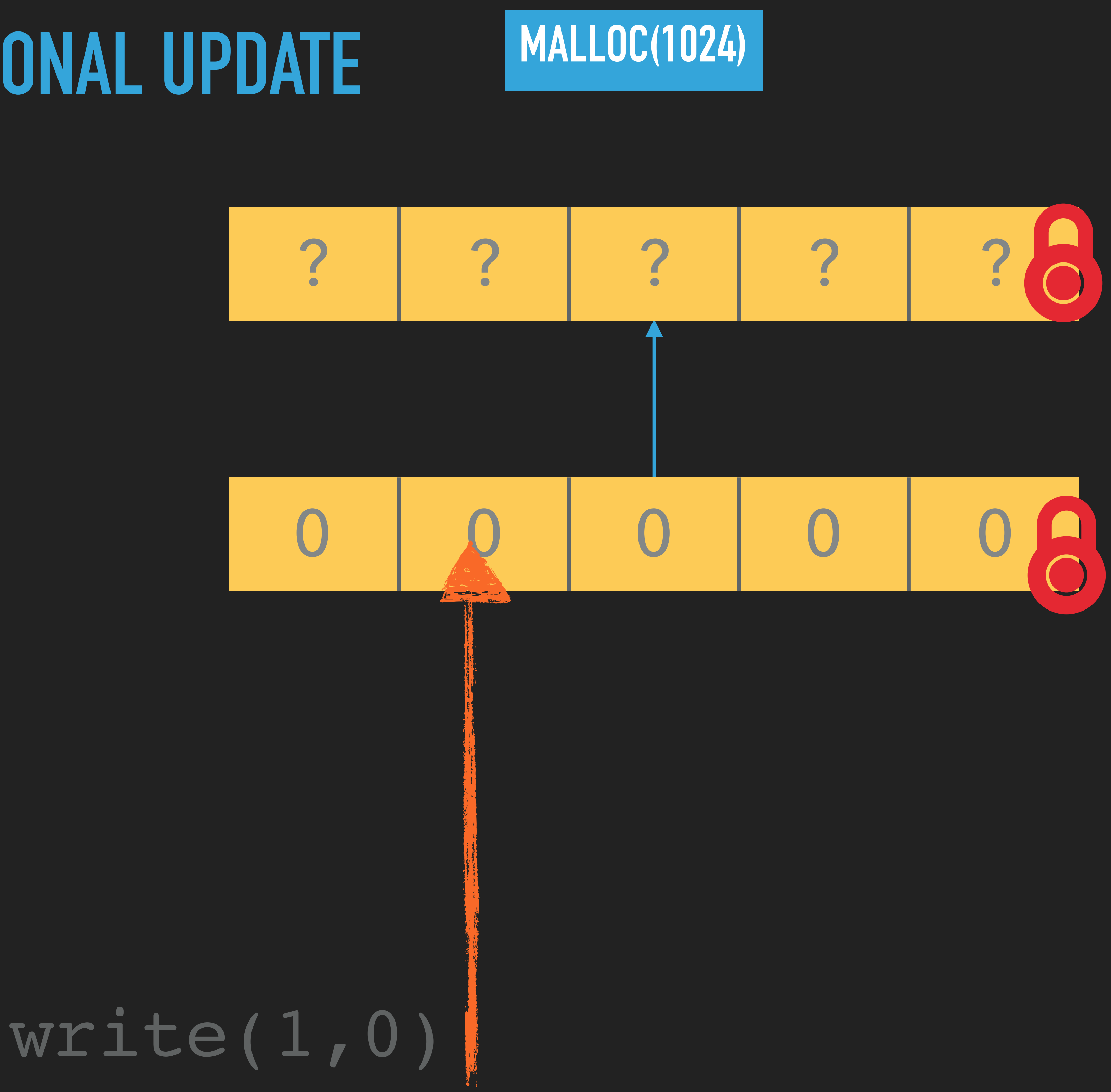
`load(1) -> 0`

IN-PLACE UPDATE



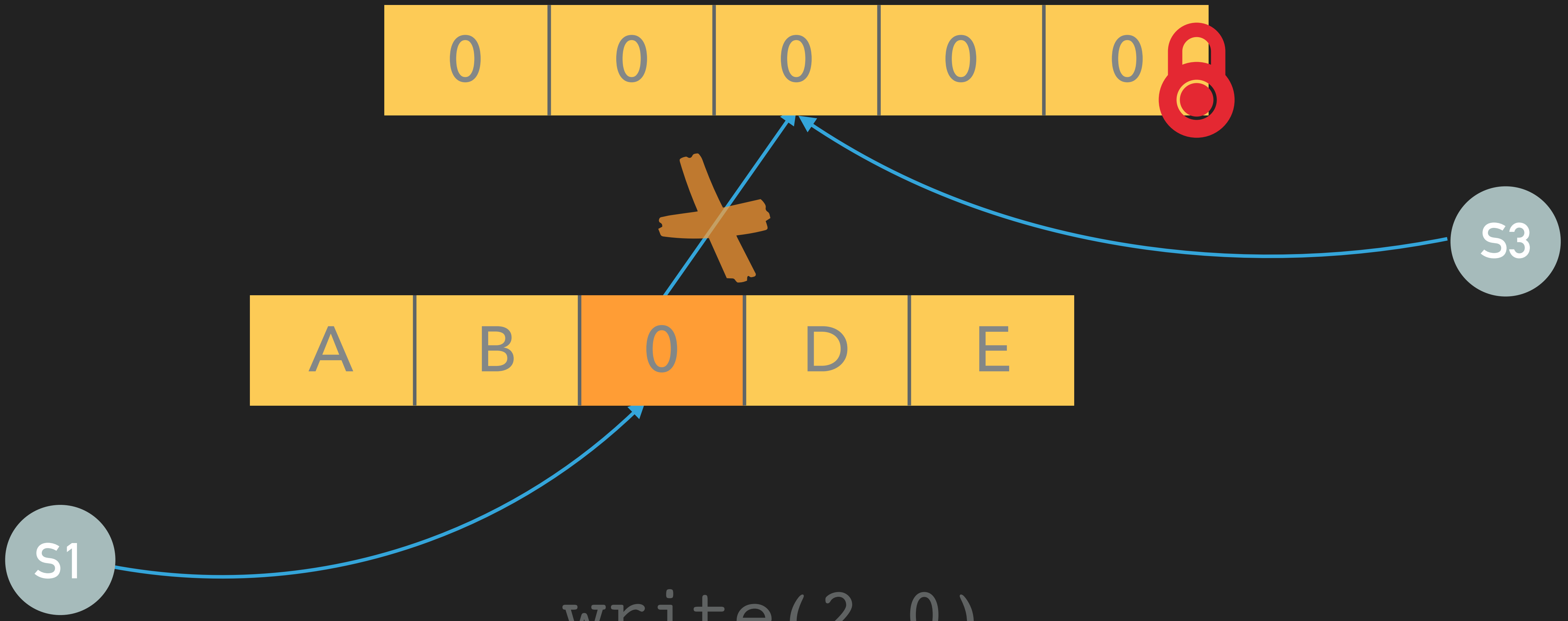
write(2,B)

CONDITIONAL UPDATE



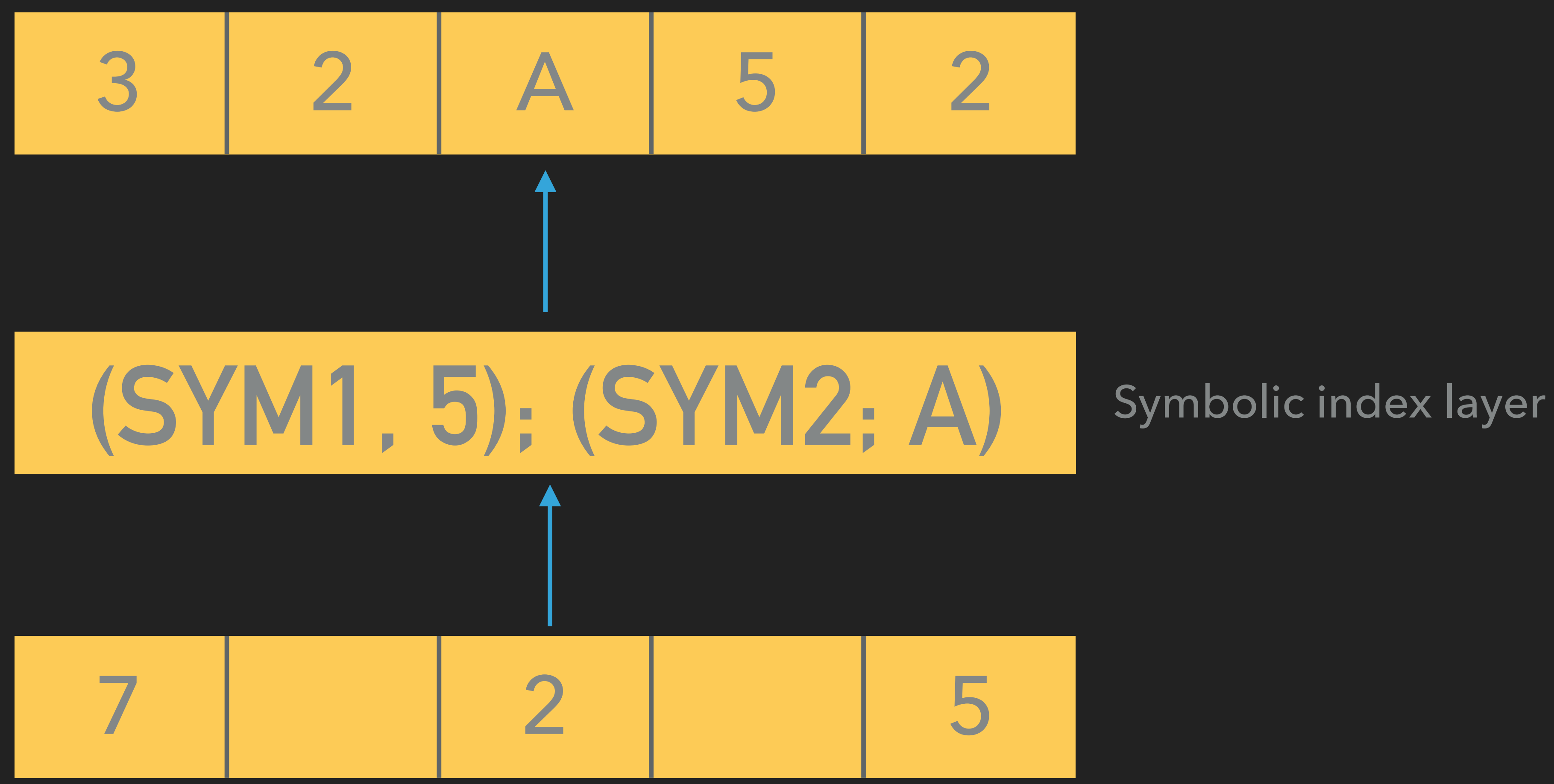
LAYER INVALIDATION

MALLOC(1024)



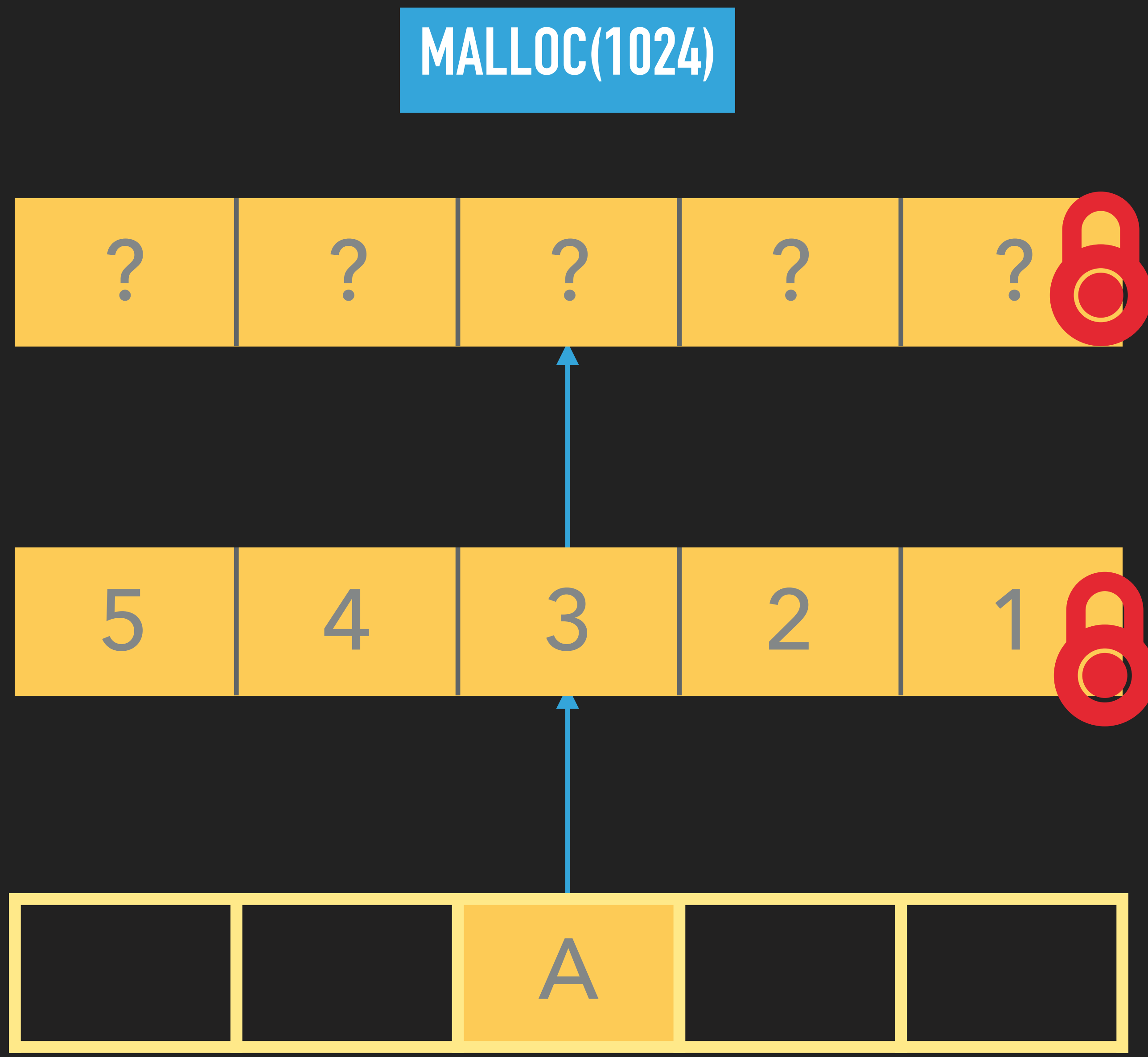
write(2,0)

HANDLING SYMBOLIC INDICES



IMPLEMENTATION

LAYER TYPES



Allocated Space

~ 10 byte

Initialised bytes
sizeof() * 1bit

Map: index -> value

EVALUATION

BENCHMARKS

vs.

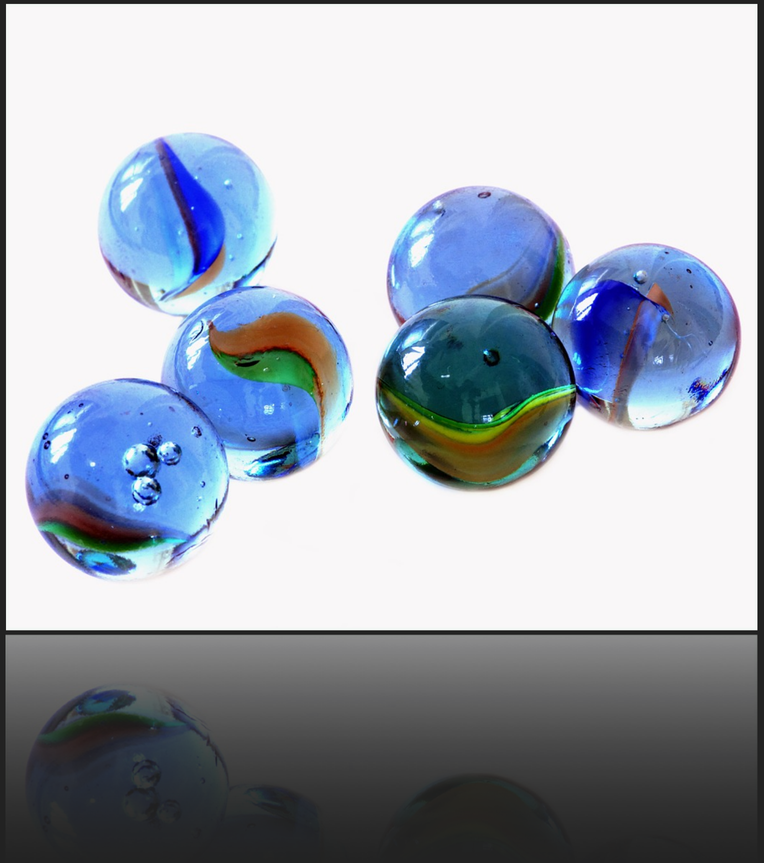
MEMORY

GNU Coreutils

Depth-First

Search Strategies

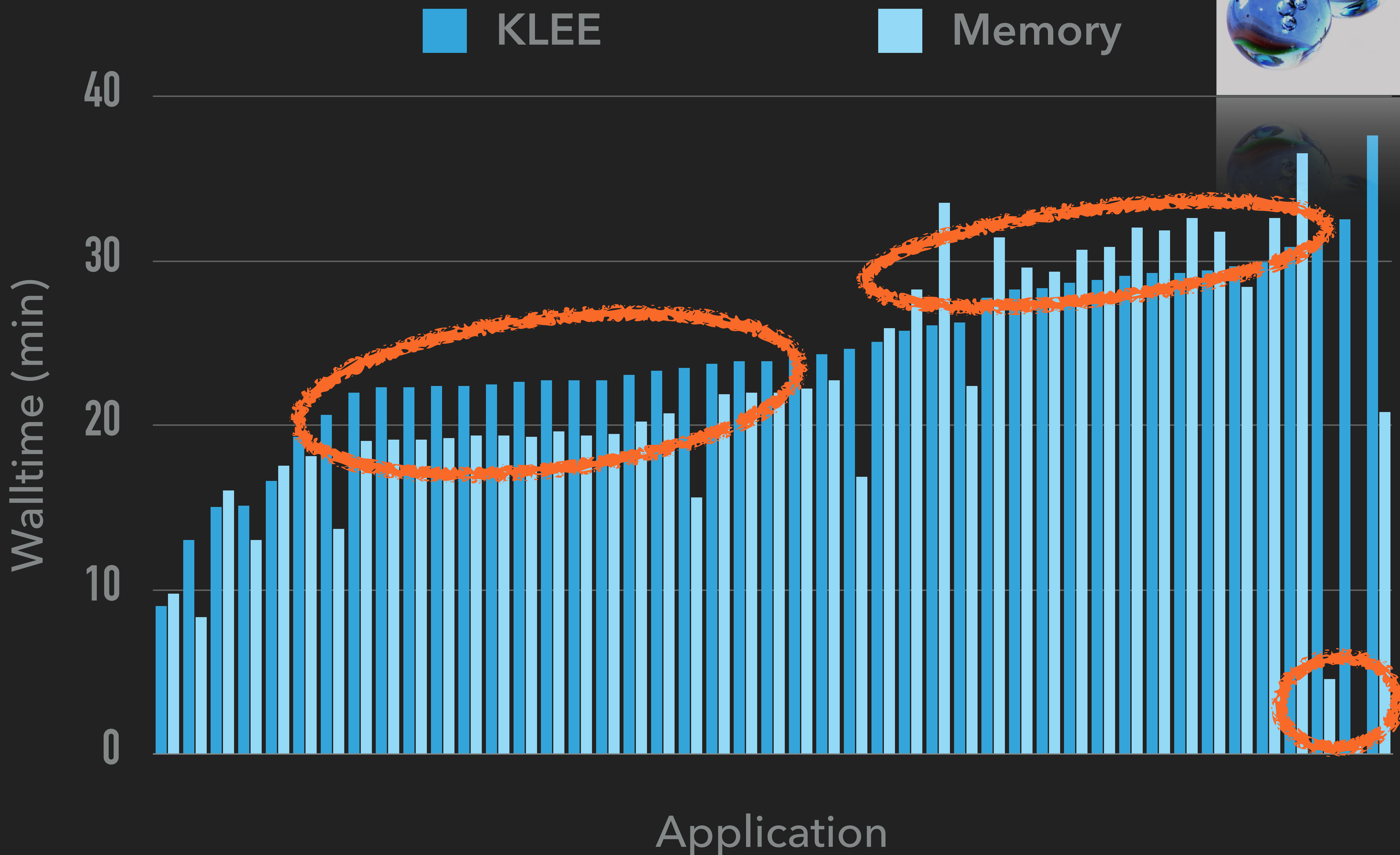
Breadth-First

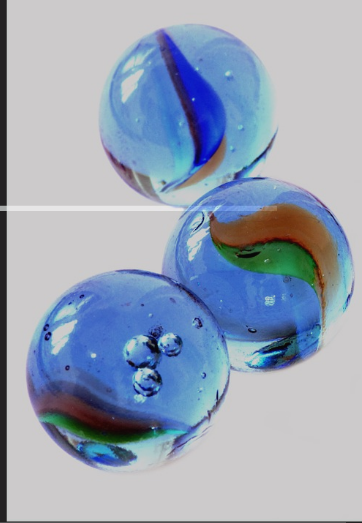


Random + Target Uncovered

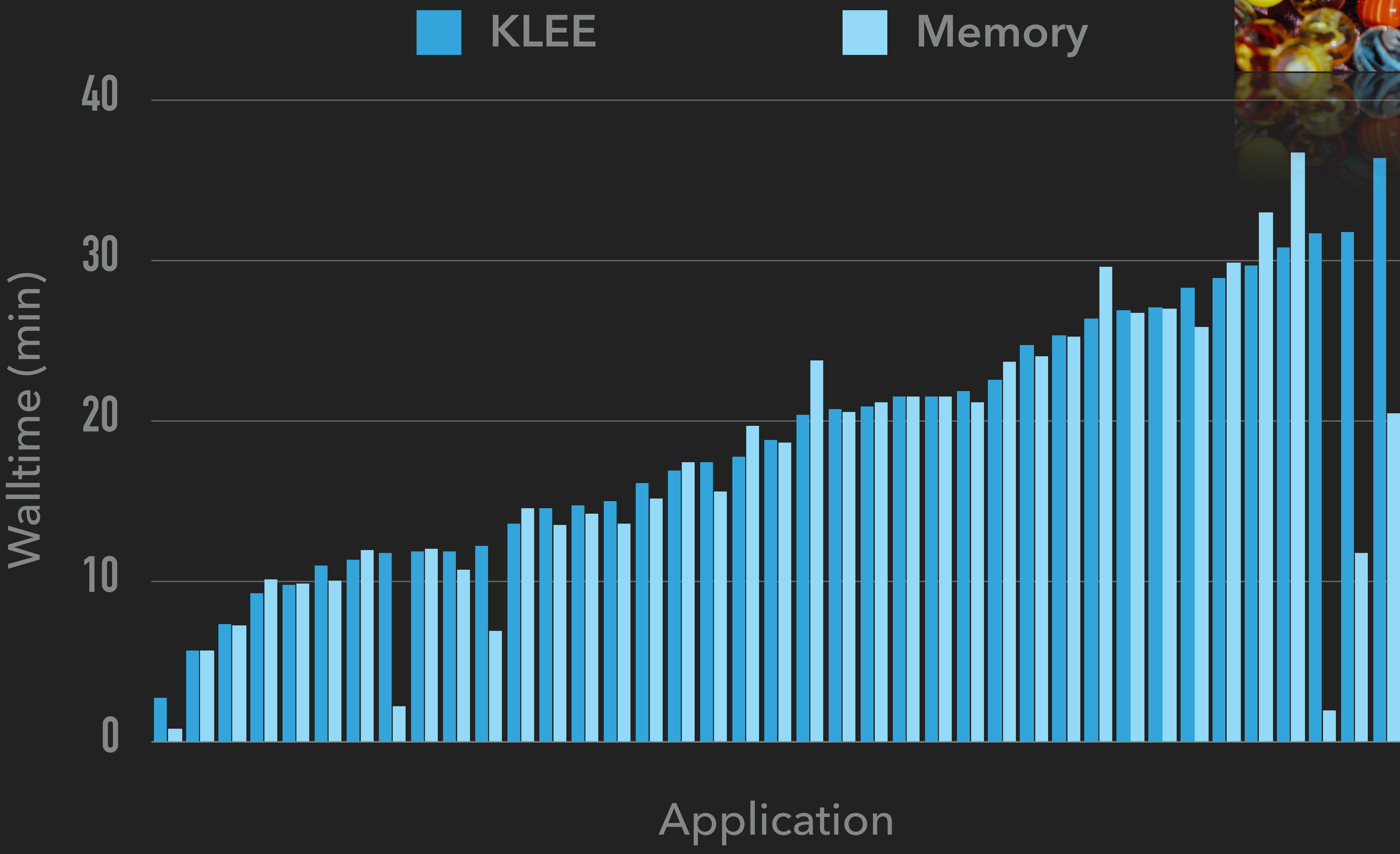
RQ1: CHANGES IN EXECUTION TIME

WALLTIME - DEPTH FIRST SEARCH



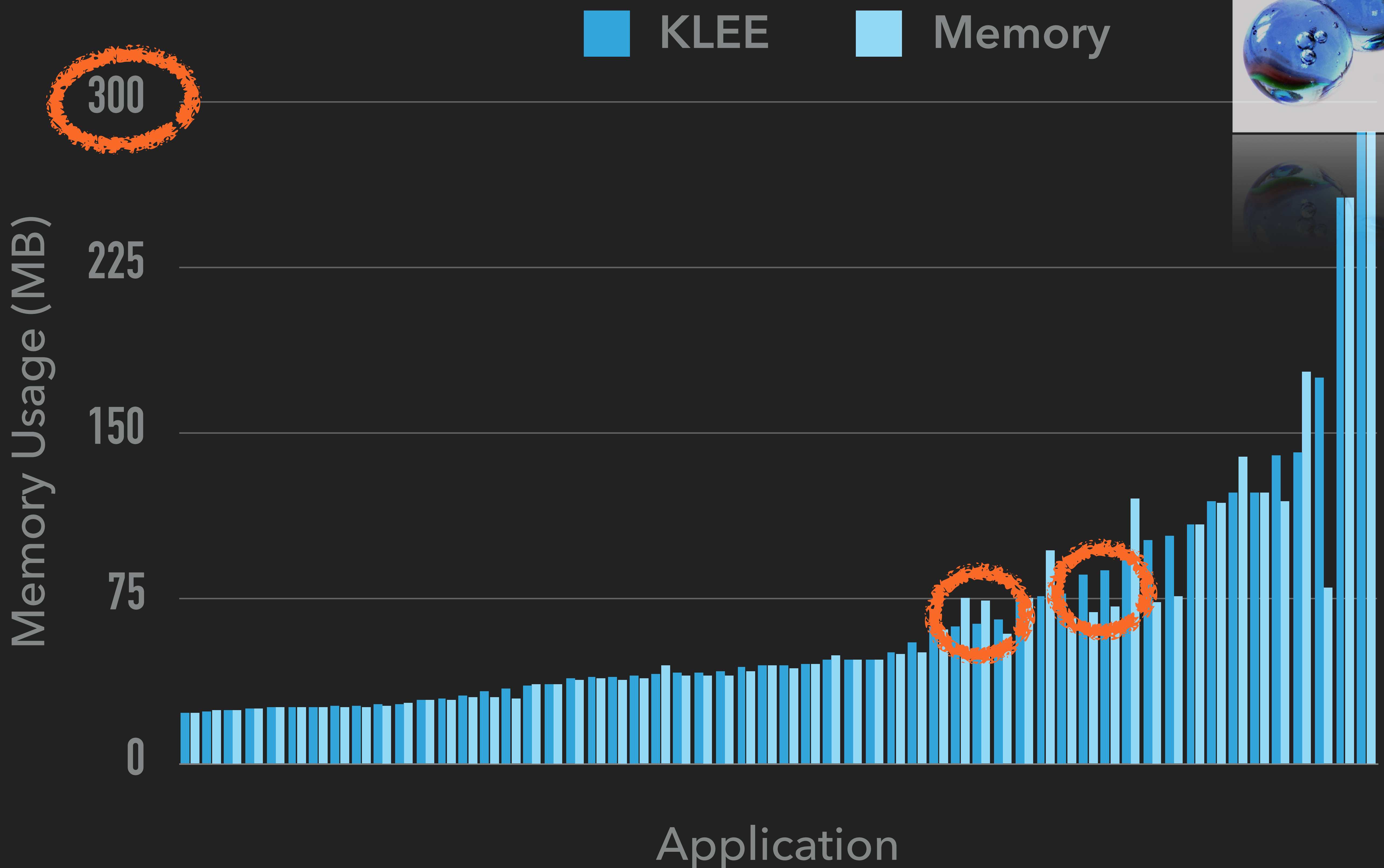


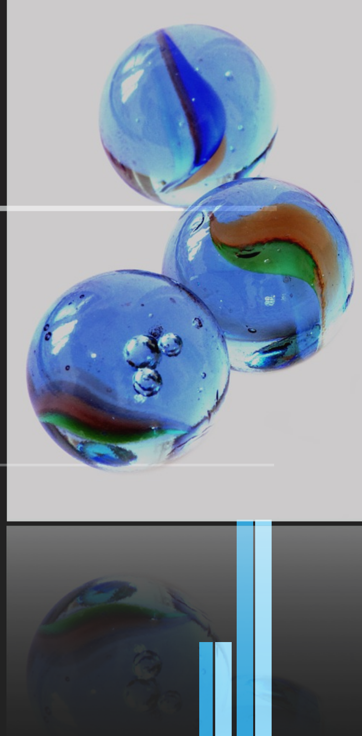
WALLTIME - BREADTH FIRST SEARCH



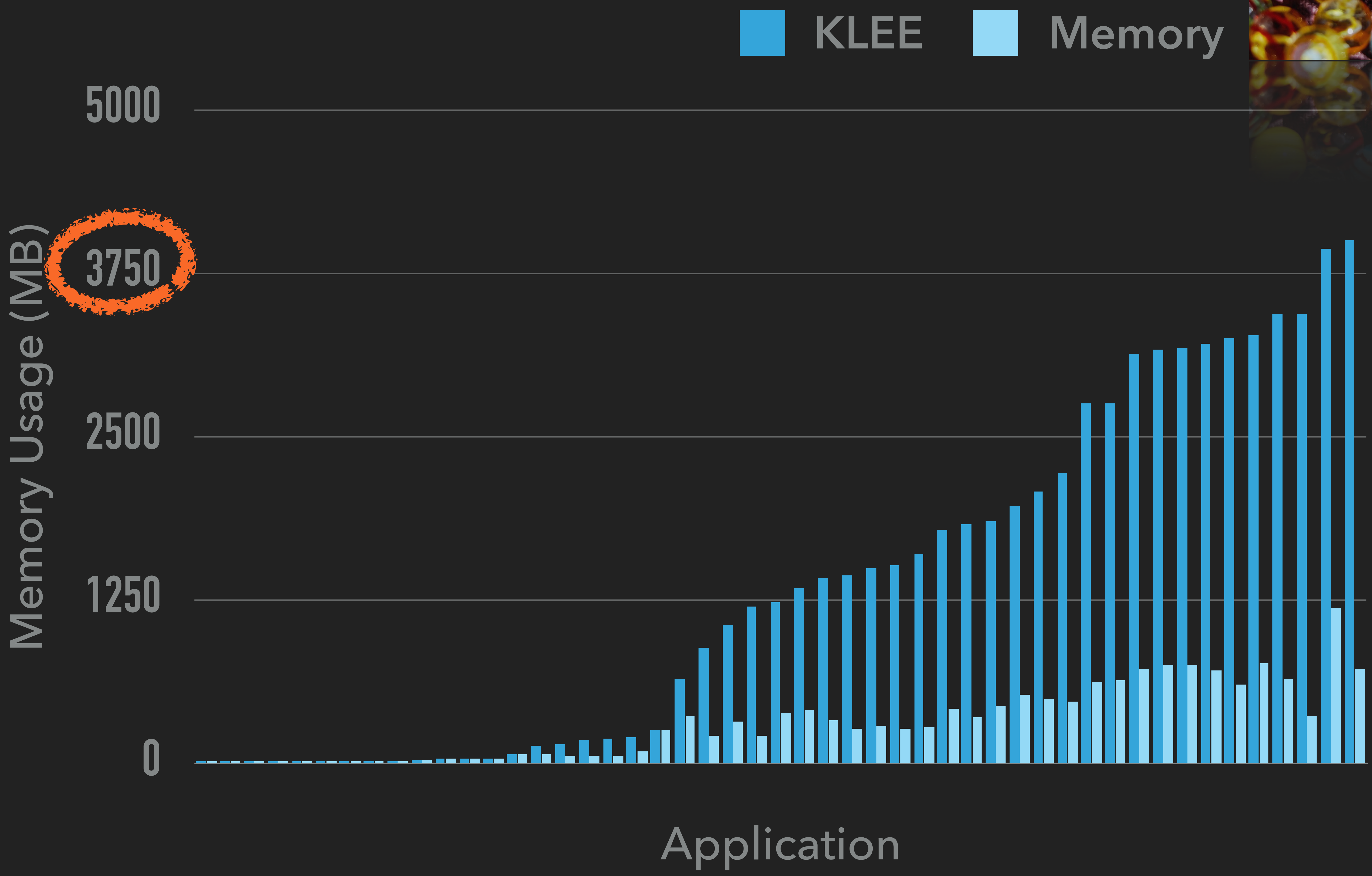
RQ2: CHANGES IN MEMORY CONSUMPTION

MEMORY USAGE - DEPTH FIRST SEARCH

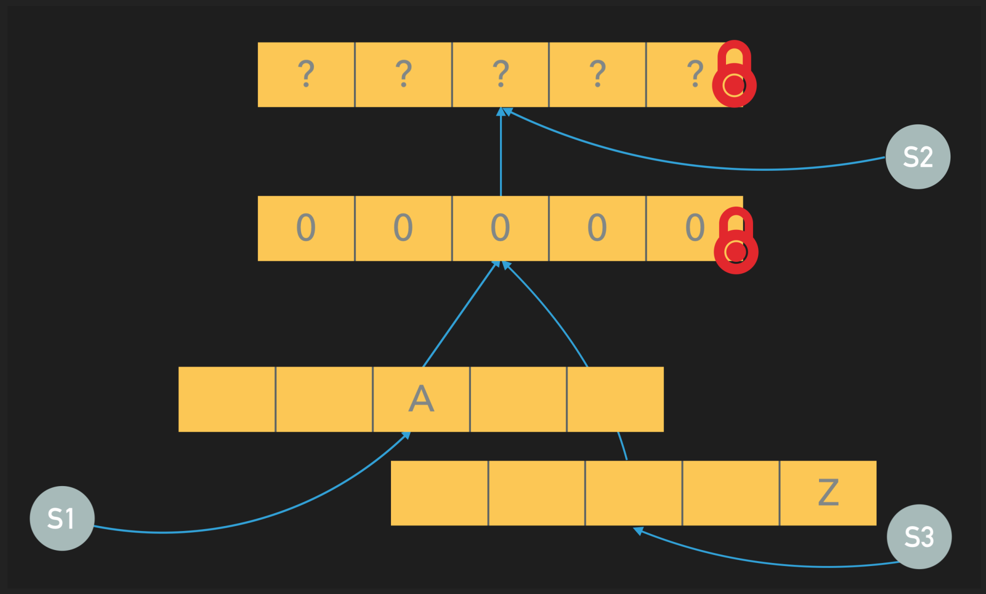


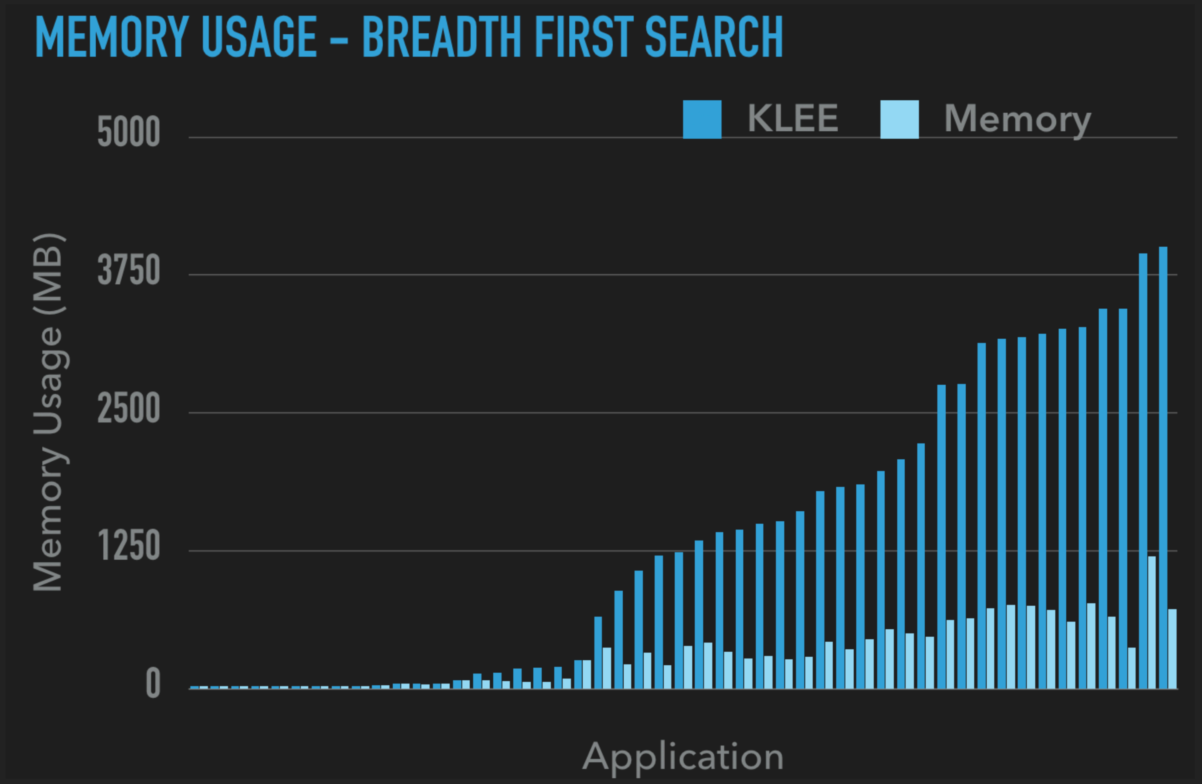


MEMORY USAGE - BREADTH FIRST SEARCH



SUMMARY





THIS RESEARCH HAS BEEN SUPPORTED BY: UK EPSRC VIA GRANT EP/N007166/1, EP/R011605/1

OBJECT STATE HASHING

$HS := I_1 \oplus V_1 \oplus \ldots \oplus I_n \oplus V_n$

HS_{prev} `0 , 0 , 0 , 0`

$HS := HS_{prev} \oplus 0 \oplus A$ `A`

DIFFERENT STRUCTURE – SAME SEMANTIC (HS_2 == HS)

`A , A , A , A`

$HS_2 := \ldots$ `0 , 0 , , 0`